Computer Vision and Image Understanding 192 (2020) 102899

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

Graph-matching-based correspondence search for nonrigid point cloud registration ☆

Seunggyu Chang [a], Chanho Ahn [a], Minsik Lee [b,*], Songhwai Oh [a]

[a] Department of Electrical and Computer Engineering and ASRI, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea
[b] Division of Electrical Engineering, Hanyang University, 55 Hanyangdaehak-ro, Sangnok-gu, Ansan 15588, Republic of Korea

ARTICLE INFO

Communicated by Nikos Paragios

MSC:
65D19
68R10
05C70

Keywords:
Nonrigid registration
Graph matching
Point cloud
Mesh

ABSTRACT

Nonrigid registration finds transformations to fit a source point cloud/mesh to a target point cloud/mesh. Most nonrigid registration algorithms consist of two steps; finding correspondence and optimization. Among these, finding correspondence plays an important role in registration performance. However, when two point clouds have large displacement, it is hard to know correct correspondences and an algorithm often fails to find correct transformations. In this paper, we propose a novel graph-matching-based correspondence search for nonrigid registration and a corresponding optimization method for finding transformation to complete nonrigid registration. Considering global connectivity as well as local similarity for the correspondence search, the proposed method finds good correspondences according to semantics and consequently finds correct transformations even when the motion is large. Our algorithm is experimentally validated on human body and animal datasets, which verifies that it is capable of finding correct transformations to fit a source to a target.

1. Introduction

Point cloud registration is a key process for capturing full shapes of 3D objects. Capturing the point cloud of a 3D object is a fundamental process for computer vision and robotics problems such as 3D localization, 3D scene/object recognition, 3D reconstruction in *augmented/virtual reality* and generating free-viewpoint videos. However, capturing a high-quality 3D object in a single shot is challenging due to missing points occluded by other parts or sparse points in parts captured from distant surfaces. These are inevitable shortcomings we face in acquiring 3D data. Registration of simultaneously captured multi-view point clouds can complement missing and low-resolution regions, however, this is not enough for capturing the hidden sides of an object, such as an armpit or a thigh of a human body. On the other hand, when an object is deformable, nonrigid registration can supplement these by combining point clouds of different frames representing different poses of the object. For example, a hidden or missing human body part of one frame can be complemented by another frame in which the corresponding part is exposed to the camera.

Nonrigid registration can transform another point cloud of the same deformable object into the reference point cloud. Nonrigid registration algorithms generally consist of two procedures, one is finding correspondences and the other is updating parameters in the way that reduces some metric between the correspondence pairs. The difference of nonrigid registration from rigid registration is that nonrigid registration finds transformation parameters for each point to describe the motion to fit a source frame to a target frame whereas rigid registration finds one set of transformation parameters. Hence, for nonrigid registration, it is important to find a correct match for every point to determine where to transform it. In order to acquire a high-quality 3D shape from a deformable object, there are two requirements: A good registration algorithm that can find correct matches between two point clouds of an object that has large motions, and an ability to process two massive point clouds containing hundreds of thousands of points. However, satsifying both conditions is difficult for nonrigid registration.

The nonrigid iterative closest point (nonrigid ICP) method, the most popular nonrigid registration technique, finds correspondences as the closest points in every iteration hoping to find correct matches in the end. However, nearest search based on point position are sensitive to noises. Various feature extraction methods like Shape Contexts (SC) (Belongie et al., 2002) and Point Feature Histogram (PFH) (Rusu et al., 2008a,b) are developed to improve correspondence search by characterizing local surface and shape of each points. Further, Chui and Rangarajan (2003) relax binary correspondence condition of ICP

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.cviu.2019.102899.

* Corresponding author.
E-mail address: mleepaper@hanyang.ac.kr (M. Lee).

https://doi.org/10.1016/j.cviu.2019.102899
Received 16 November 2018; Received in revised form 26 December 2019; Accepted 28 December 2019
Available online 2 January 2020

to soft-assignment. On the other hand, methods to find robust transformations from erroneous matching are developed. Ma et al. (2014) remove outliers from a putative correspondence set containing erroneous matching and Ma et al. (2015, 2013) estimate transformation parameters from erroneous matching using robust L_2E estimator. However, the performance degrades when an underlying motion of two point sets is large and often computing features for massive point sets are inexecutable due to memory limit. Another main stream of nonrigid registration is to formulate registration problem as a probability density estimation problem modeled as Gaussian Mixture Models (Jian and Vemuri, 2010; Wang et al., 2015, 2014; Myronenko and Song, 2010). These approaches are proposed to handle noises, missing, and outliers, however, they are prone to errors when two point clouds under registration are related to large motion.

In this work, we propose a novel correspondence matching and initial alignment methods for nonrigid registration based on graph matching. Graph matching finds correspondences of vertices of two graphs by maximizing similarities between vertices and edges of the two graphs. This makes the matching process consider neighborhood connectivity, which contains both the local and global information. In order to incorporate graph matching in nonrigid registration, we first divide point clouds into a number groups and apply graph matching to find groupwise correspondences. In addition, we relax one-to-one matching problem to many-to-one matching problem to improve registration performance. As a consequence, our method can effectively match semantically identical human body parts of two frames even when the motions are large. Furthermore, the search based on a small number of point groups reduces memory usage compared to other nonrigid registration methods based on dense features, which makes our method favorable in handling large-scale point clouds. Using the matching results obtained from graph matching, we compute groupwise transformation parameters to find initial nonrigid alignment. These initial alignments are coarse but they provide great clues for the following refinement step. We evaluate our methods on the Lincoln human motion dataset (Collet et al., 2015) and the TOSCA animal dataset (Bronstein et al., 2008). In short, the contribution of the paper can be summarized as follows:

- We propose a novel graph-matching-based correspondence search method for nonrigid registration, which provide coarsely aligned point cloud even when the underlying motion is large.
- We propose many-to-one matching problem for nonrigid registration and its solver to enhance robustness of matching and empirically show that many-to-one matching preserves more local structures.
- We propose an initial alignment method which refines the rough registration results produced by graph matching procedure.
- We propose a nonrigid registration framework which can handle large-scale point clouds for high-quality 3D reconstruction.

The rest of the paper is organized as follows. Short review of a literature are introduced in 2. The proposed graph-matching-based nonrigid registration method is explained module by module in Section 3. Qualitative and quantitative experiments on human body and animal body datasets are conducted in Section 4, followed by conclusion and future work in Section 5.

2. Related work

In this section we briefly review the background material our work is based on.

2.1. Feature extraction and correspondence search

Many of nonrigid registration algorithms are composed of two steps, and the first one is to find correspondence point to which a point in a source frame move. To find the correspondences, various feature extraction methods are introduced. Basic physical properties like surface normals and curvatures are used as point features in (Tevs et al., 2009; Anguelov et al., 2005). Laplacian convolution filters (Allen et al., 2002) incorporate edge information into features, making matches more feasible. Integral features (Huang et al., 2006) incorporate multiscale features whereas (Liao et al., 2009) applies scale-invariant feature transform (SIFT) to perform scale invariant matchings in 3D. Belongie et al. (2002) proposed shape context (SC), which is a histogram of relative coordinates of neighbors capturing local shape of a point and later it is further developed to point feature histogram (PFH) (Rusu et al., 2008a,b) to include more physical properties of a local 3D shape. Because PFH is fundamentally a statistic, it is suitable for extracting features from an orderless point cloud and has gained lots of attentions for matching point clouds and meshes. Fast point feature histogram (FPFH) (Rusu et al., 2009) improves PFH to reduce its computation time. However, features extraction methods introduced hitherto represent only local information and lack global information nor neighborhood information, because the features are computed withing a bounded region of a point of interest.

Another strategy is to use a two-step process to find robust correspondences: (i) computing a putative correspondence set which may contain some erroneous matches, and (ii) removing erroneous matches using some constraints. The RANSAC (Fischler and Bolles, 1981) and its variants (Raguram et al., 2008) repeatedly generates solutions from small randomly sampled subsets of a putative match set and filter out outliers by testing each solution. Ma et al. (2014) proposes the vector field consensus (VFC) to estimate a consensus of inliers from a putative correspondence set by interpolating a vector field.

2.2. Nonrigid registration

Iterative closest point (ICP) (Besl and McKay, 1992) based nonrigid registration is one of the most popular point registration algorithm. ICP iterates two sub steps, finding correspondence points by nearest neighbor and updates transform parameters in the sense of least squares. However, due to binary correspondence assumption, ICP is sensitive to noise, missing, and outliers. Chui and Rangarajan (2003) relax binary correspondence condition of ICP to soft-assignment to estimate robust transform parameters. Tsin and Kanade (2004) find transformation parameters by maximizing kernel correlation (KC) which considers all point relations into KC calculation and acts as a robust M-estimator (Huber, 1981). Ma et al. (2015) proposed RPM-L2E to estimate robust transformation parameters from a erroneous matching set using L_2E estimator. Many of nonrigid registration algorithms formulate alignment of two point sets into probability density estimation problem, especially using Gaussian Mixture Models (GMMs) (Jian and Vemuri, 2010). Assuming that points in one point cloud is normally distributed to the other point cloud, they substitute point assignment problem as parameter estimation problem of a mixture distribution. Myronenko and Song (2010) proposed coherent point drift (CPD) algorithm which forces GMM centroids move coherently based on motion coherence theory (Yuille and Grzywacz, 1989). To capture spatially asymmetric distribution, asymmetric Gaussian models are also proposed (Wang et al., 2014, 2015). Ma et al. (2016) proposed PR-GLS utilizing local features to GMMs to preserve local and global structures with an assumption that transformation is lying reproducing kernel Hilbert space (RKHS). This RKHS assumption of a transformation space and following Tikhonov regularization are popularly applied to other algorithms to robustly estimate transformation (Wang et al., 2016; Ma et al., 2018). Wang et al. (2016) and (Ma et al., 2018) proposed context-aware Gaussian fields (CA-LapGF) and proposed robust point matching using manifold regularization (MR-RPM) respectively to utilize intrinsic geometry of point sets by applying graph Laplacian as a manifold regularizer.

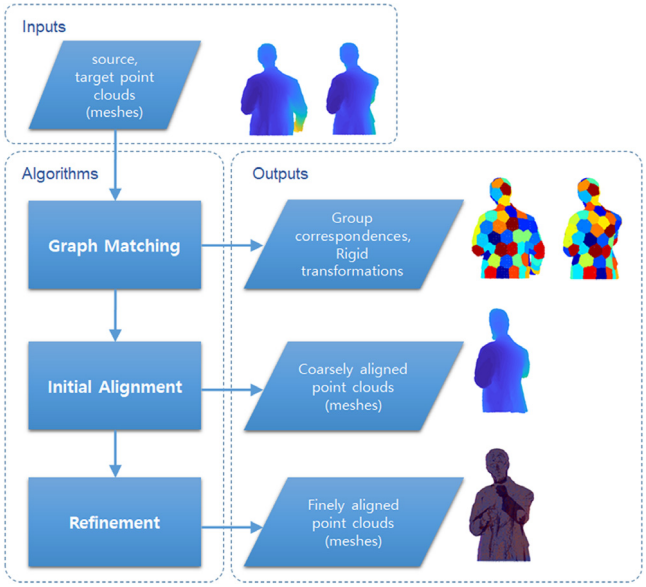


Fig. 1. Overall work flow of graph-matching-based correspondence search for nonrigid point cloud registration.

