Break and Splice: A Statistical Method for Non-Rigid Point Cloud Registration

Qinghong Gao,[1] Yan Zhao,[1,3] Long Xi,[1] Wen Tang[1] and Tao Ruan Wan[2]

[1]Creativity Technology, Bournemouth University, Poole, UK
{qgao, zhaoy, lxi, wtang}@bournemouth.ac.uk
[2]Design & Technology, Department of Media, University of Bradford, Bradford, UK
T.Wan@bradford.ac.uk
[3]Department of Information Science and Technology, Northwest University, Xi'an, China

Abstract
3D object matching and registration on point clouds are widely used in computer vision. However, most existing point cloud registration methods have limitations in handling non-rigid point sets or topology changes (e.g. connections and separations). As a result, critical characteristics such as large inter-frame motions of the point clouds may not be accurately captured. This paper proposes a statistical algorithm for non-rigid point sets registration, addressing the challenge of handling topology changes without the need to estimate correspondence. The algorithm uses a novel Break and Splice *framework to treat the non-rigid registration challenges as a reproduction process and a Dirichlet Process Gaussian Mixture Model (DPGMM) to cluster a pair of point sets. Labels are assigned to the source point set with an iterative classification procedure, and the source is registered to the target with the same labels using the Bayesian Coherent Point Drift (BCPD) method. The results demonstrate that the proposed approach achieves lower registration errors and efficiently registers point sets undergoing topology changes and large inter-frame motions. The proposed approach is evaluated on several data sets using various qualitative and quantitative metrics. The results demonstrate that the* Break and Splice *framework outperforms state-of-the-art methods, achieving an average error reduction of about 60% and a registration time reduction of about 57.8%.*

Keywords: non-rigid registration, point cloud, topology changes, Gaussian Mixture Model, computer vision

CCS Concepts: • Computing methodologies → Point-based models; Computer vision tasks; Scene understanding

1. Introduction

Point cloud registration is a crucial step in the 3D acquisition and has many applications in computer vision, including 3D reconstruction, pose estimation, augmented reality, object matching and recognition [TCL*12, CTJ*18, WSMG*16, SBB17]. Accurate registration of multiple point clouds obtained from different views or time instants is necessary for building a complete and consistent 3D model of the scene or object of interest. In addition, point cloud registration enables us to estimate the relative pose and motion of objects, recognize and match objects in different scenes and create virtual and augmented reality experiences [ZSG*18, DWB06].

While many registration methods work well on rigid objects [AHB87, CSK05], they often perform poorly on dynamic scenes or deformed objects. This is because these objects have non-rigid deformations and motions that cannot be modelled by rigid transformations. In addition, non-rigid objects may undergo topology changes such as connections and separations, which pose additional challenges for registration methods that rely on correspondences between the source point sets and the target point sets point [HAWG08, MQZ*15]. Therefore, developing registration methods that can handle non-rigid objects and dynamic scenes is an active research area in computer vision.

Topological changes are common in dynamic scenes. Many previous works on non-rigid registration have failed to effectively address the connection and separation issues, and large inter-frame motions, which can lead to misregistration and inaccurate reconstructions. In addition, large inter-frame motions can cause significant deformations and changes in topology. Chui *et al.* [CR03] and
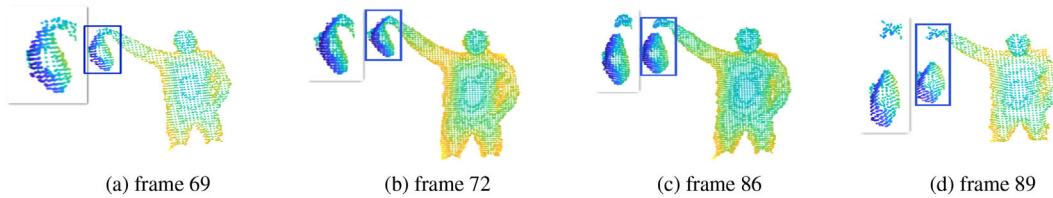
(a) frame 69   (b) frame 72   (c) frame 86   (d) frame 89

Figure 1: *Non-rigid registration challenges. The (a), (b), (c) and (d) show that the hat is separated from the hand from frame 69 to 89 in the public data set, including two object separations [SBCI17].*

Yang *et al.* [YOF15] proposed non-rigid registration methods based on point correspondence to estimate affine transformations between the source and the target point sets. However, these methods may not be robust to changes in the topology of the point sets, such as connections and separations. The accuracy of these methods is highly dependent on the accuracy of the correspondence estimation, which can be challenging in the presence of large deformations or changes in topology.

To address these issues, recent approaches have focused on developing statistical methods that do not rely on explicit point correspondence or feature extraction, but instead model the probabilistic relationship between the point sets. Myronenko *et al.* [MSCP*07] and Hirose [Hir21] have utilized statistical methods, such as the motion coherence theory, to estimate maximum likelihood solutions for non-rigid registration. However, some of these methods, such as Coherent Point Drift (CPD), can suffer from local minima and slow convergence. More recently, Zampogiannis *et al.* [ZFA21] have explored a bidirectional estimation method, where pairs of point clouds are aligned from the source to the target and from the target to the source. However, this method still struggles to handle large inter-frame motions, where the source and target point clouds may undergo significant deformation between frames.

The challenge of handling objects' *separation and connection* can be best illustrated through an example shown in Figure 1. In this example, from frame 69 to frame 89 where the hat is separated from the hand, large inter-frame motions are manifested. The difficulty of non-rigid registration stems from the effectiveness of the method in dealing with a single point set when it is rapidly separated into two point sets (the hat and the hand). The connection between the two point sets is regarded as a reverse process of separation.

In this paper, we propose a novel statistical approach that can handle changes in topology and large inter-frame motions. The proposed framework utilizes a statistical approach based on the Dirichlet Process Gaussian Mixture Model (DPGMM) to handle non-rigid point sets without the need for explicit correspondence estimation, leading to improved registration performance compared to previous methods. Our method in this paper regards non-rigid registration as a reproduction process with a four-step registration scheme, as shown in Figure 2, which generates a model as close as possible to the target model. Another contribution of this paper is that we design a *Cluster and Refine* scheme to handle the distribution irregularities of point sets, making the topology of source points as same as that of target points, which results in a great improvement of the accuracy and efficiency of the proposed statistic-based methods. The proposed method is evaluated on five datasets using a variety of qualitative and quantitative metrics. It is important to note that the experimental datasets only contain single object separations and connections against a simple background.

2. Related Work

Non-rigid point cloud registration methods can be categorized into general transformation models and Gaussian mixture models.

Chui *et al.* [CR03] used a thin plate spline (TPS) to define a general transformation model that consists of an affine transformation and a TPS smoothness kernel. Yang *et al.* [YOF15] used a global and local mixture distance to estimate the correspondence between the source point set and the target point set and update the rigid and non-rigid transformations and minimized the mixture distance using a TPS. Huang *et al.* [HAWG08] utilized a rigid local transformation for each point to obtain a global non-rigid registration. Meanwhile, the local affine transformation [ACP03, ARV07] is frequently applied in non-rigid registration because the surface representation allows surface details to be captured precisely due to its more freedom. However, general transformation methods depend on correspondence estimation based on the features of the source point set and target point set, whose result would directly affect the registration accuracy and efficiency.