3. Proposed method

The proposed graph-matching-based nonrigid registration method consists of three main modules: First, the graph matching module divides point clouds of two frames, i.e., source and target, into groups, and finds the correspondences between them under the assumption that the motion of each group can be approximated as a rigid transform. This module inevitably incurs some errors due to some incorrect matches or the rigid approximation, so the subsequent modules refine the transformed source point cloud. The second module, the initial alignment module, relaxes the rigid transforms into affine transforms to allow more flexibility and refine them. In this module, potential mismatches between groups are also corrected. In the third module, to further refine the transformed point cloud, a traditional nonrigid ICP algorithm is applied to yield the final nonrigid registration result. The overall process is illustrated in Fig. 1. We assume that all transformation parameters and points appearing in this section are based on homogeneous coordinates.

3.1. Graph matching

The first module of the proposed algorithm is to find the group correspondences of two point clouds and their group-wise rigid motions. We assume that a nonrigid motion can be approximated as a collection of rigid motions of small sub-parts. As the purpose of this module and the following smooth alignment module is to generate a coarse alignment for the later refinement, this approximation is good enough for our purpose. Inputs of this module are two point clouds $\mathcal{P}^S$ and $\mathcal{P}^T$, i.e., source and target, respectively. We divide each point cloud into K groups of points and find correspondences between the source point-group and the target point-group. We formulate the correspondence search as a graph matching problem because graph matching can consider local and global information simultaneously based on vertex similarities and edge similarities. Graph matching works well for matching a nonrigid deformable shape like an articulated human body.

Graph matching is a problem of finding matches between vertices of two graphs. Let $G^S = (\mathcal{V}^S, \mathcal{E}^S)$ and $G^T = (\mathcal{V}^T, \mathcal{E}^T)$ denote source and target graphs respectively. Each graph consists of vertices $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^N$, where N is the number of vertices, and edges, $\mathcal{E} = \{(\mathbf{v}_i, \mathbf{v}_j) | \mathbf{v}_i, \mathbf{v}_j \in \mathcal{V}, \mathbf{v}_i \text{ and } \mathbf{v}_j \text{ are connected}\}$. Graph matching finds a one-to-one map $\phi : \mathcal{V}^S \longrightarrow \mathcal{V}^T$ that maximizes the following equation provided that the G^S and G^T have the same number of vertices; i.e., $N^S = N^T$.

$$\sum_{v \in \mathcal{V}^S} S_v\Big(\mathbf{v}, \phi(\mathbf{v})\Big) + \sum_{(\mathbf{u},\mathbf{v}) \in \mathcal{E}^S} S_e\Big((\mathbf{u}, \mathbf{v}), \big(\phi(\mathbf{u}), \phi(\mathbf{v})\big)\Big) \tag{1}$$

where $S_v(\cdot,\cdot)$ and $S_e(\cdot,\cdot)$ denote vertex similarity and edge similarity which scores high value when inputs pairs are similar. To simplify Eq. (1), an affinity matrix $\mathbf{M} \in \mathbb{R}^{N^S N^T \times N^S N^T}$ is introduced, where the elements are computed as follows:

$$\mathbf{M}(c_{i_1 j_1}, c_{i_2 j_2}) = \begin{cases} S_v(\mathbf{v}_{i_1}, \mathbf{u}_{j_1}), \\ \qquad \mathbf{v}_{i_1} \in V^S,\ \mathbf{u}_{j_1} \in \mathcal{V}^T, \\ \qquad \text{if } i_1 = i_2,\ j_1 = j_2, \\ S_e\big((\mathbf{v}_{i_1}, \mathbf{v}_{i_2}), (\mathbf{u}_{j_1}, \mathbf{u}_{j_2})\big), \\ \qquad \mathbf{v}_{i_1}, \mathbf{v}_{i_2} \in \mathcal{V}^S, \mathbf{u}_{j_1}, \mathbf{u}_{j_2} \in \mathcal{V}^T, \\ \qquad (\mathbf{v}_{i_1}, \mathbf{v}_{i_2}) \in \mathcal{E}^S,\ (\mathbf{u}_{j_1}, \mathbf{u}_{j_2}) \in \mathcal{E}^T, \\ \qquad \text{if } i_1 \neq i_2, j_1 \neq j_2, \\ 0, \qquad \text{otherwise,} \end{cases} \tag{2}$$

where $c_{ij} = (i-1)N^T + j$. Using the affinity matrix $\mathbf{M}$, graph matching can be reformulated as the following quadratic assignment problem (QAP), which finds a permutation matrix $\mathbf{P}$ instead of ϕ (Loiola et al., 2007).

$$\begin{aligned} &\underset{\mathbf{P}}{\text{maximize}} \quad vec(\mathbf{P})^T\, \mathbf{M}\, vec(\mathbf{P}) \\ &\text{subject to} \quad \mathbf{P}\mathbf{P}^T = \mathbf{I},\ \mathbf{P} \in \{0,1\}^{N^S \times N^T} \end{aligned} \tag{3}$$

where $vec(\cdot)$ denotes the vectorization operator.

We will describe how to construct graphs from two point clouds and how to define a similarity function between two graphs to make graph matching results consistent with underlying semantics. The similarity function is defined to enforce a smooth-motion constraint on neighboring points, which is commonly used in nonrigid ICP.

3.1.1. Graph construction

To construct a graph from a point cloud, we divide two point clouds P^S and P^T into K groups respectively. Simple clustering algorithms such as K-means or farthest point sampling with a nearest-neighbor assignment can be used for grouping. The K clusters then become K vertices of a graph. For edge construction, the simplest way is to connect all pairs of vertices in the graph, which is called a fully-connected graph. However, a fully-connected graph contains redundant edges that have semantically less or no meaning, e.g., an edge between groups located on a head and a foot in case of a human body. These redundant edges increase computation time and may act as obstacles in finding correct matches. If this is the case, we can consider connecting an edge only when two groups adjoin. Constructing a graph in this manner is a direct conversion of a physical surface into a graph, which will be referred to as a *naïve graph* in rest of the paper. However, a naïve graph sometimes produces an unfavorable matching result for nonrigid registration. To begin with, we need to define some terminologies for a graph matching result.

Definition 1. Let $G_1 = (\mathcal{V}^1, \mathcal{E}^1)$ and $G_2 = (\mathcal{V}^2, \mathcal{E}^2)$ be two graphs. An edge $(\mathbf{u}, \mathbf{v}) \in \mathcal{E}^1$ is called a **non-matched edge** if $\big(\phi(\mathbf{u}), \phi(\mathbf{v})\big) \notin \mathcal{E}^2$

Definition 2. Let $G^n = (\mathcal{V}, \mathcal{E}^n)$ be a graph. G^n is called an **n-step graph** if $\forall (\mathbf{u}, \mathbf{v}) \in \mathcal{E}^n$, $\mathbf{u}$ is reachable from $\mathbf{v}$ along the edges whose number is less than or equal to n in a naïve graph $G^1 = (\mathcal{V}, \mathcal{E}^1)$ whose vertices are the same as vertices of G^n.

In general, graph matching is applied on two graphs having the same structure. Then the result of a graph matching is a one-to-one correspondence map ϕ of vertices. From the matching result ϕ, we can think of correspondences between edges in accordance with ϕ. If the two graphs have the same structure, then the correspondence map

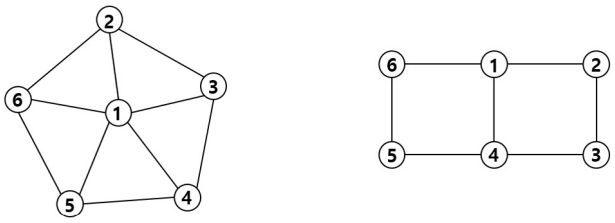


Fig. 2. An example of non-matched edges when matching two graphs with different structures.

between edges induced by ϕ will be also a one-to-one map. However, when two graphs have different structures, there occur edges in the source graph which cannot be mapped to the target graph. We define the term *non-matched edge* to represent the edges that cannot be mapped to the target graph. Physically, a non-matched edge can be interpreted as a tearing of a surface if naive-graphs are used in graph matching, and sometimes it is not reasonable in reality. However, non-matched edges are inevitable when the number of edges in the source graph is larger than the number of edges in the target graph. Fig. 2 illustrates such an example. The numbers in the vertices of two graphs represent matching pairs. An edge between vertex number 5 and vertex number 1 is a non-matched edge because the two vertices are connected in the source graph on the left, whereas they are not connected in the target graph on the right. It is preferable to construct a graph to reduce the number of non-matched edges. In addition, even when there are non-matched edges, it is desirable to have a matching result that the distances between their end vertices are small in the target graph. Empirically, we found out that constructing a graph as a 2-step graph leads to a reasonable match.

3.1.2. Similarity functions

To compute an affinity matrix between two graphs, similarity functions should be designed. Similarity functions play an important role in graph matching. We assume that the whole nonrigid motion is approximated as a set of rigid motions of grouped points. Hence, we first compute a rigid motion for every pair of groups between the source and target point clouds, using the rigid ICP algorithm. If two groups in a pair come from the same part of an object, then a large part the source group will overlap with the target group when the source group is aligned with the optimal rigid transformation. Then the distance between the transformed source group and the target group will be low. Otherwise, if the two groups in a pair come from different parts of the object, the distance between the transformed source group and the target group will be high. As distance are inversely proportional to similarity, we define the vertex similarity function as follows:

$$S_v(\mathbf{v}_i, \mathbf{u}_j) = exp\Big(-\lambda_v \; dist\; \big(T_{ij}(\mathbf{P}_i^S), \mathbf{P}_j^T\big)\Big) \tag{4}$$

where $\mathbf{v}_i \in \mathcal{V}^S$ and $\mathbf{u}_j \in \mathcal{V}^T$, $T_{ij}(\cdot)$ denotes the optimal rigid transform of point group represented by $\mathbf{v}_i$ to point group represented by $\mathbf{u}_j$, $\mathbf{P}_i^S$ denotes matrix representation of points in the ith group of the source and $\mathbf{P}_j^T$ denotes matrix representation of points in the jth group of the target. As vertices of our graph represent grouped points, $dist(\cdot)$ should be a set distance measure and we choose the normalized *Chamfer* distance as the distance function. The rigid transform $T_{ij}(\cdot)$ can be rewritten as $T_{ij}(\mathbf{X}) = \mathbf{R}_{ij}(\mathbf{X} - \mathbf{C}_i) + \mathbf{C}_i + \mathbf{b}_{ij}$ when $\mathbf{X}$ is a matrix representation of points and $\mathbf{R}_{ij}$ and $\mathbf{b}_{ij}$ are rotation and translation parameters of the rigid motion, respectively. $\mathbf{C}_i$ is a centroid matrix whose columns are the same as the centroid of the ith group.