Regarding Gaussian mixture models, Myronenko *et al.* [MSCP*07, MS10] proposed a CPD algorithm. They formulated the registration as a maximum likelihood estimation problem, where one point set moves coherently to align with the other set under motion coherence constraint over the velocity field. The motion coherence theory [YG89] considers two adjacent points that tend to move coherently, and this motion coherence is an important feature that influences the smoothness of the transformation. Golyaniet *et al.* [GTRS16] extended the CPD registration algorithm using correspondence priors and a coarse-to-fine optimization strategy to achieve robust non-rigid point registration with an improved speed of the registration process. Bai *et al.* [BYG17] proposed a statistical framework by aligning two-point sets represented by Gaussian mixture models. Hirose [Hir21] proposed the Bayesian Coherent Point Drift (BCPD) method, which formulates CPD in a Bayesian setting to improve registration accuracy and efficiency. These methods achieved good results for the connection but failed to register the separation of two objects. The main reason is that the CPD-based framework requires all points to be transformed coherently as a whole (*e.g.* a single point set) whose displacement must meet the CPD condition. When there are separations and connections during the object
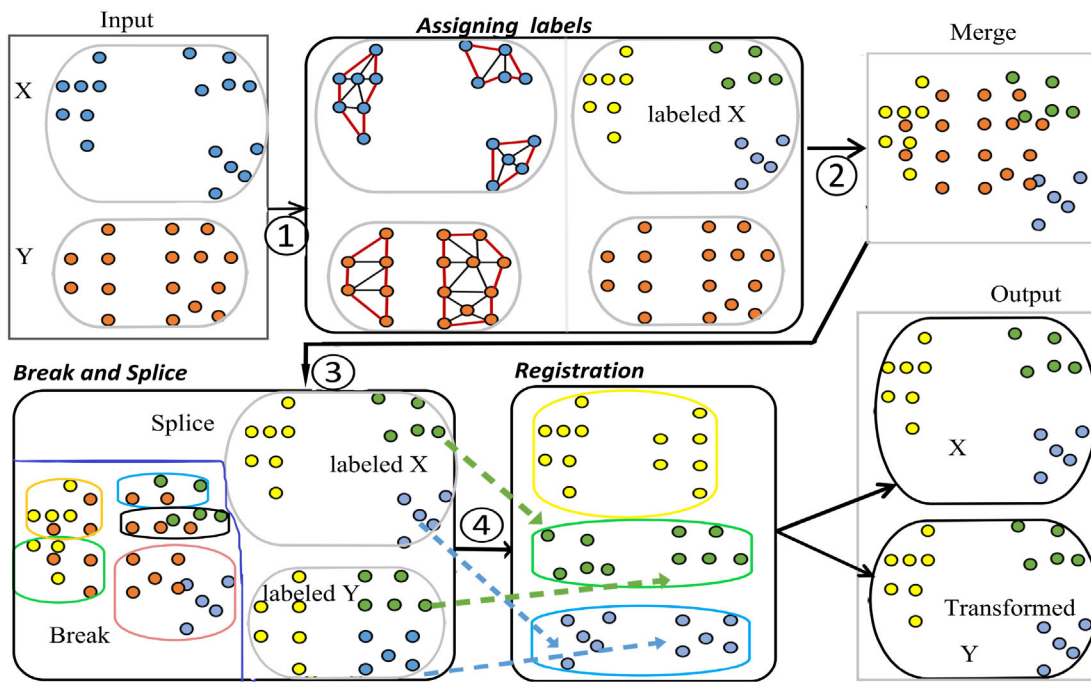
Figure 2: *Overview of* Break and Splice *registration framework (The different colours mean different labels, and the black arrow indicates the process of our non-rigid registration.):* Step 1, *extracting boundaries of point sets to determine the partition template and allocating labels to the partition template* **X**. *In* Step 2, *the labelled point set* **X** *and the unlabelled point set* **Y** *are merged into one.* Step 3, *assigning labels to point set* **Y** *according to the labels of partition template* **X**. *Specifically, clustering the merged point sets into different groups and assigning labels to unlabelled points in each cluster.* **X** *and* **Y** *will be re-assembled by* Splice *together the points in different clusters, respectively.* Step 4, *point sets with the same labels are registered to obtain the transformed source point set* **Y** *(The dashed arrows indicate registration with the same labels.).*

topology change, as shown in examples in Figure 1, the point set will separate into two sets. Thus, CPD-based methods would fail to address the non-rigid registration challenges to be solved in our paper. Recently, Zampogiannis *et al.* [ZFA21] proposed a framework to address the issues of separation and connection. However, this method did not work well on large inter-frame motions, as shown in our comparative studies (Section 4).

Non-rigid registration methods have been applied to dynamic reconstructions [SBI18, NFS15, DKD*16, GXY*17]. Newcombe *et al.* [NFS15] proposed a DynamicFusion system that fused the live frame depth maps into the canonical space via the estimated warp field to achieve high quality. However, the experiment in DynamicFusion does not deal with large inter-frame motions with object topology change issues of separations and connections. The Fusion4D [DKD*16] and Kaiwen [GXY*17] used RGBD camera inputs to reconstruct dynamic scenes simultaneously. Although Kaiwen's method tackles the connection, object separation remains unsolved. Slavcheva *et al.* [SBI18] and Li and Guo [LG20] propose new methods to tackle this issue by incorporating volumetric data. However, the 3D detail information of volume-based methods is lower than that of other point-based registration methods.

Although CPD-based methods are inadequate in achieving non-rigid registration of point sets with separations directly, we can leverage the power of clustering methods to separate the topology

of a source point set and then match the clusters based on non-rigid registrations. In terms of clustering, the DPGMM [Fer73] infers the number of clusters from the data set instead of setting a prescribed number of clusters as traditional clustering methods. As for registration, the BCPD method [Hir21] has shown point sets registrations with large deformation, high accuracy and less computational cost.

In this paper, we proposed a *Break and Splice* registration framework that integrates DPGMM and BCPD to achieve non-rigid registration with topology changes. We also utilize a statistical approach to refine the clusters to increase the match of the similar topologies of the point sets to improve the accuracy and efficiency of both DPGMM and BCPD in partitions and registration. In the following sections, we detail our *Break and Splice* registration framework and methodology.

3. Methodology

Given two 3D point sets **X** and **Y**, our proposed framework aims to handle registrations between two point sets **X** and **Y** that exhibit *Connection* and *Separation* topology changes. The framework *Break and Splice* is designed to reproduce the states of *Connection* and *Separation* between point sets. For brevity, we use a *Separation* example to introduce our framework, where **X** is the target point set, and **Y** is the source point set, as shown in Figure 2.
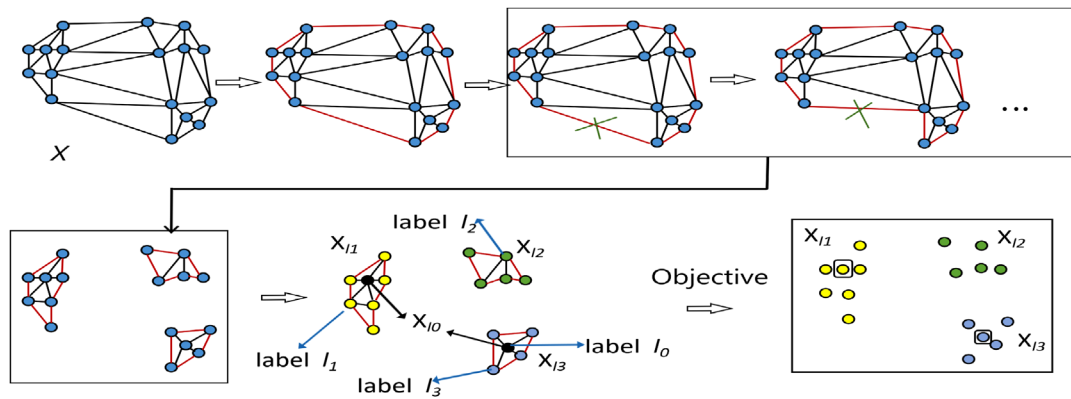
B. Bao & H. Fu / EG Author Guidelines

Figure 3: *Assigning labels for* **X***: The top shows the process of extracting boundaries using triangulation, and the bottom illustrates the allocating labels* l_1, l_2 *and* l_3 *based on extracted boundaries. The red is the extracted boundaries. Black circles are the inner points* $\mathbf{X}_{l_0}$. *Our objective is to allocate labels to those inner points.*

The *Break and Splice* framework can be divided into three modules: (a) *Assigning labels* to aim to determine the partition template (point sets with more boundaries, such as **X**) to be allocated different labels; (b) *Break and Splice* assigns labels to **Y** based on the labels of **X**; (c) *Registration* utilizes the BCPD algorithm to achieve registration between point sets with same labels.

We can take four steps to achieve *Separation* between the point sets **X** and **Y**. *Step 1*, we determine the partition template by using Delaunay triangulation to extract boundaries of the target and source point sets and assign labels for each point of partition template **X** based on the boundaries in the *Assigning labels module* (Section 3.1). *Step 2*, we mix the source and target point sets into one to reduce the cluster difference that will occur in the *Break and Splice module* (Section 3.2). *Step 3*, in the *Break and Splice module*, we utilize the DPGMM to cluster point sets and assign labels to **Y** according to the labels of **X** in each cluster. We then splice together the partitions of points with the same labels to generate point subsets that need to be registered. *Step 4*, in the *Registration* module (Section 3.3), the BCPD method is used to register the source point set groups and the target groups which have the same labels.

3.1. Assigning labels

The boundary extracting approach [Awr16] is used to identify the boundaries of point sets **X** and **Y**. We select the point sets with more boundaries as the partition template, which will be allocated to different labels. We take target point set **X** as the example for boundary identification and assigning labels, for brevity's sake, as shown in Figure 3.

3.1.1. *Boundary identification*

The point set **X** is projected to a 2D plane, and the Delaunay triangulation is formed on plane points (shown in Figure 3). One of the triangle sides along the periphery of **X** is associated with only one triangle. All such edges on that side form the initial boundary. Let d_{max} indicate the maximum distance between the near-

est neighbouring points in **X**. Any initial boundary edges whose length exceeds $2d_{max}$ will be removed (d_{max} is achieved by K-nearest-neighbour). The removal of long boundary edges continues iteratively until every edge along the boundary is at most d_{max} in length.

3.1.2. *Assigning labels for the target point set*

Once the boundary of **X** is identified, different connected components are assigned different labels. We can get an adjacency matrix after using Delaunay triangulation to extract the boundaries of **X**. The elements in the adjacency matrix represent the number of triangles that share the same edges connected by nodes. Then the breadth-first search (BFS) method is used to search for connected components. Assuming that there are three different connected components by boundary identification from **X**, the nodes/points in the same connection component will be assigned the same label. The points in **X** will be allocated the labels $L_x = \{l_1, l_2, l_3\}$.

3.2. Break and splice

The Break and Splice framework is similar to that of the bisection method. The Break process will not end until the category of the labels of target points in each cluster is unique. Just like the bisection method, the process of splitting the interval will not end until the solution of a continuous function is found. This framework aims to assign the labels of target points to the source points. The reason to set the process of assigning labels to source points as a binary tree structure is that there is more than one way to assign labels to source points in one cluster after only one partition. To make the label of source points in one cluster unique, we need to re-partition the cluster where the labels of the target points are different, which will be repeated until the category of the labels of target points is unique in one cluster.

After attaining the labels L_x for the points set **X**, the goal then is to allocate labels L_x to the points set **Y**. Assigning L_x to **Y** can be regarded as the process of *Break* and *Splice*. *Break* involves the
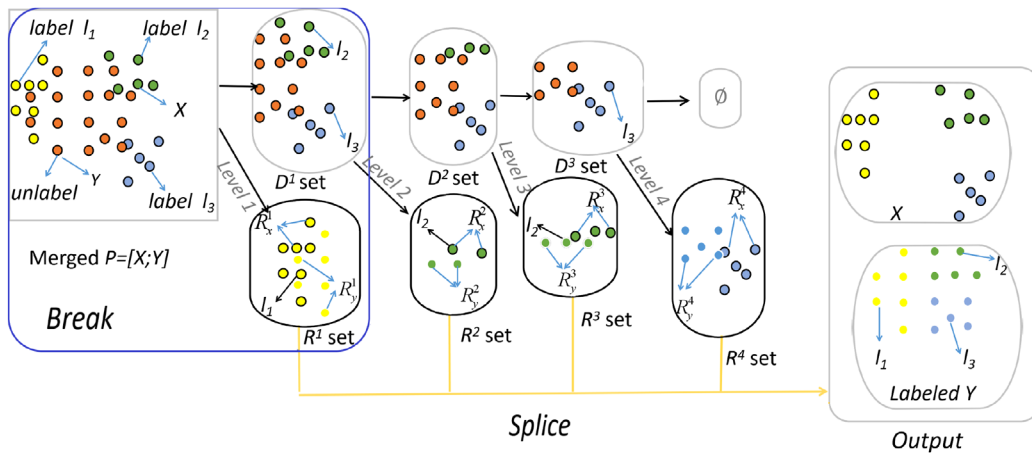
Figure 4: *The structure of the* Break and Splice *module for assigning labels to source points set* **Y** *(The black arrow indicates the process of our* Break and Splice*): The blue box shows* Break*, which involves the partitions and labels allocation of point sets. The tangerine arrows represent* Splice*, which indicates the stitching of the partitions based on the labels. The* Break *is a binary tree with merged point sets as a root node, and the leaf node R^j set keeps the points that will not be re-partitioned at the $j + 1$ level, and the branch node D^j set contains the points to be divided. R_x^j and R_y^j represent the target points, and source points in the R^j set, $\varnothing$ demonstrates that there is no point left to be partitioned, which marks the end of the partition.* Splice *re-assembles the points (R_x^j and R_y^j) in different R^j set to recover the labelled source point set* **Y** *and target point set* **X***.*

partition of the merged point sets $\mathbf{P} = [\mathbf{X}; \mathbf{Y}]$ and the label allocation in each partition, which is the key, to handling the topology changes between point sets for registration. *Splice* is to splice the partitions of **Y** together based on the allocated labels. Figure 4 shows how the *Break and Splice* process is carried out. *Break* can be regarded as a binary tree generation process. The root node of the binary tree is the merged points set $\mathbf{P} = [\mathbf{X}; \mathbf{Y}]$, and the leaf nodes R^j set include the target point sub-sets R_x^j and the source point sub-sets R_y^j. The branch node D^j set keeps the merged points sub-set that will be divided at the $j + 1$ level. The binary tree grows until one of the leaf nodes is empty $\varnothing$. It is worth noting that the points in each leaf node R^j set have the same labels, and the points in branch node D^j set have different labels. Namely, the number of category labels in nodes determines whether the node is a leaf node or a branch node. *Splice* recovers the labelled target points set **X** and labelled source points set **Y** by re-assembling the R_x^j and R_y^j, respectively.