In case of edge similarity, as discussed in the previous section, matched groups in the target point cloud, who are adjacent in the source point cloud, should be close to each other. If this is the case, the transformed point group $\mathbf{P}_{i_2}^S$ based on the rigid motion $T_{i_1 j_1}$ of neighboring groups will not distant apart from that based on its own rigid motion, $T_{i_2 j_2}$, when $(\mathbf{v}_{i_1}, \mathbf{v}_{i_2}) \in \mathcal{E}^S$ and $(\mathbf{u}_{j_1}, \mathbf{u}_{j_2}) \in \mathcal{E}^T$. Accordingly, similar to the vertex similarity function, the edge similarity function is designed as follows:

$$S_e\Big((\mathbf{v}_{i_1}, \mathbf{v}_{i_2}),\; (\mathbf{u}_{j_1}, \mathbf{u}_{j_2})\Big) = exp\Big(-\lambda_e \; dist\big(T_{i_1 j_1}(\mathbf{P}_{i_2}^S),\; T_{i_2 j_2}(\mathbf{P}_{i_2}^S)\big)\Big) \tag{5}$$

3.1.3. Many-to-one matching

The original graph matching problem finds one-to-one correspondence between two graphs. However, in our registration problem, a one-to-one matching may cause non-matched edges and accordingly, deteriorates quality of registration. For registration purpose, disregarding the one-to-one correspondence assumption, which is unusual for a graph matching problem, can result in better quality of registration. For example, in Fig. 2, if vertex number 5, along with vertex number 4, of the first graph matches to vertex number 4 of the second graph, all neighbors are preserved for the matched vertices of two graphs. Discarding the one-to-one constraint can help reduce the number of non-matched edges. Hence, we propose a many-to-one-relaxed graph matching problem and its solver for the nonrigid registration problem. The original graph matching problem (3) can be written in general form as below:

$$\begin{aligned} \underset{\mathbf{x}}{\text{maximize}} \quad & \mathbf{x}^T \mathbf{M}\mathbf{x} \\ \text{subject to} \quad & \mathbf{x} \in \Pi, \end{aligned} \tag{6}$$

where $\mathbf{M}$ is a similarity matrix and Π indicates a one-to-one matching solution space, which is obviously a subset of $\mathbb{Z}^{N^S N^T}$, where $\mathbb{Z}$ indicates the set of integers. If we relax Π to have many-to-one matching, the problem (6) becomes binary quadratic programming (BQP) (Merz and Freisleben, 2002). BQP is known to be NP-hard (Garey, 1979) and solved with diverse heuristics (Katayama and Narihisa, 2001; Merz and Freisleben, 2002). In this section, we propose a method to solve this BQP as alternating sequences of linear programming(LP)-relaxed sub-problems.

To admit many-to-one matching as a solution, we relax the one-to-one and integer constraints of Π in Eq. (6) into Π_1 as follows:

$$\Pi_1 = \{\mathbf{x} | (\mathbf{I}_{N^S} \otimes \mathbb{1}^{1\times N^T})\mathbf{x} = \mathbb{1}^{N^S \times 1},\; \mathbf{0} \le \mathbf{x} \le \mathbb{1}^{N^S N^T \times 1}\}, \tag{7}$$

where $\otimes$ denotes Kronecker product, $\mathbf{I}_n$ an $n \times n$ identity matrix, and $\mathbb{1}$ a matrix whose elements are all one. Although, it seems that Π_1 admits a real number solution, when Π_1 is used as a constraint of an LP, the optimal solution will be an integer vector.

Theorem 1. *A matrix* $\mathbf{A}$ *is totally unimodular (TU) if every square submatrix of* $\mathbf{A}$ *has determinant* $+1, -1$ *or* 0. *If* $\mathbf{A}$ *is TU and* $\mathbf{b}$ *is integral, then linear programs of forms like* $\{\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} | \mathbf{A}\mathbf{x} \ge \mathbf{b},\; \mathbf{x} \ge \mathbf{0}\}$ *or* $\{\max_{\mathbf{x}} \mathbf{c}^T\mathbf{x} | \mathbf{A}\mathbf{x} \le \mathbf{b}\}$ *have integral optima, for any* $\mathbf{c}$.

Theorem 1 is a well-known property of LP to ease solving integer programming (IP) (Wolsey, 1998). As the constraint matrix of Π_1 is obviously a TU matrix, we can utilize Theorem 1 to ease the problem. Hence, we reformulate the problem so that it is an LP for each variable as follows:

$$\begin{aligned} \underset{\mathbf{x},\mathbf{y}}{\text{maximize}} \quad & \mathbf{x}^T \mathbf{M}\mathbf{y} \\ \text{subject to} \quad & \mathbf{x}, \mathbf{y} \in \Pi_1, \quad \mathbf{x} = \mathbf{y}. \end{aligned} \tag{8}$$

However, we cannot use problem (8) directly for an alternating optimization, due to the equality constraint. In order to resolve this issue, we relaxe the equality constraint into an additional objective term as follows:

$$\begin{aligned} \underset{\mathbf{x},\mathbf{y}}{\text{maximize}} \quad & L(\mathbf{x}, \mathbf{y}, \lambda) = \mathbf{x}^T (\mathbf{M} + \lambda \mathbf{I})\mathbf{y} \\ \text{subject to} \quad & \mathbf{x}, \mathbf{y} \in \Pi_1, \end{aligned} \tag{9}$$

where $\mathbf{I}$ is a identity matrix of which the size is equal to that of $\mathbf{M}$, and λ is a penalty parameter. When λ is large enough, the optimal solution of problem (9) can be proved to be exactly the same as the optimal solution of problem (8).

Theorem 2. *The optimal solution of problem (9) is the same as the optimal solution of problem (8) if* $\lambda \geq \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T\mathbf{M}\mathbf{y} - \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y}$*, where* $C_1 = \{(\mathbf{x},\mathbf{y}) \in \mathbb{Z}^{n^2} \times \mathbb{Z}^{n^2} | \mathbf{x},\mathbf{y} \in \Pi_1,\ \mathbf{0} \leq \mathbf{x} \leq \mathbb{1}\}$ *and* $C_2 = \{(\mathbf{x},\mathbf{y}) \in \mathbb{Z}^{n^2} \times \mathbb{Z}^{n^2} | \mathbf{x},\mathbf{y} \in \Pi_1, \mathbf{x} = \mathbf{y},\ \mathbf{0} \leq \mathbf{x} \leq \mathbb{1}\}$.

Proof. Due to the constraint $\mathbf{x} \in \Pi_1$ and Theorem 1, $\mathbf{x}$ can only have either 0 or 1 as its elements and the number of elements whose values are 1 should be N^S. If $\lambda \geq \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T\mathbf{M}\mathbf{y} - \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y}$, the optimal solution of problem (9) is in C_2 as a result of Lemma 1. Hence, $\mathbf{x} = \mathbf{y}$ and $\mathbf{x}^T\mathbf{y} = \mathbf{x}^T\mathbf{x} = N^S$, which is a constant. Thus, the objective function of problem (9) becomes $\mathbf{x}^T(\mathbf{M}+\lambda\mathbf{I})\mathbf{y} = \mathbf{x}^T\mathbf{M}\mathbf{y} + \lambda\mathbf{x}^T\mathbf{y} = \mathbf{x}^T\mathbf{M}\mathbf{y} + \lambda N^S$. Omitting the constant term, λN^S, in the objective function, we obtain problem (8). □

Lemma 1. *The optimal solution of problem (9) is in* C_2 *if* $\lambda \geq \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T\mathbf{M}\mathbf{y} - \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y}$.

Proof. If the optimal solution $(\mathbf{x}^*,\mathbf{y}^*)$ of problem (9) is in C_2, then it is trivial that the optimal solution of (9) is the same as the optimal solution of problem (8) because $\lambda\mathbf{x}^T\mathbf{y}$ is just a constant under the constraint C_2.

$$\begin{aligned} \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T(\mathbf{M}+\lambda\mathbf{I})\mathbf{y} &= \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y} + \lambda\mathbf{x}^T\mathbf{y} \\ &= \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y} + \text{Const} \end{aligned} \tag{a}$$

Otherwise, let us assume that the optimal solution $(\mathbf{x}^*,\mathbf{y}^*)$ of problem (9) is not in C_2, that is $(\mathbf{x}^*,\mathbf{y}^*) \in C_2 - C_1$. Then the following inequality holds:

$$\begin{aligned} \max_{(\mathbf{x},\mathbf{y})\in C_1-C_2} \mathbf{x}^T(\mathbf{M}+\lambda\mathbf{I})\mathbf{y} &\leq \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T\mathbf{M}\mathbf{y} + \max_{(\mathbf{x},\mathbf{y})\in C_1-C_2} \lambda\mathbf{x}^T\mathbf{y} \\ &< \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T\mathbf{M}\mathbf{y} + \max_{(\mathbf{x},\mathbf{y})\in C_2} \lambda(\mathbf{x}^T\mathbf{y} - 1) \\ &\leq \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T\mathbf{M}\mathbf{y} + \max_{(\mathbf{x},\mathbf{y})\in C_2} \lambda\mathbf{x}^T\mathbf{y} \\ &= \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T(\mathbf{M}+\lambda\mathbf{I})\mathbf{y} \end{aligned} \tag{b}$$

The -1 in the second row of (b) appears because $\mathbf{x}^T\mathbf{x}$ is at least larger than $\mathbf{x}^T\mathbf{y}$ by one due to the binary constraint. Applying the premise $\lambda \geq \max_{(\mathbf{x},\mathbf{y})\in C_1} \mathbf{x}^T M\mathbf{y} - \max_{(\mathbf{x},\mathbf{y})\in C_2} \mathbf{x}^T M\mathbf{y}$ on the $-\lambda$, we get third row of (b). The equality in the last row of (b) holds because $\lambda\mathbf{x}^T\mathbf{y}$ is just a constant under the constraint C_2. However, (b) contradicts the assumption that the optimal solution of problem (9) is not in C_2. □

Furthermore, to induce a reasonable many-to-one matching result, we replace some zero elements of the similarity matrix $\mathbf{M}$ in (2) into $S_e(\cdot,\cdot)$ assuming that there is a self-edge $(\mathbf{v}_{j_1},\mathbf{v}_{j_1}) \in \mathcal{E}^T$ for vertices that satisfy $i_1 \neq i_2, j_1 = j_2$ or $(\mathbf{v}_{i_1},\mathbf{v}_{i_1}) \in \mathcal{E}^S$ for vertices that satisfy $i_1 = i_2, j_1 \neq j_2$. We leave the elements whose corresponding vertices satisfy $i_1 = i_2, j_1 = j_2$ as $S_v(\cdot,\cdot)$, the original form in (2). We solve the problem (9) by solving an LP alternately for the two variables while gradually increasing λ to search for the desired solution as described in line 11–13 of Algorithm 1. The overall process of the graph matching module is described in Algorithm 1. Here, RigidICP($\mathcal{P}^S$, $\mathcal{P}^T$) denotes the rigid ICP function which outputs rotation and translation parameters to transform $\mathcal{P}^S$ to $\mathcal{P}^T$. $\bar{\mathbf{X}}_i$ denotes the transformation matrix of the ith group and $\mathbf{c}_i$ denotes the centroid of group i. Outputs of the graph matching module consists of three components: grouping results ($\mathcal{C}^S$, $id^S(\cdot)$), transformation results ($\{\bar{\mathbf{X}}_i\}_{i=1:K}$), and matching results (Φ). These three components are passed to the initial alignment module to form energy functions. Especially, the transformation results, $\{\bar{\mathbf{X}}_i\}_{i=1:K}$, become the baselines for the sparse regularizer in Section 3.2.3.