The process of *Break* can be divided into *Cluster* and *Refine*. *Cluster* is to utilize DPGMM [Fer73] to attain the initial partitions C_k, $k \in \{1, \ldots, K\}$, where K is the number of partitions. The advantage of DPGMM is that it automatically discovers the number of clusters and is likely to converge to the data's actual clusters [MFHM19]. To guarantee the consistency of the partitions for target points set X and source points set Y, we take *Refine* to overcome the significant irregularity of the initial partitions. Based on the pruned partitions, we allocate the labels of target points C_x^k to the source points C_y^k. Figure 5 illustrates the structure of *Break* at *Level 1* of the binary tree. Especially, it is different from *Step 1*. In *Step 1*, the KNN is used to search the K points close to the initial point, and then the K points will be assigned the label of the initial point. DPGMM, as a clustering method, clusters the merged points according to the coordinates of points without defining the number of clusters (such as the K in the K-means clustering method).

3.2.1. *Cluster*

Assume the mixed point set **P** has been divided into K clusters by DPGMM. $\mathbf{P} = \{\mathbf{C}_1, \ldots, \mathbf{C}_K\}$ and $\mathbf{C}_k = \{\mathbf{C}_x^k, \mathbf{C}_y^k\}$, $k \in \{1, \ldots, K\}$. $\mathbf{C}_x^k$ and $\mathbf{C}_y^k$ are the target points sets and source points sets in cluster $\mathbf{C}_k$, respectively. The number of target points in cluster $\mathbf{C}_k$ is N_k. And the label of target points $\mathbf{x}_i$ in cluster $\mathbf{C}_k$ is denoted as $L_{\mathbf{x}_i}^k$. The K clusters are divided into R' *set* (R' set includes target points with a single label and source points after cluster) and D' *set* based on the label category of target points $\mathbf{C}_x^k$. Especially, even if no cluster forms an R' *set* during the initial iteration, the original mixed point cloud will be divided into several clusters, and the distribution of the target point and the source point in each cluster is similar. Therefore, with the refinement and clustering of the mixed point sub-sets, there will always be an R' *set* where the labels of target points are unique.

$$\tau_k = \prod_{i=1}^{N_k} \delta \left(L_{\mathbf{x}_i}^k - \frac{\sum_{i=1}^{N_k} L_{\mathbf{x}_i}^k}{N_k} \right) \tag{1}$$

where δ is the *Dirac function*. If $\tau_k = 1$, then $\mathbf{C}_k$ belongs to R' *set*, otherwise $\mathbf{C}_k$ belongs to D' *set*. Equation (1) indicates that the R' *set* includes those clusters where the label categories of $\mathbf{C}_y^k$ are unique. D' *set* consists of those clusters where the label categories of $\mathbf{C}_y^k$ are various.

3.2.2. *Refine*

The irregularity of the initial partitions achieved by DPGMM will affect the subsequent partition results, confusing label allocation. Furthermore, the confusion label will increase the error of registration. Thus, we propose *Refine* to prune the clusters in R' *set*. The regularized clusters compose the R *set*, which will be spliced together
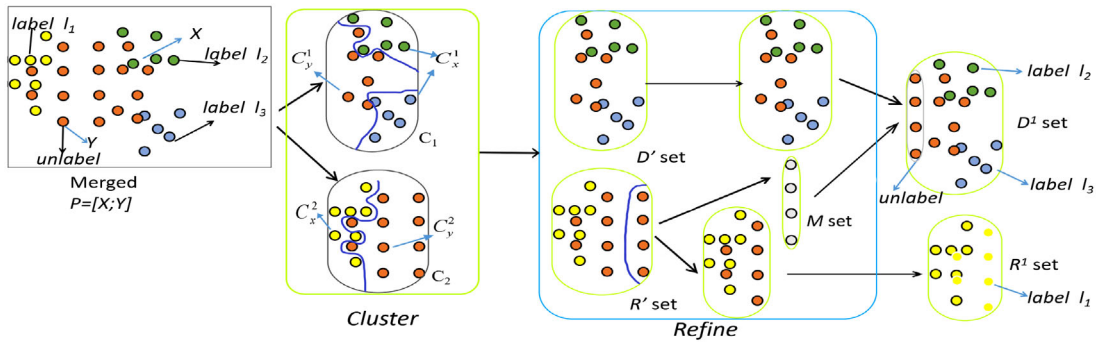
Figure 5: *The structure of* Break *at* Level 1 *of Figure 4 (The black arrow indicates the process of the* Break *at Level 1).* Cluster *involves the initial partitions (C_1 and C_2) of merged points using the DPGMM clustering method. The target points sub-set C_x^1 in C_1 have different labels and C_2 contains the target points sub-set C_x^2 with the same labels. Besides, the cluster C_1 and cluster C_2 also include the source points sub-set C_y^1 and C_y^2. The clusters with single labels form the R' set. D' set includes those clusters with different labels.* Refine *aims to overcome the significant irregularity of R' set. The irregular point sets are selected as M set to be mixed with D' set to generate the brunch node D^1 set, which will be divided at* Level 2*. The R' set's remaining source points will then be allocated the labels of the target points in the same clusters. The target points and the source points with the same labels form the R^1 set, which is the leaf node in Figure 4.*

to recover the labelled source points set **Y**. We design the refining path and direction according to the characteristics of point distribution. Then, the label L_x will be assigned to the source points $\mathbf{y}_j$ in the regularized clusters.

Assume that $\mathbf{C}_{\hat{k}} = [\mathbf{C}_x^{\hat{k}}, \mathbf{C}_y^{\hat{k}}]$ is a cluster in R' set, and $\mathbf{C}_x^{\hat{k}}, \mathbf{C}_y^{\hat{k}}$ are target points and source points in $\mathbf{C}_k$, respectively. The degree of dispersion of the $\mathbf{C}_x^{\hat{k}}, \mathbf{C}_y^{\hat{k}}$ in each dimension determines the path for refining points.

$$Std_{\mathbf{x}}^d = \sqrt{\frac{\sum_{i=1}^{N_k} \left(\left[\mathbf{C}_x^{\hat{k}}\right]_{i,d} - \overline{\left[\mathbf{C}_x^{\hat{k}}\right]}_{:,d} \right)}{N_k - 1}} \qquad (2)$$

$$Std_{\mathbf{y}}^d = \sqrt{\frac{\sum_{i=1}^{n_k-N_k} \left(\left[\mathbf{C}_y^{\hat{k}}\right]_{i,d} - \overline{\left[\mathbf{C}_y^{\hat{k}}\right]}_{:,d} \right)}{N_k - 1}} \qquad (3)$$

where $[\mathbf{C}^{\hat{k}}]_{i,d}$ represents the coordinate of the ith point in the dth dimension. $\overline{[\mathbf{C}^{\hat{k}}]}_{:,d}$ indicates the mean value of the coordinates of all points in the d_{th} dimension. $Std_{\mathbf{x}}^d$ and $Std_{\mathbf{y}}^d$ show the standard deviation of $\mathbf{C}_x^{\hat{k}}, \mathbf{C}_y^{\hat{k}}$ in dth dimension, $d \in \{1, 2, 3\}$.

$$d^* = \arg\max_{d \in \{1,2,3\}} \left| Std_{\mathbf{x}}^d - Std_{\mathbf{y}}^d \right| \qquad (4)$$

d^* involves the refining path, in which the object has the largest deformation. For example, the stretching of an object in one dimension will lead to its squashing in another dimension. For example, when $d^* = 1$, we will refine points along with X-axis. If $\left| Std_{\mathbf{x}}^{d^*} - Std_{\mathbf{y}}^{d^*} \right| < \gamma$, there is no need to refine the $\mathbf{C}_{\hat{k}}$. Otherwise, we must also determine the refining direction based on the refining path.

$$Ske_{\mathbf{y}} = \frac{\frac{1}{n_k-N_k} \sum_{j=1}^{n_k-N_k} \left(\left(\left[\mathbf{C}_y^{\hat{k}}\right]_{j,d^*} - \overline{\left[\mathbf{C}_y^{\hat{k}}\right]}_{:,d^*} \right) \right)^3}{\left[\frac{1}{n_k-N_k-1} \sum_{j=1}^{n_k-N_k} \left(\left[\mathbf{C}_y^{\hat{k}}\right]_{i,d^*} - \overline{\left[\mathbf{C}_y^{\hat{k}}\right]}_{:,d^*} \right)^2 \right]^{\frac{2}{3}}} \qquad (5)$$

$Ske_{\mathbf{y}}$ shows the skewness of the source points $\mathbf{C}_y^{\hat{k}}$ in the d^*th dimension. We denote the index of the minimal $[\mathbf{C}_y^{\hat{k}}]_{j,d^*}$ as J_{min} and the index of the maximal $[\mathbf{C}_y^{\hat{k}}]_{j,d^*}$ as J_{max}. If $Ske_{\mathbf{y}} < 0$, we will separate the $[\mathbf{C}_y^{\hat{k}}]_{J_{min},:}$ from $\mathbf{C}_y^{\hat{k}}$ and move it into a temporary set M set. If $Ske_{\mathbf{y}} > 0$, we will move the $[\mathbf{C}_y^{\hat{k}}]_{J_{max},:}$ into M set. $[\mathbf{C}_y^{\hat{k}}]_{j,:}$ represents the jth point in $\mathbf{C}_y^{\hat{k}}$. It is worth noting that the initial M set is an empty set.

Figure 6 takes the C_2 in Figure 5 as an example to illustrate the relationship between $Ske_{\mathbf{y}}$ and M set. Because of the positive skewness of the source points $\mathbf{C}_y^2$ on the x-axis, we transfer the source points with the maximal x-coordinate to the M set.

Repeat calculating Equations (2)–(5) to determine the refining path and direction until $|Std_{\mathbf{x}}^{d^*} - Std_{\mathbf{y}}^{d^*}| < \gamma$. The pseudocode for refining the $\mathbf{C}_y^{\hat{k}}$ in R' set is shown in Alg. 1.

After being separated from R' set, M set will be merged with D' set to generate the branch node D set to be divided again. Also, the labels of $\mathbf{C}_x^{\hat{k}}$ will be assigned to $\mathbf{C}_y^{\hat{k}}$. The process for clustering and refining will be repeated until there are no points in D set to be divided.

As for *Splice*, it is a process to re-assemble the labelled source points set **Y** and the target points set **X**. We denote the source group with label l as $\mathbf{G}_{\mathbf{y}}^l$ and the target group with label l as $\mathbf{G}_{\mathbf{x}}^l$. Splicing the R set together can reduce the number of partitions and achieve less computational cost for registration to improve registration accuracy.

3.3. Registration

BCPD algorithm [Hir21] is used to register the source group $\mathbf{G}_{\mathbf{y}}^l$ and target group $\mathbf{G}_{\mathbf{x}}^l$. BCPD is the Bayesian formulation of the CPD [MS10]. The key difference between BCPD and CPD is that BCPD defines motion coherence using a prior distribution instead of
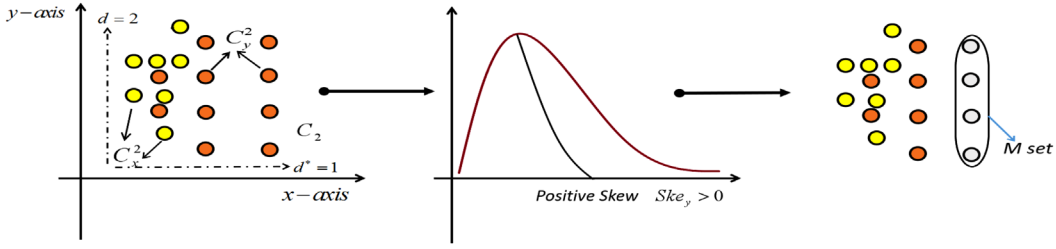
Figure 6: *Illustration for the impact of* skewness *on* Refine. *Due to the positive skew of the* $\mathbf{C}_y^2$ *on the x-axis, the points in the* $\mathbf{C}_y^2$ *with the largest* x-*coordinate will be transferred to the* M set.

Algorithm 1. Refine $\mathbf{C}_y^{\hat{k}}$ in R set

Input: $\mathbf{C}_x^{\hat{k}}, \mathbf{C}_y^{\hat{k}}, \gamma$
Output: $\mathbf{C}_y^{\hat{k}}$, M set
Initialize $M = \varnothing$
repeat

    Calculate Eq. 2, Eq. 3 and Eq. 4 to get $Std_{\mathbf{x}}^d, Std_{\mathbf{y}}^d, d^*$.

    if $\left| Std_{\mathbf{x}}^{d^*} - Std_{\mathbf{y}}^{d^*} \right| > \gamma$ **then**

        Calculate Eq. 5 to get $Ske_{\mathbf{y}}$ and J_{min}, J_{max}; **if** $Ske_{\mathbf{y}} < 0$

        then

            $M\ set = \left[M\ set; \left[\mathbf{C}_y^{\hat{k}} \right]_{J_{min},:} \right]; \left[\mathbf{C}_y^{\hat{k}} \right]_{J_{min},:} = \varnothing;$

        else

            $M\ set = \left[M\ set; \left[\mathbf{C}_y^{\hat{k}} \right]_{J_{max},:} \right]; \left[\mathbf{C}_y^{\hat{k}} \right]_{J_{max},:} = \varnothing;$

until $\left| Std_{\mathbf{x}}^{d^*} - Std_{\mathbf{y}}^{d^*} \right| < \gamma;$

---

the regularization term in CPD. Besides, the transformation model in BCPD is a combination of non-rigid and similarity transformations, which enables it to handle the registration task with large deformation. As for the computational time, BCPD uses the Nyström method [WS01] and the KD tree search [Ben75] to accelerate registration without losing registration accuracy.

The key models in the BCPD algorithm can be generalized as follows:

Transformation model:

$$\boldsymbol{\tau}(\mathbf{y}_i) = T(\mathbf{y}_i + \mathbf{v}_i) = s\mathbf{R}(\mathbf{y}_i + \mathbf{v}_i) + \mathbf{t} \tag{6}$$

where $s \in \mathbb{R}$ is a scale factor, $\mathbf{R} \in \mathbb{R}^{D \times D}$ is a rotation matrix, $\mathbf{t} \in \mathbb{R}^D$ is a translation vector and $\mathbf{v}_i \in \mathbb{R}^D$ is a displacement vector that characterizes a non-rigid transformation.

Prior distribution:

$$p(\mathbf{v}|\mathbf{y}) = \phi(\mathbf{v}; \mathbf{0}, \lambda^{-1}\mathbf{G} \otimes \mathbf{I}_D) \tag{7}$$

where λ is a positive constant and $\otimes$ denotes the Kronecker product. $\mathbf{G} = (g_{mm'}) \in \mathbb{R}^{M \times M}$ with $g_{mm'} = \mathcal{K}(\mathbf{y}_m, \mathbf{y}_{m'})$ is a positive definite matrix, where $\mathcal{K}(,)$ is a positive-definite kernel. $\phi(\mathbf{v}; \mathbf{0}, \lambda^{-1}\mathbf{G} \otimes \mathbf{I}_D)$

is the *multivariate normal distribution* of $\mathbf{v}$ with a mean vector $\mathbf{0}$ and a covariance matrix $\lambda^{-1}\mathbf{G} \otimes \mathbf{I}_D$.