3.2. Generating initial alignment

In the graph matching module, we find group correspondences between the source point cloud and the target point cloud. Based on our assumption that the whole nonrigid motion of a point cloud can be approximated as a set of rigid motions of small point groups, we can find the corresponding rigid motion for each point group when a matching function ϕ is provided. If we compute the rigid motion T_{ij} for all pairs of the source point group and the target point group, where i, j are the group indices for the source and target respectively, we can say that the estimate for the true rigid motion of the ith point group of the source point cloud is $T_{i\,\phi(i)}$. Then, we can obtain a coarsely transformed point cloud by applying the corresponding rigid transform to each point group of the source point cloud. However, the resultant point cloud looks clumsy as in Fig. 3. Each transformed point group forms an island in the transformed point cloud and the whole shape looks like a carapace of a tortoise. This is because there is no constraint to smooth rigid motion parameters over neighboring groups. Further, we still have non-matched edges which look like a stretched cheese as in Fig. 3. To be a good initialization for nonrigid ICP, i.e., the final refinement module, we need to smooth the surface of the transformed point cloud and get rid of non-matched edges. To resolve this, we relax rigid transforms obtained from the graph matching module into affine transforms so that the point groups can be slightly stretched or shrinked to form a seamless surface with neighboring groups at their boundaries. We solve this by formulating an energy minimization problem with a group sparse regularizer, inspired by Amberg et al. (2007). Amberg et al. (2007) formulate a nonrigid registration problem as a least-squares problem which consists of three energy terms; distance, stiffness and landmark terms. We substitute the stiffness term to a smoothness term and introduce a group sparse regularizer to constrain the solution to be close to rigid motions obtained from graph matching. Similar to Amberg et al. (2007), the transformation is parameterized as one 3×4 affine transformation matrix $\mathbf{X}_i$ per group. Stacking the transpose of $\mathbf{X}_i$'s we get a $4K \times 3$ optimization variable matrix $\mathbf{X}$:

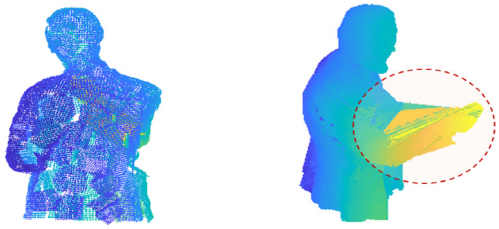
$$\mathbf{X} := [\mathbf{X}_1 \ \cdots \ \mathbf{X}_K]^T \tag{10}$$

Algorithm 1: Overall graph matching process

Input : source point cloud $\mathcal{P}^S$, target point cloud $\mathcal{P}^T$, number of groups K

Output: source centroids $\mathcal{C}^S$, target centroids $\mathcal{C}^T$, source index function $id^S(\cdot)$, target index function $id^T(\cdot)$, transformation tensor $\bar{\mathbf{X}}$

1 **Function** GraphMatching($\mathcal{P}^S$, $\mathcal{P}^T$, K)
2 $\quad\mathcal{C}^S$, $id^S \leftarrow$ cluster source point cloud $\mathcal{P}^S$ into K groups;
3 $\quad\mathcal{C}^T$, $id^T \leftarrow$ cluster target point cloud $\mathcal{P}^T$ into K groups;
4 $\quad$**for** $i \leftarrow 1$ ***to*** K **do**
5 $\quad\quad$**for** $j \leftarrow 1$ ***to*** K **do**
6 $\quad\quad\quad\mathcal{P}_i^S \leftarrow$ center subtracted i-th point group of $\mathcal{P}^S$;
7 $\quad\quad\quad\mathcal{P}_j^T \leftarrow$ center subtracted j-th point group of $\mathcal{P}^T$;
8 $\quad\quad\quad\mathbf{R}_{ij}$, $\mathbf{b}_{ij} \leftarrow$ RigidICP($\mathcal{P}_i^S$, $\mathcal{P}_j^T$);
9 $\quad$construct affinity matrix $\mathbf{M}$ according to (2);
10 $\quad$initialize $\mathbf{x}, \mathbf{y}, \lambda > 0, \rho > 1$;
11 $\quad$**while** *not converge* **do**
12 $\quad\quad\mathbf{x}^{k+1} \leftarrow \arg\max_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}^k, \lambda^k)$ s.t $\mathbf{x} \in \Pi_1$;
13 $\quad\quad\mathbf{y}^{k+1} \leftarrow \arg\min_{y} L(\mathbf{x}^{k+1}, \mathbf{y}, \lambda^k)$ s.t $\mathbf{y} \in \Pi_1$;
14 $\quad\quad\lambda^{k+1} \leftarrow \rho\lambda^k$;
15 $\quad\Phi \leftarrow$ reshape $\mathbf{x}^k$ into $K \times K$ matrix;
16 $\quad$**for** $i \leftarrow 1$ ***to*** K **do**
17 $\quad\quad j \leftarrow \arg\max_k \Phi(i,k)$;
18 $\quad\quad\bar{\mathbf{X}}_i \leftarrow [\mathbf{R}_{ij}, \mathbf{b}_{ij} + \mathbf{c}_i - \mathbf{R}_{ij}\mathbf{c}_i]^T$, where $\mathbf{c}_i \in \mathcal{C}^S$ is the i-th element of $\mathcal{C}^S$;



(a) (b)

Fig. 3. Transformed points according to the results of the graph matching module and their corresponding rigid motions. (a) The transformed surface look like a carapace of a tortoise, (b) Some transformed meshes look like stretched cheese because of incorrect matches.

3.2.1. Distance term

The distance term consists of sum of squared distances between source points and the corresponding target points.

$$E_d(\mathbf{X}) := \sum_{\mathbf{p}_l \in \mathcal{P}^S} dist^2(\mathbf{t}_l,\ \mathbf{X}_{id^S(\mathbf{p}_l)}\,\mathbf{p}_l) \tag{11}$$

where $\mathbf{t}_l$ is the target point of $\mathbf{p}_l$ and $id^S(\cdot)$ is a function that outputs the group index of $\mathbf{p}_l$. Usually, $\mathbf{t}_l$ is determined as the closest point of $\mathbf{v}_l$ in the target point cloud in many registration algorithms such as ICP. However, this approach can be easily fooled by nearby noisy points, so it is prone to increase errors. To prevent this, we set $\mathbf{t}_l$ to be the closest point of transformed $\mathbf{p}_l$ based on the rigid motion found in the graph matching module. To convert (11) into a matrix-multiplication form, we introduce an $n \times 4K$ matrix $\mathbf{D}$ whose elements are defined as follows:

$$\mathbf{D}(l,k) = \begin{cases} \mathbf{p}_l(k \bmod 4), & \text{if } 4(c(\mathbf{p}_l)-1) < k \le 4c(\mathbf{p}_l) \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where n is the number of points, K is the number of groups of the source point cloud, $\mathbf{D}(l,k)$ indicates the lth row, kth column element of $\mathbf{D}$, and $\mathbf{p}_l(k)$ indicates the kth element of $\mathbf{p}_l$. For example, let us assume that without loss of generality, $\mathbf{p}_1$, $\mathbf{p}_2$ are in group 1, and $\mathbf{p}_3$, $\mathbf{p}_4$ are in group 2. Then the matrix $\mathbf{D}$ look like this:

$$\mathbf{D} = \begin{bmatrix} \mathbf{p}_1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \mathbf{p}_2 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0\ 0\ 0\ 0 & \mathbf{p}_3 & & & 0 & \cdots & 0 \\ 0\ 0\ 0\ 0 & \mathbf{p}_4 & & & 0 & \cdots & 0 \\ \vdots & & \ddots & & & & \vdots \end{bmatrix} \tag{13}$$

Then, we can rewrite Eq. (11) as a matrix multiplication form

$$E_d(\mathbf{X}) := \|\mathbf{DX} - \mathbf{U}\|_F^2 \tag{14}$$

where $\mathbf{U} = [\mathbf{t}_1\ \cdots\ \mathbf{t}_n]^T$.

3.2.2. Smoothness term

The smoothness term is introduced to smooth the surfaces of the transformed source point cloud, especially surfaces on the boundaries of the transformed point groups. Let $\mathbf{c}_i$ be the center points of the ith point group in source and assume that group j is adjacent to group i. Similar to the node similarity function, the smoothness term penalizes the distance between two transformed locations of center point $\mathbf{c}_i$ according to affine transformations $\mathbf{X}_i$ and $\mathbf{X}_j$. In this manner, the smoothness term is defined as

$$E_s(\mathbf{X}) := \sum_{\mathbf{c}_i \in C^S} \sum_{\mathbf{c}_j \in \mathcal{N}(\mathbf{c}_i)} \left\|\mathbf{X}_i\mathbf{c}_i - \mathbf{X}_j\mathbf{c}_i\right\|^2 \tag{15}$$

Let $G^S = (\mathcal{V}^S, \mathcal{E}^S)$ be a naive-graph of the source point cloud constructed as described in Section 3.1.1. To rewrite Eq. (15) into a matrix-multiplication form, we introduce a $2|\mathcal{E}^S| \times 4K$ matrix $\mathbf{D}_S$ which is subdivided into $\{\mathbf{D}_S^{(i)}\}_{i=1}^K$, each of which is constructed from group i and its neighbors. If group j is a neighbor of group i, then a vector $\mathbf{d}$, whose element is defined as Eq. (16), form a row of $\mathbf{D}_S^{(i)}$ as follows:

$$\mathbf{d}(k) = \begin{cases} \mathbf{c}_i(k \bmod 4) & if\ 4(i-1) < k \le 4i, \\ -\mathbf{c}_i(k \bmod 4) & if\ 4(j-1) < k \le 4j, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

where $\mathbf{d}(k)$ and $\mathbf{c}_i(k)$ indicates the kth element of the vector $\mathbf{d}$ and $\mathbf{c}_i$ respectively. For example, if $i = 1$ and $j = 2$, then $\mathbf{d}$ becomes:

$$\mathbf{d} = \begin{bmatrix} \mathbf{c}_1^T & -\mathbf{c}_1^T & 0 & \cdots & 0 \end{bmatrix} \tag{17}$$

In this manner, we can rewrite Eq. (15) into a matrix-multiplication form as

$$E_s(\mathbf{X}) := \|\mathbf{D}_S\mathbf{X}\|_F^2 \tag{18}$$

3.2.3. Sparse regularizer

One of the purposes of the initializer module is to remove unfavorable effects induced by non-matched edges. We expect that the parameters of semantically correct matches remain similar to rigid transformation parameters obtained from the graph matching module while those of the incorrect matches change a lot. The rigid transforms obtained from the graph matching module can be interpreted as a solution corrupted by small Gaussian noises and sparse outliers. To get rid of these noises and outliers effectively, we introduce a group sparse regularizer. With an altered definition for the 2, 1-norm notation, $\|\cdot\|_{2,1}$, on $\mathbf{X}$ in Eq. (10) such that $\|\mathbf{X}\|_{2,1} := \sum_{i=1}^K \|\mathbf{X}_i\|_F$, we formulate the sparse regularizer $E_r(\mathbf{X})$ as

$$E_r(\mathbf{X}) := \|\mathbf{X} - \bar{\mathbf{X}}\|_{2,1} \tag{19}$$

where $\bar{\mathbf{X}}$ denotes a stack of rigid transformation parameters obtained from the graph matching module, i.e.,

$$\bar{\mathbf{X}} := [\bar{\mathbf{X}}_1\ \cdots\ \bar{\mathbf{X}}_K]^T \tag{20}$$

with $\bar{\mathbf{X}}_i = [\mathbf{R}_i,\ \mathbf{b}_i + \mathbf{c}_i - \mathbf{R}_i\mathbf{c}_i]$, in which $\mathbf{R}_i$, $\mathbf{b}_i$, and $\mathbf{c}_i$ denoting a 3×3 rotation matrix, a 3×1 translation matrix, and 3×1 centroid matrix of group i, respectively, obtained from the graph matching module.

3.2.4. Complete formulation

The complete formulation of the initial alignment module is given as a weighted sum of all energy functions:

$$E(\mathbf{X}) := E_d(\mathbf{X}) + \gamma E_s(\mathbf{X}) + \lambda E_r(\mathbf{X}) \tag{21}$$

Rewriting (21) into a matrix-multiplication form, we obtain

$$\begin{aligned} E(\mathbf{X}) &:= \|\mathbf{DX} - \mathbf{U}\|_F^2 + \gamma \left\|\mathbf{D}_S\mathbf{X}\right\|_F^2 + \lambda \left\|\mathbf{X} - \bar{\mathbf{X}}\right\|_{2,1} \\ &= \left\| \begin{bmatrix} \mathbf{D} \\ \gamma\mathbf{D}_S \end{bmatrix} \mathbf{X} - \begin{bmatrix} \mathbf{U} \\ \mathbf{0} \end{bmatrix} \right\|_F^2 + \lambda \left\|\mathbf{X} - \bar{\mathbf{X}}\right\|_{2,1} \\ &= \left\|\mathbf{AX} - \bar{\mathbf{B}}\right\|_F^2 + \lambda \left\|\mathbf{X} - \bar{\mathbf{X}}\right\|_{2,1} \end{aligned} \tag{22}$$

where $\mathbf{A} = [\mathbf{D}^T\ \gamma\mathbf{D}_S^T]^T$ and $\bar{\mathbf{B}} = [\mathbf{U}^T\ \ \mathbf{0}^T]^T$. Eq. (22) becomes an objective function of a group lasso problem with alternative variables $\mathbf{Y} = \mathbf{X} - \bar{\mathbf{X}}$ and $\mathbf{B} = \bar{\mathbf{B}} - \mathbf{A}\bar{\mathbf{X}}$, i.e.,

$$E(\mathbf{Y}) := \|\mathbf{AY} - \mathbf{B}\|_F^2 + \lambda \|\mathbf{Y}\|_{2,1} \tag{23}$$

The total energy function $E(\mathbf{Y})$ can be minimized using ADMM (Boyd et al., 2011) with the augmented Lagrangian of (23):

$$L_\rho(\mathbf{Y}, \mathbf{V}, \mathbf{Z}) = \|\mathbf{AY} - \mathbf{B}\|_F^2 + \lambda \|\mathbf{V}\|_{2,1} + (\rho/2)\|\mathbf{Y} - \mathbf{V} + \mathbf{Z}\|_F^2 \tag{24}$$

After solving the minimization problem of (24) to get an optimal $\mathbf{Y}^*$, the optimal transformation parameter $\mathbf{X}^*$ can be recovered by $\mathbf{X}^* = \mathbf{Y}^* + \bar{\mathbf{X}}$. After that, we can generate an initialization for the final step by transforming each point group of the source point cloud with the optimal solution $\mathbf{X}^*$. The overall initial alignment process is described in Algorithm 2, where $\mathcal{S}_\alpha(\cdot)$ denotes the soft thresholding operator (Tibshirani, 1996), i.e., $\mathcal{S}_\alpha(\mathbf{x}) = (1 - \alpha/\|\mathbf{x}\|)_+\mathbf{x}$, with an

Algorithm 2: Overall initial alignment process

Input : source point cloud $\mathcal{P}^S$, target point cloud $\mathcal{P}^T$, number of group K, source centroids C^S, target centroids C^T, source index function id^S, target index function id^T, transformation tensor $\bar{\mathbf{X}}$

Output: transformed source point cloud $\mathcal{P}^{TR-i}$, transformation matrix $\mathbf{X}^*$

Function `InitialAlignment`($\mathcal{P}^S$, $\mathcal{P}^T$, K, C^S, C^T, id^S, id^T, $\bar{\mathbf{X}}$)
- compute matrix $\mathbf{A}$ and $\mathbf{B}$ according to (22);
- initialize $\mathbf{Y}, \mathbf{V}, \mathbf{Z}$ with zeros;
- **while** *not converge* **do**
  - $\mathbf{Y}^{k+1} \leftarrow (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}(\mathbf{A}^T\mathbf{B} + \rho(\mathbf{Z}^k - \mathbf{V}^k))$;
  - **for** $i \leftarrow 1$ ***to*** K **do**
    - $\mathbf{V}_i^{k+1} \leftarrow \mathcal{S}_{\lambda/\rho}(\mathbf{Y}_i^{k+1} + \mathbf{Z}_i^k/\rho)$;
  - $\mathbf{Z}^{k+1} \leftarrow \mathbf{Z}^k + \rho(\mathbf{Y}^{k+1} - \mathbf{V}^{k+1})$;
- $\mathbf{X}^* \leftarrow \mathbf{Y}^k + \bar{\mathbf{X}}$;
- $\mathcal{P}^{TR-i} = \emptyset$;
- **for** $j \leftarrow 1$ ***to*** K **do**
  - $\mathcal{P}^{TR-i} \leftarrow \mathcal{P}^{TR-i} \cup \{i$-th point group of $\mathcal{P}^S$ multiplied by $\mathbf{X}_i\}$;

Algorithm 3: Overall process of nonrigid registration with graph-matching-based initialization

Input : source point cloud $\mathcal{P}^S$, target point cloud $\mathcal{P}^T$, number of group K

Output: transformed source point cloud $\mathcal{P}^{TR-r}$, transformation matrix $\mathbf{X}$

1 $C^S, C^T, id^S, id^T, \bar{\mathbf{X}} \leftarrow$ `GraphMatching`($\mathcal{P}^S$, $\mathcal{P}^T$, K);
2 $\mathcal{P}^{TR-i}, \mathbf{X}^* \leftarrow$ `InitialAlignment`($\mathcal{P}^S$, $\mathcal{P}^T$, K, C^S, C^T, id^S, id^T, $\bar{\mathbf{X}}$);
3 $\mathcal{P}^{TR-r}, \mathbf{X} \leftarrow$ `NonrigidICP`($\mathcal{P}^{TR-i}$, $\mathcal{P}^T$, $\mathbf{X}^*$);

element-wise positive indicator function $(\cdot)_+ = \mathbb{1}(\cdot > 0)$. The outputs of the initial alignment module are the transformed point cloud $\mathcal{P}^{TR-i}$ and the affine transformation matrix $\mathbf{X}^*$. $\mathcal{P}^{TR-i}$ will be passed to the final refinement module and further refined to complete the overall registration processes.

3.3. Refinement and the overall process

Applying group-wise transformation $\mathbf{X}^*$ obtained from the initial alignment module to the source point cloud results in $\mathcal{P}^{TR-i}$ which resembles the target point cloud and contains smooth surfaces. However, the transformed source point cloud still contains errors because it is transformed through group-wise motions. To reduce errors arising from the assumption of group-wise motions, we apply nonrigid ICP for the final refinement. Because the shape of the resultant transformed point cloud of the initial alignment module is already very similar to the target point cloud, a simple off-the-shelf nonrigid ICP algorithm (Amberg et al., 2007) successfully reduces the remaining errors. The overall process of the proposed nonrigid registration is described in Algorithm 3, where `NonrigidICP`($\mathcal{P}^S$, $\mathcal{P}^T$) denotes a nonrigid ICP function which outputs the transformed point cloud P^{TR-r} transformed from $\mathcal{P}^S$ to fit $\mathcal{P}^T$ and its transform parameters $\mathbf{X}$.

4. Experiments

In this section, we evaluate the proposed method on the Lincoln (Collet et al., 2015) and the TOSCA (Bronstein et al., 2008) datasets. Lincoln consists of multiple frames of human body scans. We use the upper part of the body for registration as its motion is large whereas the lower part hardly moves. TOSCA consists of scans of animals and humans in various poses. Originally, all the datasets are in the form of mesh structures, consisting of vertices and faces. We use the vertices of a mesh as a point cloud for the graph matching and the initial alignment modules, and use the face information in illustrating the transformed points as a mesh or in the final refinement module if required.

To evaluate performance of registration, we use the normalized *Chamfer distance* (*nChamfer*) as a set distance metric:

$$dist(\mathcal{X}, \mathcal{Y}) := \frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}}\min_{\mathbf{y}\in\mathcal{Y}}\|\mathbf{x}-\mathbf{y}\| + \frac{1}{|\mathcal{Y}|}\sum_{\mathbf{y}\in\mathcal{Y}}\min_{\mathbf{x}\in\mathcal{X}}\|\mathbf{x}-\mathbf{y}\| \tag{25}$$

where $|\mathcal{X}|$ denotes the cardinality of $\mathcal{X}$. As the normalized Chamfer distance is a symmetric function, it has a high value when there exists a region where transformed points are scarce and target points are dense. Hence, it is a desirable metric for measuring quality of registration, in the sense that it measures how well the source points are evenly transformed into target points. In addition, we use RMSE, which is equivalent to one-sided term of the normalized Chamfer distance. As some of our baseline algorithms are not scalable, we downsampled the points for fair comparison in Section 4.3. Because the normalized Chamfer distance increases as the density of points gets increased, we use RMSE for fair comparison. The definition of RMSE is:

$$RMSE(\mathcal{X}, \mathcal{Y}) := \frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}}\min_{\mathbf{y}\in\mathcal{Y}}\|\mathbf{x}-\mathbf{y}\| \tag{26}$$

where $\mathcal{X}$ denotes the source point set and $\mathcal{Y}$ the target point set. However, be aware that RMSE cannot capture missing regions in $\mathcal{X}$ where points are densely exist in $\mathcal{Y}$. All experiments are conducted using MATLAB® on Intel® Xeon® E5-2680v4@2.40 GHz processor.

4.1. Analysis on graph matching module

Given two frames of point clouds, source and target, we divide each point cloud into K groups. We set $K = 100$ for the Lincoln dataset, and $K = 50$ for the TOSCA dataset if not mentioned. In this subsection, we investigate the effects of many-to-one matching compared to one-to-one matching. For one-to-one matching, we use the reweighted random walk graph matching (RRWM) (Cho et al., 2010) which is one of the state-of-the-art graph matching algorithms.

Effects of many-to-one matching To investigate the effects of graph matching, we measure two metrics; ratio of non-matched edges and mean distance between two transformed centroids for neighboring groups in the source frame. The point groups are transformed based on the matching result and its corresponding rigid motion parameters. Experiments are done on all datasets varying the number of steps for graph construction. Lower values of both metrics imply that the matching results are consistent with those of neighboring groups. The results are summarized in Table 1. For both metrics, many-to-one matching scores lower values than one-to-one matching on all datasets. In other words, the proposed many-to-one matching generates more consistent matching among neighboring groups, which implies that the transformed surfaces are more smooth and less prone to wrinkles and tearings. Hence, we can say that many-to-one matching provides more favorable initial rigid motion parameters than one-to-one matching.