4. Experiment

We evaluate our approach by performing experiments on five data sets, three public data sets and two of our own data sets. The experiments are implemented on Intel Xeon CPU E5-1603 @ 2.80 GHz and 32 G RAM.

There are three groups of parameters in our framework: the maximum distance between neighbouring points d_{max} for extracting boundaries in *Assigning labels* module; the hyperparameters $\{\alpha, \rho, \beta, D\}$ for *Cluster* and γ for *Refine* in *Break and Splice* module and λ for registration. We set these three groups of parameters empirically as follows in our experiments: $d_{max} = 0.03$; $\alpha = 1$, $\rho = 1$, $\beta = 3$, $D = 3$; $\gamma = 10^{-3}$ and $\lambda = 10$. In addition, these data sets are filtered by down-sampling to compare the BCPD method and tend to find the distribution and change of every point after registration. Its value is 0.002. At last, these data sets are without background since our non-rigid registration focuses on the deformed objects, and the background does not include the deformation. On the other hand, it is to compare FB-Warp without any background.

We define large inter-frame motions as such that two frames are separated by 20 frames at least, and the gap between objects usually is large on Refs. [IZN*16, SBCI17], and RGB-D data sets created by ourselves. We test our method against the BCPD method [Hir21] and the FB-Warp [ZFA21] for separation and connection problems with large inter-frame motion. We compare our algorithm and the FB-Warp by quantitative evaluation. In terms of performance metrics, the accuracy of registration is measured by Root Mean Square Error (RMSE), Similarity (AS) [AE18], Structural Similarity using the colour-based feature (SSIM) [AE20] and the computation time measures the efficiency of the algorithms. In these experiments, only the point sets are used to register.

4.1. Non-rigid registration

We choose data sets with dynamic scene topology changes to evaluate our algorithm on large inter-frame motions. A boxing sequence in [IZN*16] as shown in Figure 7 and sequences of Hat and Alex in Ref. [SBCI17] as shown in Figures 9 and 11 are selected because these public data sets include both separations of two objects and deformations. In addition, we created RGB-D data sets achieved by
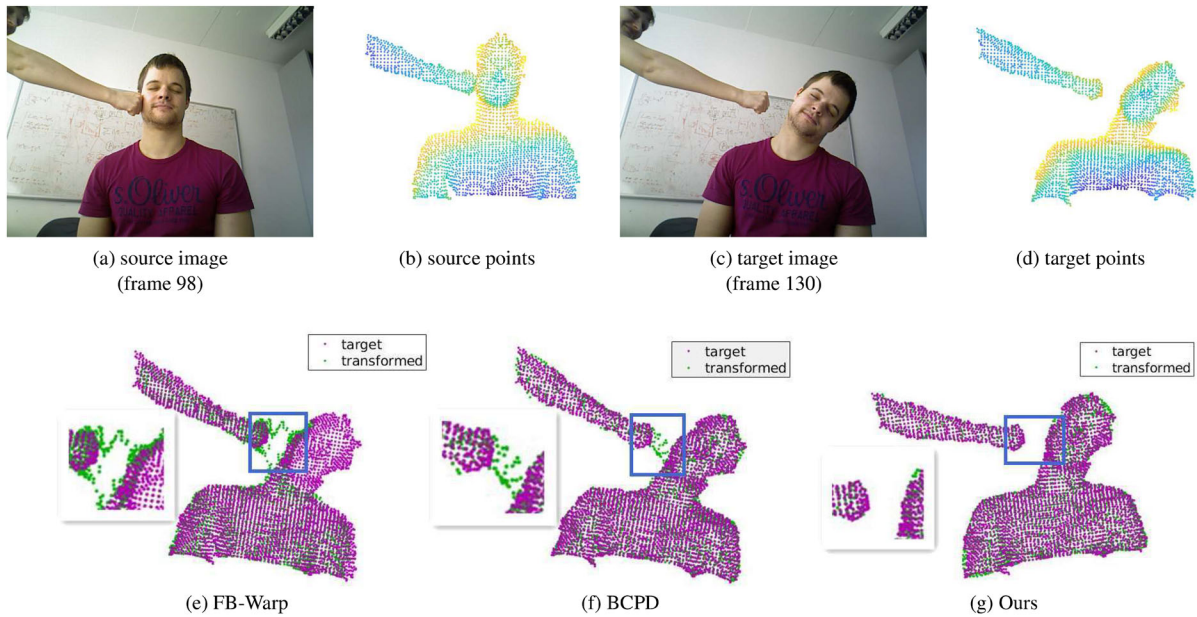
B. Bao & H. Fu / EG Author Guidelines

Figure 7: *The results of Boxing data: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the result of registration from source points to target points, and the blue areas show the main differences: (e) uses the method of FB-Warp, (f) uses the method of BCPD without* Break and Splice *and (g) is our algorithm.*
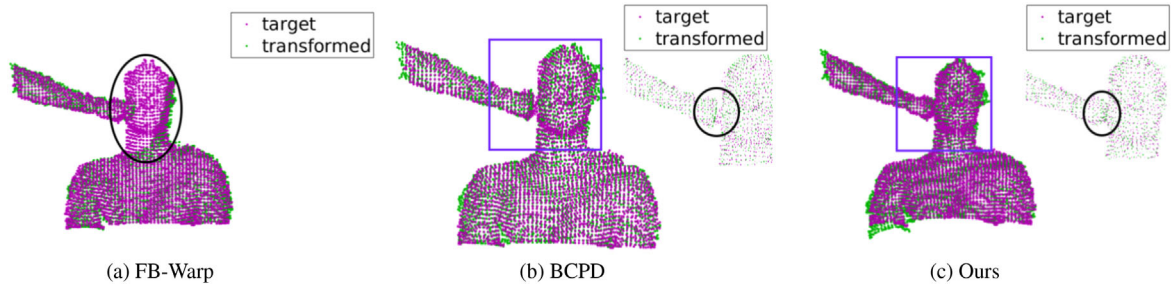

Figure 8: *The results of connection registration on Boxing data and the source point set and target point set in Figure 7 are exchanged (Figure 7d as the source point set): (a) use the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

Kinect v2 as shown in Figures 13 and 15 to demonstrate a rigid bunny separated from a table surface and scenes that consists of the separation of a deformed soft pillow from a table surface. Finally, we experiment with the connection (shown in Figure 16) on the pillow data set.

Figure 7 shows the results of boxing, and source points (frame 95) and target points (frame 130) are obtained by down-sampling without any background. Significantly, body deformation also occurs in addition to the separation. The second row of Figure 7 shows the comparison of registration results. It can be seen that BCPD and FB-Warp are unable to handle registration with the separation, as shown in the final results containing points between the fist and the face (green points). However, our method can effectively register the source and the target points. Similarly, the results of Alex sequences (Figure 9) and Hat data sets (Figure 11) show that our *Break and Splice* framework can achieve a better result than others.

In addition, to experiment with the connection, we exchange the source point set and target point set, and the results are shown in Figure 8 (Boxing data), Figure 10 (Alex data) and Figure 12 (Hat data). The green points in Figure 8a on the face only locate on edge. Although the result of BCPD shows good registration, between the fist and the face only show the target points (the black circle in Figure 8b) compared with ours. For the Alex and Hat data results, there are a few points between the topology changes, but the distribution is uniform in our results.