Grouped points of the Lincoln2 dataset and the corresponding matching results using many-to-one matching are illustrated in Fig. 4. The first row represents the source frame and the second row represents the target frame. The first column shows input meshes, the second column shows grouped points and the third column shows matching results. Numbering of the third column is based on the target point group index, i.e., the indices of the source groups indicate those of their matched target groups. Colors in the second column are set distinctively for each matched pair. We can see in the third row, first column, of Fig. 4 that multiple groups are simultaneously matched to group number 51 and group number 67. As a consequence, non-matching

Table 1
Effects of n-step graph on graph matching in terms of ratio of non-matched edges and mean distance between transformed neighbor groups. For one-to-one matching, RRWM (Cho et al., 2010) is used. For many-to-one matching, ours is used

		Non-matched edges (%)				Mean neighbor distance (mm)			
		1-step	**2-step**	**3-step**	**4-step**	**1-step**	**2-step**	**3-step**	**4-step**
One-to-one matching	**Lincoln1**	24.9	22.1	20.7	**19.7**	154.81	135.11	130.18	**127.61**
	Lincoln2	20.2	21.3	**19.4**	24.7	124.85	112.87	**106.96**	110.02
	wolf	**15.4**	17.3	19.2	19.2	27.38	**26.94**	27.33	27.33
	horse	26.6	26.6	**25.2**	26.6	**38.79**	**38.79**	40.28	41.45
	cat	**18.1**	18.8	18.8	20.1	**19.23**	20.76	29.03	22.87
	centaur	24.6	25.4	23.8	**23.1**	31.54	**29.12**	29.95	29.64
	david	**13.4**	18.1	25.2	27.6	17.03	17.06	17.39	16.80
	dog	**18.4**	22.0	21.3	21.3	18.46	20.29	17.80	18.58
	michael	**19.0**	21.4	19.0	23.0	17.22	20.56	**15.95**	16.47
Many-to-one matching	**Lincoln1**	**13.0**	13.3	15.1	14.7	118.85	**117.01**	117.37	117.16
	Lincoln2	18.6	**14.4**	19.4	20.9	104.57	**88.61**	89.92	90.26
	wolf	13.5	**11.5**	**11.5**	17.3	25.88	**25.24**	25.67	26.62
	horse	11.9	11.9	**10.5**	11.2	27.22	27.22	**27.06**	27.22
	cat	11.8	**7.6**	13.2	13.2	15.92	14.32	**14.27**	14.45
	centaur	22.3	**16.9**	19.2	23.8	27.07	21.25	**21.03**	22.34
	david	19.7	**11.0**	14.2	20.5	15.41	11.07	**10.60**	11.48
	dog	17.0	**6.4**	12.8	9.9	13.11	**12.38**	13.05	12.74
	michael	**14.3**	17.5	15.1	16.7	12.38	13.16	**12.07**	13.16

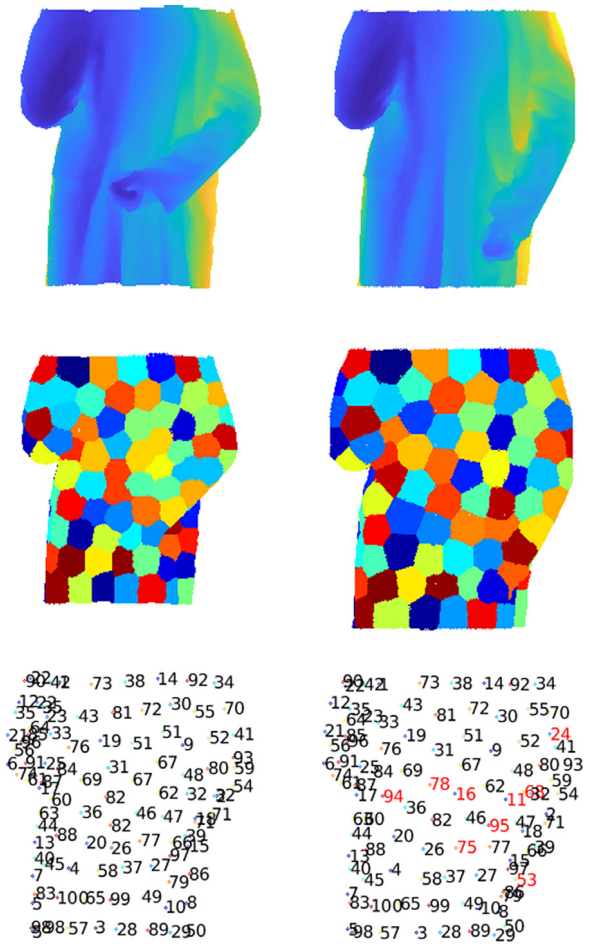
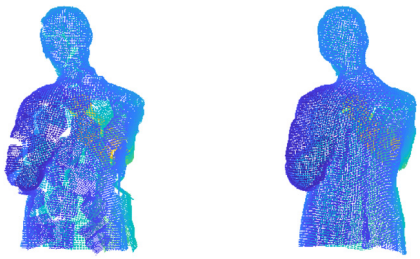


Fig. 4. Group correspondences found by graph matching module on Lincoln dataset. The first column represents the source frame and the second column represents the target frame. The first row shows input meshes, the second row shows grouped points and the third row shows matching results. Numbering is based on the target point group index, i.e., the indices of the source groups indicate those of the matched target groups. Indices written in red indicate non-matching groups. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

groups occur in the target frame, which are written in red in the third column of Fig. 4. The non-matching groups are mostly located in unveiled regions which is occluded by the right arm in the source frame. Some non-matching groups such as 16, 78, and 94 can be interpreted as newly appeared regions by stretched clothes. The appearance patterns of the non-matching groups explain the actual physical motion quite well. Consequently, matches are generated to have similar semantics without any awkward motions like one illustrated in Fig. 3(b).

Analysis on n-step graph As discussed in Section 3.1.1, the number of neighboring steps, i.e., n of an n-step graph, for edge construction affects the quality of graph matching; ratio of non-matched edges and mean distance between neighboring groups in target graph. We also investigate the effects of an n-step graph on various datasets and various types of matching schemes as described in Table 1. In terms of the non-matched edges, the value tends to decrease as the number of step increases for certain steps. After reaching the minimum point, the value tends to increase as the number of step increase. The same pattern is found on mean neighbor distance between two transformed neighbors. As discussed in Section 3.1.1, the ratio of non-matched edges represents tearing of surfaces, and the mean distance between two transformed neighbors indicates indirectly the degree of tearing. Hence, having lower values for both metrics is preferable. In Table 1, although the best configurations defer by datasets, many-to-one matching yields the best performance at 2 or 3-step graph in terms of mean distance between transformed neighboring centroids. The same is true for the ratio of non-matched edges except for the Lincoln1 and michael data and 2-step graphs mostly have the best performance. As there are chances to redeem the drift of neighboring centroids in the following modules, and the performance of many-to-one matching always exceeds that of one-to-one matching in terms of both metrics, we use many-to-one matching and 2-step graphs for the rest of the experiments.

Effects of the number of groups The graph matching module assumes that the nonrigid motion can be approximated by a set of rigid motion of grouped parts. Hence, an approximation error occurs and the error increases as the number of groups gets smaller. Although the approximation error is reduced by the initial alignment module and the refinement module, the number of groups still affects the registration performance. To achieve a high quality registration result, a large number of groups would be helpful. However, increasing the number of groups also increases computation time. In terms of computation time, the graph matching module is a critical path of the entire registration processes as we have to compute all rigid motion parameters for all point group pairs for constructing a similarity matrix. The computation time of the graph matching module is quadratically proportional to



(a) (b)

Fig. 5. (a) Transformed points after the graph matching module and (b) those after the initial alignment module.

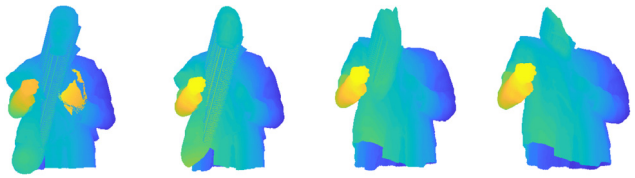
Table 2
Effects of the number of groups on registration performance and computation time. K denotes the number of groups.

		$K = 25$	$K = 50$	$K = 100$
Lincoln1	*nChamfer* (mm)	4.46	3.60	1.23
	RMSE (mm)	0.13	0.2	0.03
	time (s)	690.4	705.4	1257.3
Lincoln2	*nChamfer* (mm)	3.09	3.70	2.59
	RMSE (mm)	0.23	0.16	0.08
	time (s)	309.5	599.3	1223.1

the number of groups. To see the trade-off between the registration performance and computation time, we measure the computation time for graph matching module by varying the number of point group K by 25, 50, 100 for the Lincoln dataset. Here, to further reduce the computation time, we exploited a parallel loop iteration provided by MATLAB®. When $K = 100$, the registration performance achieves the best in terms of both *nChamfer* and RMSE. It is hard to tell which one is better between $K = 25$ and $K = 50$, as *nChamfer* and RMSE show different trends in Table 2. However, when $K = 25$, the matching result yields some erroneous match on Lincoln1 data as illustrated in Fig. 6 and deteriorates registration performance, which is captured by *nChamfer*. We can see that the computation time clearly increases as the number of groups increases.

4.2. Analysis on initial alignment module

The initial alignment module smoothes the surface of transformed points in which the points are transformed according to rigid motions transforming their belonging source groups to the matched target groups. As discussed in Section 3.2, the transformed points according to their corresponding rigid motions result in carapace-like surfaces as illustrated in Fig. 3(a). The initial alignment module refines rigid transform parameters to affine transforms by which the transformed points form more smoothly connected surfaces than before. To see the effects of the initial alignment module, we evaluate the average distance between two transformed centroids which are neighbors in the source frame. We empirically found that $\gamma = 5$ and $\lambda = 1e-2$ in (21) were suitable in the experiments. The results are summarized in Table 3. We set $K = 100$ for the Lincoln dataset and $K = 50$ for the TOSCA dataset. The distance between the two transformed neighboring centroids decreases after passing through the initial alignment module in all cases. This means that the initial alignment module successfully smooth the cracked surfaces from the graph matching module. The qualitative effects of the initial alignment module are illustrated in Fig. 5. Fig. 5(a) illustrates transformed surfaces after applying the graph matching module, whereas Fig. 5(b) illustrates those after the initial alignment module. We can confirm that surface cracks are considerably reduced after passing through the initial alignment module.



(a) input (b) $\gamma = 10$ (c) $\gamma = 50$ (d) $\gamma = 100$

Fig. 6. Examples of failure cases of the initial alignment module on Lincoln1 with 25 point groups: **(a)** input and results of the initial alignment module with **(b)** $\gamma = 10$, **(c)** $\gamma = 50$, and **(d)** $\gamma = 100$.

Table 3
Effects of initial alignment module. "GM" indicates the results of the graph matching module whereas "IA" indicates the results of the initial alignment module.

Dataset	GM (mm)	IA (mm)
Lincoln1	117.01	**110.83**
Lincoln2	88.61	**83.02**
wolf	25.24	**25.10**
horse	27.22	**27.12**
cat	14.32	**14.26**
centaur	21.03	**20.72**
david	10.60	**10.49**
dog	13.05	**12.95**
michael	12.07	**11.96**

Failure cases The initial alignment cannot correct all erroneous matches, especially when a matched target group of a erroneous group is far away from the matched target groups of neighbors. This phenomenon occurs when the number of group is too small that the group-wise rigid motion assumption cannot approximate well the whole nonrigid motion. Fig. 6 shows failure cases of the initial alignment module on the Lincoln1 data when the number of group K is 25. Fig. 6(a) shows an input containing an erroneous match that the back of the head of Lincoln is matched to clothes. Fig. 6(a)–6(d) shows some results of the initial alignment module with increasing the smoothness coefficient, γ. The erroneous match remains until $\gamma = 50$. When $\gamma = 100$, although the erroneous match are removed, the whole shape is destroyed and disappeared due to the large smoothing effect (the right shoulder is over lifted and face is disappeared).