In order to demonstrate the effectiveness of our *Break and Splice* framework in dealing with different scenes, we use an RGB-D camera to create additional point cloud data sets and compare our method with BCPD and FB-Warp on these data sets. We use a rigid object (bunny, Figure 13) and a non-rigid object (pillow, Figure 15) to conduct the experiment. Figure 13 shows that when the bunny is placed far away from the table, we use this final state as target points
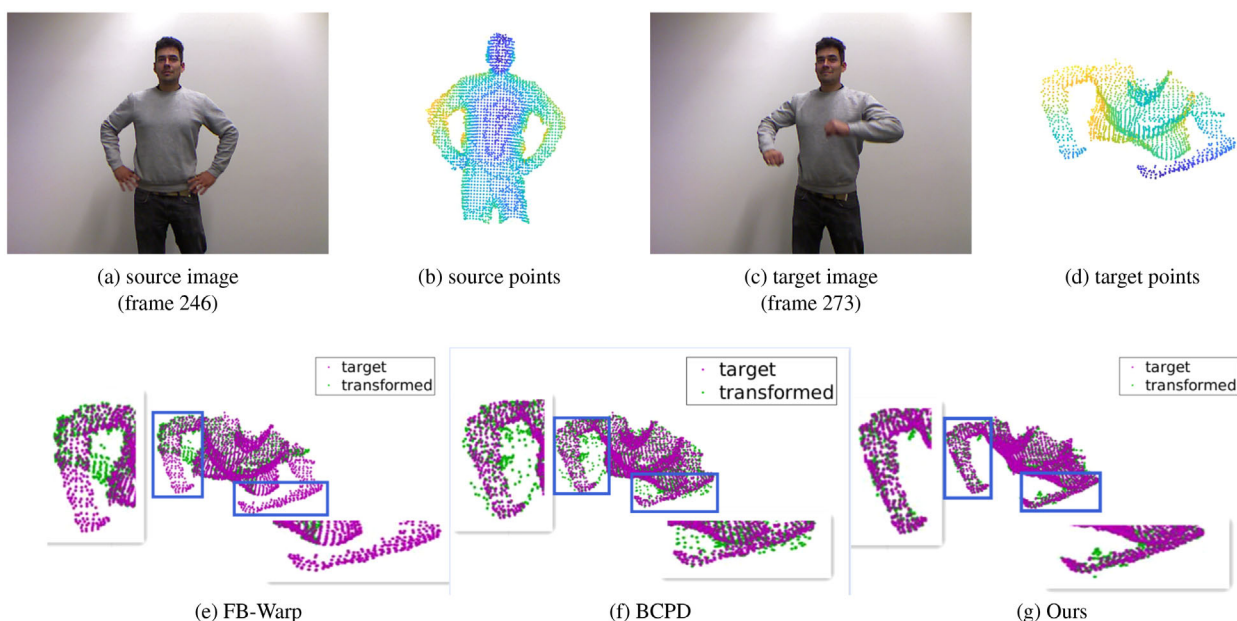
Figure 9: *The results of Alex data: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the results of registration from source points to target points, and the blue areas show the main differences: (e) uses the method of FB-Warp, (f) uses the method of BCPD without* Break and Splice *and (g) is our algorithm.*
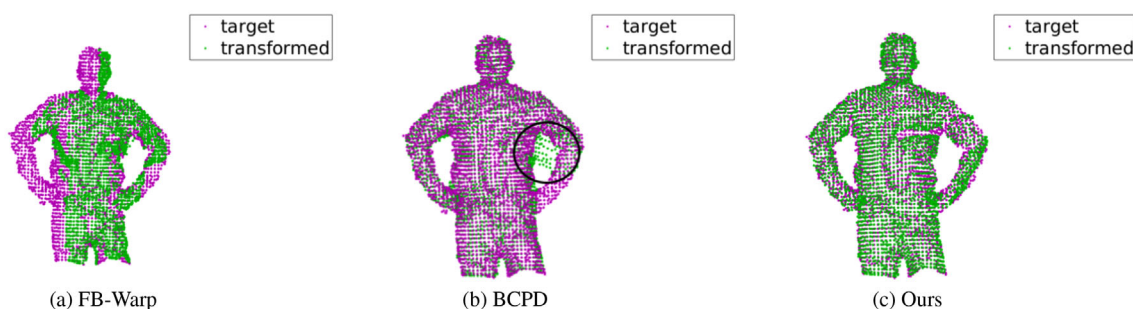

Figure 10: *The results of connection registration on Alex data and the source point set and target point set in Figure 9 are exchanged (Figure 9d as the source point set): (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

and its initial state as source points. Figure 13e shows that although the result of FB-Warp contains no points between the bunny and the table, many green points are distributed on the boundary of the transformed bunny points. The BCPD have many green points between the bunny and the table, as shown in Figure 13f. The result of our method shown in Figure 13g has well-distributed transformed points without any points between the object and the table.

For the connection, Figure 13a is used as the target point set to register. Our method can achieve a suitable result, especially for the bunny. The result of FB-Warp shows most of the green points of the bunny distribute the head of the bunny, and the result of BCPD fails to register the bunny (in Figure 14b).

In the Pillow experiment (Figure 15), the data set simultaneously includes significant deformation and separation. Our method pro-

duces a better result than that of FB-Warp and BCPD. In the FB-Warp, the transformed points appear at the bottom of the pillow, and some points remain between the pillow and the table. In Figure 15f, although the transformed points are well-distributed for a pillow, it fails to handle the separation between the pillow and the table. Whereas in Figure 15g, our *Break and Splice* can effectively deal with combined deformation and separation events and yield a significantly improved result.

In addition, we exchange the source point set and target point set to experiment with connection and the results shown in Figure 16. It can be seen that the BCPD and our method can achieve good registration with well-distributed green points, but the FB-Warp has a few green points (transformed point set) on the bottom half of the pillow.
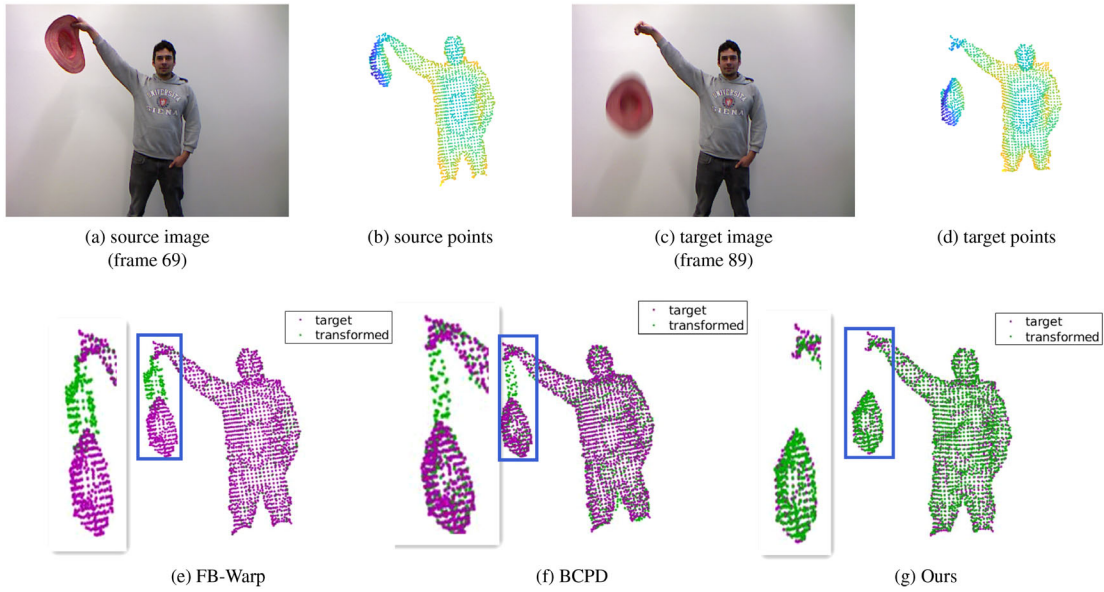
| (a) source image (frame 69) | (b) source points | (c) target image (frame 89) | (d) target points |