4.3. Analysis on entire nonrigid registration process

The whole registration process is completed by the final refinement module, applying nonrigid registration on the output points of the initial alignment module. Any off-the-shelf nonrigid registration algorithm could be used, and in this paper, we choose the optimal-step nonrigid ICP (Optimal ICP) (Amberg et al., 2007) for the final step. For comparison, five nonrigid registration algorithms, i.e., Optimal ICP, coherent point drift (CPD) (Myronenko and Song, 2010), RPM-L2E (Ma et al., 2015), PR-GLS (Ma et al., 2016), and MR-RPM (Ma et al., 2018), are used (without any separate initialization step as in the proposed algorithm) for baselines. The compared algorithms are selected from the very basics to the recent state-of-the-arts: a nearest-neighbor based method (Optimal ICP), a cornerstone of the coherent drift method (CPD), a method utilizing a robust estimator (RPM-L2E), and methods with various regularizations; Tikhonov regularization on RKHS (PR-GLS) and manifold regularization (MR-RPM).

The registration results are summarized in Table 4. Experiments are conducted with $K = 100$ for the Lincoln dataset and $K = 50$ for the TOSCA dataset. The number of points for each frame of the Lincoln dataset is larger than 120,000. As the algorithms except ours and Optimal ICP are not scalable for such a large dataset, we subsample points when running the CPD, RPM-L2E, PR-GLS, and MR-RPM.

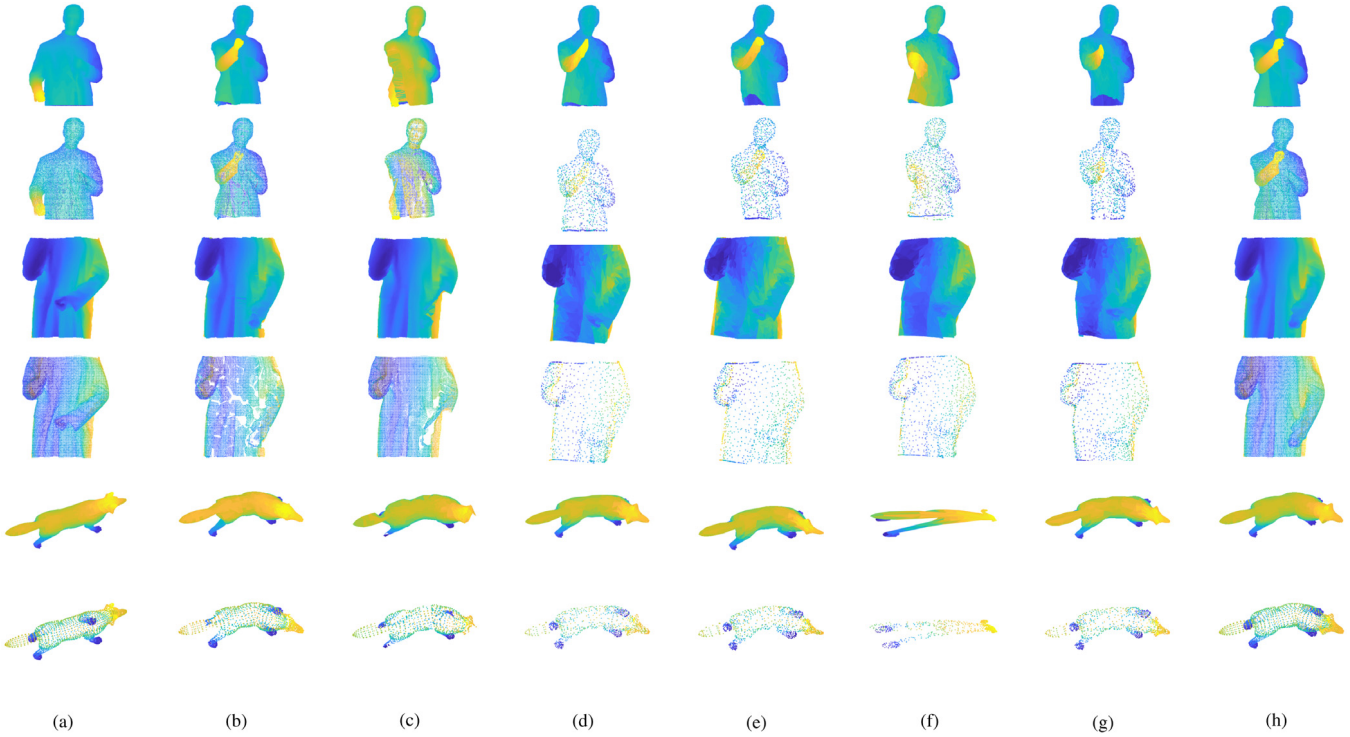


Fig. 7. Comparison of nonrigid registration algorithms. Point cloud and mesh figures of **(1, 2-row)** Lincoln1, **(3, 4-row)** Lincoln2, **(5, 6-row)** wolf dataset. Column-wise: point cloud and mesh figures of (a) source, (b) ours, (c) Optimal ICP, (d) CPD-S, (e) RPM-L2E-FPFH-S, (f) PR-GLS-S, (g) MR-RPM-FPFH-S and (h) target. "-S" indicates that the registration is performed on sub-sampled points due to un-scalability of the algorithms.

Table 4
Performances of registration algorithms. *nChamfer* indicates the normalized Chamfer distance. "-S" indicates that registration is performed on sub-sampled points. To compute *nChamfer*, both source and target points are sub-sampled. To compute RMSE, only source points are sub-sampled

Algorithms	*nChamfer* (mm)									RMSE (mm)								
	Lincoln1	Lincoln2	wolf	horse	cat	centaur	david	dog	michael	Lincoln1	Lincoln2	wolf	horse	cat	centaur	david	dog	michael
Src-S	95.75	31.63	17.21	39.76	17.37	43.74	49.74	20.23	34.26	34.28	8.94	8.31	17.87	7.49	18.83	23.80	8.71	13.61
Opt ICP-S (Amberg et al., 2007)	45.88	13.63	3.30	10.29	5.62	26.51	10.54	9.01	9.73	2.12	1.22	0.35	0.19	0.36	0.31	10.54	0.33	0.24
CPD-S (Myronenko and Song, 2010)	30.09	27.25	**1.69**	5.26	5.48	**10.02**	7.94	**4.25**	7.68	6.68	7.77	0.67	1.59	1.85	3.96	3.40	1.31	3.12
RPM-L2E-FPFH-S (Ma et al., 2015)	41.94	38.36	3.02	5.71	12.19	37.74	20.95	11.41	15.63	11.68	13.24	1.14	1.78	2.75	5.41	6.79	2.01	3.00
PR-GLS-S (Ma et al., 2016)	34.17	33.53	20.76	8.19	8.09	24.13	13.14	12.21	7.9	8.40	10.52	3.72	1.70	1.17	3.39	1.84	2.72	2.95
MR-RPM-FPFH-S (Ma et al., 2018)	46.41	39.27	2.96	5.83	13.02	15.42	19.76	10.23	27.59	10.97	12.08	1.13	1.81	4.36	4.20	4.55	2.69	7.24
Ours-S	**11.25**	**13.28**	2.90	**2.04**	**4.62**	18.25	**5.07**	4.76	**4.65**	**0.35**	**0.97**	**0.26**	**0.11**	**0.35**	**0.17**	**0.19**	**0.23**	**0.16**
Src	81.47	20.49	20.41	42.30	17.41	40.34	52.25	19.97	34.35	32.99	8.69	10.13	19.40	7.94	19.21	25.59	9.18	20.94
Opt ICP (Amberg et al., 2007)	34.96	4.73	3.55	13.66	6.85	22.43	5.62	8.26	10.26	0.22	0.13	0.18	0.07	**0.08**	0.08	**0.03**	0.07	**0.04**
Ours	**1.23**	**2.59**	**1.53**	**1.60**	**4.23**	**6.24**	**2.05**	**1.44**	**5.04**	**0.03**	**0.08**	**0.12**	**0.03**	**0.08**	**0.05**	**0.03**	**0.04**	0.08

Shape context (SC) is used as point features in original RPM-L2E and MR-RPM, which is not applicable in 3D point clouds. Hence, we substitute SC to FPFH for 3D data. Ten neighboring points are used for computing FPFH of the sub-sampled dataset using the point cloud library (PCL) (Rusu and Cousins, 2011). We sub-sampled 1% points for the Lincoln dataset, and around 1,000 points for the TOSCA dataset. To compute the normalized Chamfer distance, both the source and target points are sub-sampled, whereas only the source points are sub-sampled to compute RMSE. Table 4 shows that the proposed algorithm surpasses the other baselines in terms of RMSE on both original and sub-sampled data. In terms of normalized Chamfer distance, the proposed algorithm surpasses the other baselines on all the original data and 6 out of 9 sub-sampled data. For the other 3 sub-sampled data on which CPD yields the best performance, the proposed algorithm achieves the second best performance. One advantage of the proposed algorithm is that it works on large-scale data to which the other baselines cannot be applied. Optimal ICP achieves the second best performance in terms of RMSE, except the michael and sub-sampled david datasets. Especially, in the sense of RMSE, the registration performance of Optimal ICP looks similar to that of our algorithm. However, the actual transformed points of Optimal ICP looks quite different from those of ours because RMSE cannot point out uncovered regions on the target point cloud by the transformed points.

The final transformed points along with source, target, and each algorithm's output points are depicted in Fig. 7. As illustrated in the first and the third row of Fig. 7(c), the points of the right arm collapse to the body of the target point cloud, when transformed by the Optimal ICP, which is an undesirable registration result. The registration results of RPM-L2E, PR-GLS, and MR-RPM on the Lincoln dataset shows that they can simultaneously reduce the fitting errors while preserving the global structures. However, PR-GLS loses some degrees of joint angle of right elbow for the Lincoln1 and it cannot successfully transform source points to fit target points for the wolf data. RPM-L2E and MR-RPM nicely preserve local and global structures, however, they lose some details like sleeves or hands. On the other hand, our algorithm

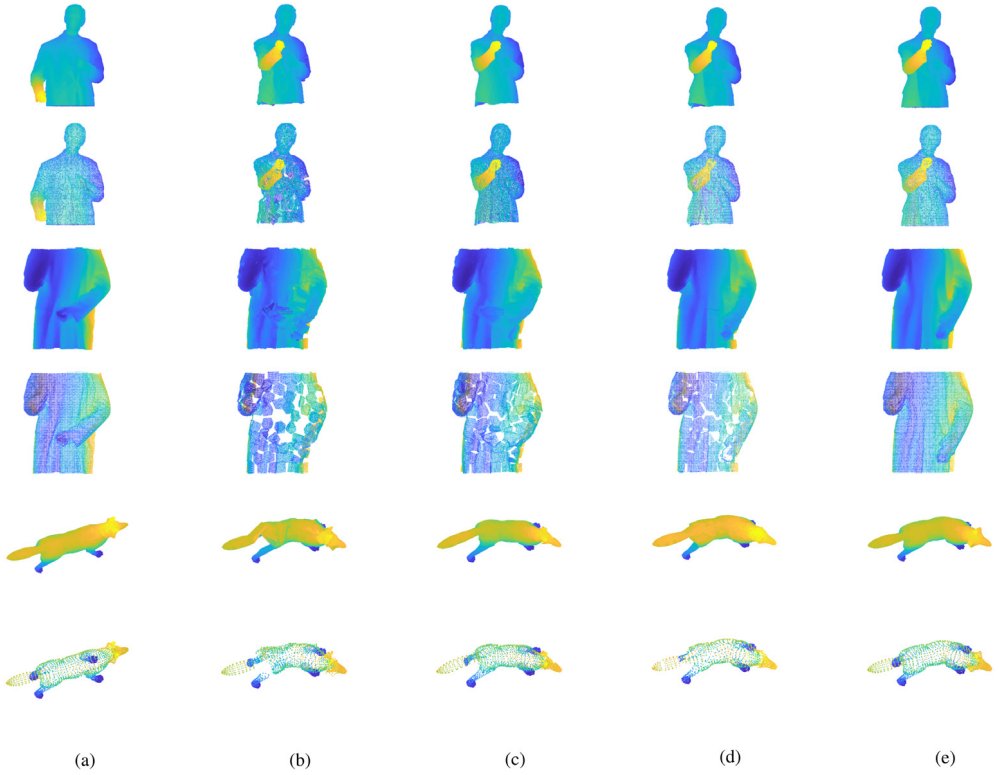


Fig. 8. Illustration of output point clouds and meshes of each module: (a) source, (b) graph matching, (c) initial alignment, (d) refinement, and (e) target. Each row represents the same dataset as in Fig. 7.

Table 5
Registration performance of various combinations of modules. *nChamfer* indicates the normalized Chamfer distance. The alphabets at the end of algorithm names indicate which modules are used for registration: G represents the graph matching module, I the initial alignment module, and R the refinement module.

Algorithms	*nChamfer* (mm)									RMSE (mm)								
	Lincoln1	Lincoln2	wolf	horse	cat	centaur	david	dog	michael	Lincoln1	Lincoln2	wolf	horse	cat	centaur	david	dog	michael
None	81.47	20.49	20.41	42.3	17.41	40.34	52.25	19.97	34.35	32.99	8.69	10.13	19.40	7.94	19.21	25.59	9.18	20.94
GM-NRR-G	8.66	8.41	2.96	3.06	5.18	8.88	3.00	2.20	5.85	3.82	3.27	1.08	0.95	0.77	1.69	0.65	0.56	0.85
GM-NRR-R	34.96	4.73	3.55	13.66	6.85	22.43	5.62	8.26	10.26	0.22	0.13	0.18	0.07	**0.08**	0.08	**0.03**	0.07	**0.04**
GM-NRR-GI	14.71	10.12	3.05	3.10	5.19	9.29	3.05	2.23	5.90	7.20	4.29	1.08	0.99	0.77	2.40	0.65	0.58	0.84
GM-NRR-GR	2.20	3.17	2.35	1.63	4.24	6.96	**2.11**	**1.44**	5.07	0.03	0.10	0.63	**0.03**	**0.08**	**0.04**	**0.03**	**0.04**	0.08
GM-NRR-GIR	**1.23**	**2.59**	**1.67**	**1.60**	**4.23**	**6.24**	2.14	**1.44**	**5.04**	**0.03**	**0.08**	**0.11**	**0.03**	**0.08**	**0.04**	**0.03**	**0.04**	0.08

successfully transforms source points to fit target points on all dataset as depicted in Fig. 7(b).

Ablation study To see the effects of each module on the performance of registration, we conduct ablation studies. We evaluate registration performance on five different combinations of modules which is summarized in Table 5. The alphabets at the end of algorithm names in Table 5 indicate which modules are used for registration: G represents the graph matching module, I the initial alignment module, and R the refinement module. We can see that both the metrics are greatly reduced after passing through the graph matching module. This is because the graph matching module successfully finds semantically meaningful matches between the source and the target. The refinement module, as its name says, further reduces the difference between the transformed points and the target points by refining the transformation parameters. The noticeable part is the initial alignment module. When the initial alignment module is applied to the outputs of the graph matching module, the two metric values are rather increased. At first glance, it seems that the initial alignment module disrupts the registration process. However, combined with the following refinement module, the error metrics are more reduced than without the initial alignment module. This is because the initial alignment module has an effect of gluing the boundaries of nearby isolated point groups. In that process, transformed points may loose some details as the module yet assumes group-wise motions. However, as the glued point groups can fit most part of the target points evenly, they can recover detailed surface patterns after the final refinement module. Otherwise, only partial surfaces of the target points can be fitted by the isolated point groups. The intermediate results of the proposed algorithm after each module are illustrated in Fig. 8.

5. Conclusions

In this paper, we proposed a novel nonrigid registration algorithm which registers two different point clouds capturing different poses of a single deformable object, by finding corresponding parts between the point clouds using graph matching. In particular, our algorithm has advantages when registering two frames with large displacements. We validated our algorithm extensively on the Lincoln and TOSCA datasets and showed that our algorithm is indeed successful in registering point clouds with large displacement, without any collapsed body parts that other algorithms exhibit. The proposed algorithm consists of three modules and among them, the graph matching module plays an important role for overall registration performance as shown in the experiments. One of the critical factors for the success of the graph matching module

is how to divide the points into groups, which determines the shapes of the resultant point groups. One possible way to improve registration performance is to develop an iterative grouping method to make the matched groups have more and more similar shapes, which is left for future work.

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.

CRediT authorship contribution statement

Seunggyu Chang: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft. **Chanho Ahn:** Methodology, Software, Formal analysis. **Minsik Lee:** Conceptualization, Methodology, Writing - review & editing, Funding acquisition. **Songhwai Oh:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition.

Acknowledgments

This work was supported by 'The Cross-Ministry Giga KOREA Project' grant funded by the Korea government (MSIT), Republic of Korea (No. GK20P0300, Real-time 4D reconstruction of dynamic objects for ultra-realistic service).

References

Allen, B., Curless, B., Popović, Z., 2002. Articulated body deformation from range scan data. ACM Trans. Graph. 21 (3), 612–619.

Amberg, B., Romdhani, S., Vetter, T., 2007. Optimal step nonrigid ICP algorithms for surface registration. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR, IEEE, pp. 1–8.

Anguelov, D., Srinivasan, P., Pang, H.-C., Koller, D., Thrun, S., Davis, J., 2005. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In: Proc. of Neural Information Processing Systems. NIPS, pp. 33–40.

Belongie, S., Malik, J., Puzicha, J., 2002. Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. (4), 509–522.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-D shapes. In: Sensor Fusion IV: Control Paradigms and Data Structures, Vol. 1611. International Society for Optics and Photonics, pp. 586–606.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. 3 (1), 1–122.

Bronstein, A.M., Bronstein, M.M., Kimmel, R., 2008. Numerical Geometry of Non-Rigid Shapes. Springer Science & Business Media.

Cho, M., Lee, J., Lee, K.M., 2010. Reweighted random walks for graph matching. In: Proc. of the European Conference on Computer Vision. ECCV, Springer, pp. 492–505.

Chui, H., Rangarajan, A., 2003. A new point matching algorithm for non-rigid registration. Comput. Vis. Image Underst. 89 (2–3), 114–141.

Collet, A., Chuang, M., Sweeney, P., Gillett, D., Evseev, D., Calabrese, D., Hoppe, H., Kirk, A., Sullivan, S., 2015. High-quality streamable free-viewpoint video. ACM Trans. Graph. 34 (4), 69.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Garey, M.R., 1979. A guide to the theory of NP-completeness. In: Computer and Intractability. WH Freeman & Co.

Huang, Q.-X., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graph. 25 (3), 569–578.

Huber, P.J., 1981. Robust Statistics. Wiley.

Jian, B., Vemuri, B.C., 2010. Robust point set registration using gaussian mixture models. IEEE Trans. Pattern Anal. Mach. Intell. 33 (8), 1633–1645.

Katayama, K., Narihisa, H., 2001. Performance of simulated annealing-based heuristic for the unconstrained binary quadratic programming problem. European J. Oper. Res. 134 (1), 103–119.

Liao, M., Zhang, Q., Wang, H., Yang, R., Gong, M., 2009. Modeling deformable objects from a single depth camera. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR, IEEE, pp. 167–174.

Loiola, E.M., de Abreu, N.M.M., Boaventura-Netto, P.O., Hahn, P., Querido, T., 2007. A survey for the quadratic assignment problem. European J. Oper. Res. 176 (2), 657–690.

Ma, J., Qiu, W., Zhao, J., Ma, Y., Yuille, A.L., Tu, Z., 2015. Robust $L_{2}e$ estimation of transformation for non-rigid registration. IEEE Trans. Signal Process. 63 (5), 1115–1129.

Ma, J., Wu, J., Zhao, J., Jiang, J., Zhou, H., Sheng, Q.Z., 2018. Nonrigid point set registration with robust transformation learning under manifold regularization. IEEE Trans. Neural Netw. Learn. Syst..

Ma, J., Zhao, J., Tian, J., Tu, Z., Yuille, A.L., 2013. Robust estimation of nonrigid transformation for point set registration. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 2147–2154.

Ma, J., Zhao, J., Tian, J., Yuille, A.L., Tu, Z., 2014. Robust point matching via vector field consensus. IEEE Trans. Image Process. 23 (4), 1706–1721.

Ma, J., Zhao, J., Yuille, A.L., 2016. Non-rigid point set registration by preserving global and local structures. IEEE Trans. Image Process. 25 (1), 53–64.

Merz, P., Freisleben, B., 2002. Greedy and local search heuristics for unconstrained binary quadratic programming. J. Heuristics 8 (2), 197–213.

Myronenko, A., Song, X., 2010. Point set registration: Coherent point drift. IEEE Trans. Pattern Anal. Mach. Intell. 32 (12), 2262–2275.

Raguram, R., Frahm, J.-M., Pollefeys, M., 2008. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In: Proc. of the European Conference on Computer Vision. ECCV, Springer, pp. 500–513.

Rusu, R.B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (FPFH) for 3D registration. In: Proc. of the IEEE International Conference on Robotics and Automation. ICRA, Citeseer, pp. 3212–3217.

Rusu, R.B., Blodow, N., Marton, Z.C., Beetz, M., 2008a. Aligning point cloud views using persistent feature histograms. In: Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, IEEE, pp. 3384–3391.

Rusu, R.B., Cousins, S., 2011. Point cloud library (pcl). In: Proc. of the IEEE International Conference on Robotics and Automation, ICRA, pp. 1–4.

Rusu, R.B., Marton, Z.C., Blodow, N., Beetz, M., 2008b. Learning informative point classes for the acquisition of object model maps. In: Proc. of the IEEE International Conference on Control, Automation, Robotics and Vision. ICARCV, IEEE, pp. 643–650.

Tevs, A., Bokeloh, M., Wand, M., Schilling, A., Seidel, H., 2009. Isometric registration of ambiguous and partial data. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 1185–1192.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B Stat. Methodol. 267–288.

Tsin, Y., Kanade, T., 2004. A correlation-based approach to robust point set registration. In: Proc. of the European Conference on Computer Vision. ECCV, Springer, pp. 558–569.

Wang, G., Wang, Z., Chen, Y., Zhao, W., 2015. A robust non-rigid point set registration method based on asymmetric gaussian representation. Comput. Vis. Image Underst. 141, 67–80.

Wang, G., Wang, Z., Chen, Y., Zhou, Q., Zhao, W., 2016. Context-aware Gaussian fields for non-rigid point set registration. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 5811–5819.

Wang, G., Wang, Z., Zhao, W., Zhou, Q., 2014. Robust point matching using mixture of asymmetric gaussians for nonrigid transformation. In: Proc. of the Asian Conference on Computer Vision. ACCV, Springer, pp. 433–444.

Wolsey, L.A., 1998. Integer Programming. Wiley.

Yuille, A.L., Grzywacz, N.M., 1989. A mathematical analysis of the motion coherence theory. Int. J. Comput. Vis. 3 (2), 155–175.