| (e) FB-Warp | (f) BCPD | (g) Ours |

Figure 11: *The results of Hat data: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the results of registration from source points to target points, and the blue areas show the main differences: (e) uses the method of FB-Warp, (f) uses the method of BCPD without* Break and Splice *and (g) is our algorithm.*

| (a) FB-Warp | (b) BCPD | (c) Ours |
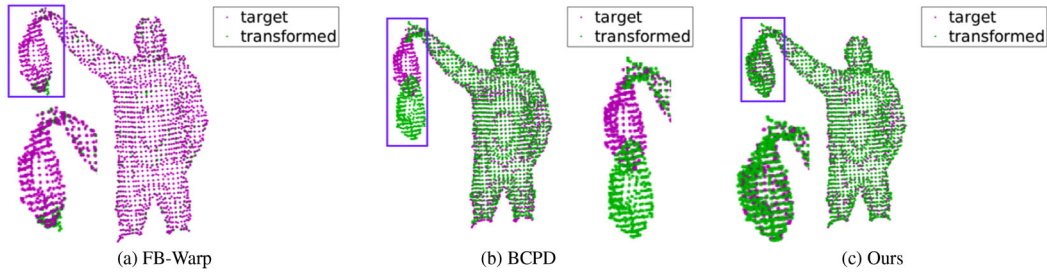
Figure 12: *The results of connection registration on Hat data and the source point set and target point set in Figure 11 are exchanged (Figure 11d as the source point set): (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

4.2. Quantitative evaluation

We evaluate the accuracy and the cost of computation of our non-rigid registration framework using RMSE (Equation 8), AS (Equations 9–11) and SSIM (Equation 12) as measurement metrics tested on two consecutive frames (Table 2) and large inter-frame motions (Table 3) with topology changes. RMSE efficiently measures the registration error. Since the ground truth of non-rigid registration is the target point set, we use AS to compare the similarity between the target point set and the transformed point set. If the value of AS is closer to 1, the transformed point sets are the more similar target point sets. The similarity of colour can also be an important metric because the colour is not influenced by registration, and it is easy to find the difference between transformed point sets and target point sets. Table 1 shows the number of point sets during the experiment.

$$RMSD = \sqrt{\frac{\sum_{i=1}^{N}(y_i - x_i)^2}{N}} \qquad (8)$$

Table 1: *Number of point sets.*

	Alex	Boxing	Hat	Bunny	Pillow
Source point set	1264	2813	1683	2411	2244
Target point set	1294	2786	1731	2145	2504

where y_i is transformed point sets, x_i is a target point sets, N is the number of target point sets.

$$s_{X,Y} = \sqrt{\frac{\sum_{i=1}^{M}\left(1 - \frac{2}{\pi} * arccos\left(\left|\frac{\vec{n}_i^y \cdot \vec{n}_i^x}{\|\vec{n}_i^y\|\|\vec{n}_i^x\|}\right|\right)\right)^2}{M}} \qquad (9)$$

$$s_{Y,X} = \sqrt{\frac{\sum_{j=1}^{N}\left(1 - \frac{2}{\pi} * arccos\left(\left|\frac{\vec{n}_j^y \cdot \vec{n}_j^x}{\|\vec{n}_j^y\|\|\vec{n}_j^x\|}\right|\right)\right)^2}{N}} \qquad (10)$$
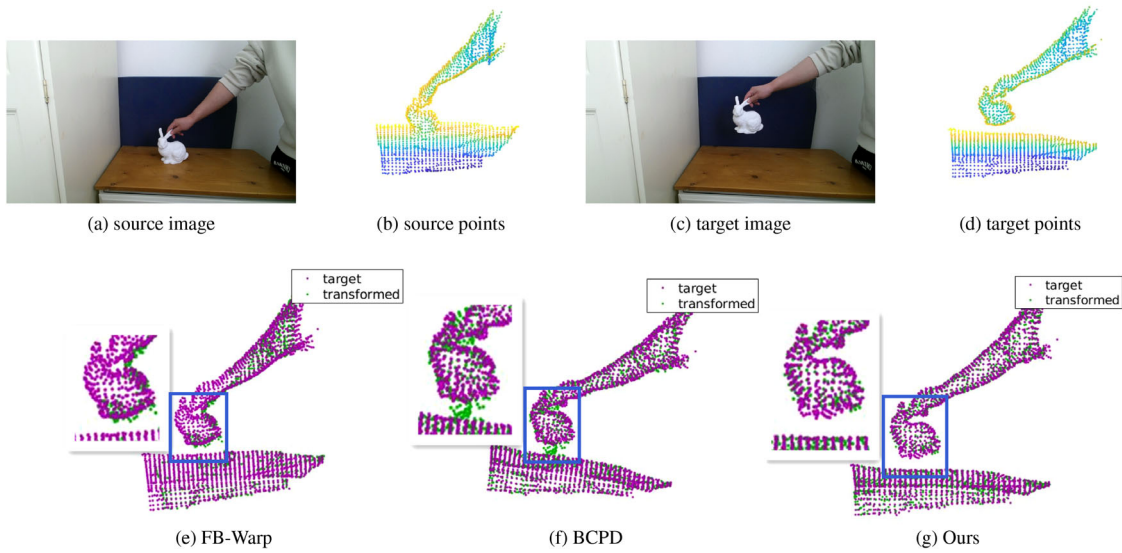
Figure 13: *The results of our data set with Bunny: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the results of registration from source points to target points, and the blue areas show the main differences: (e) uses the method of FB-Warp, (f) uses the method of BCPD without* Break and Splice *and (g) is our algorithm.*
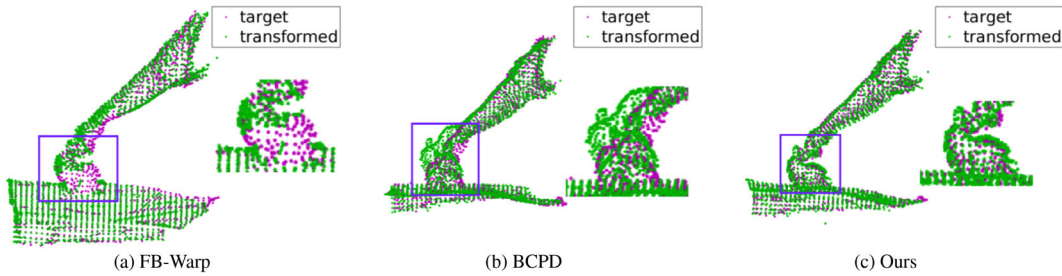

Figure 14: *The results of connection registration on Bunny data and the source point set and target point set in Figure 13 are exchanged (Figure 13d as the source point set): (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

$$AS = min\{s_{X,Y}, s_{Y,X}\} \qquad (11)$$

where Y is the transformed point sets, X is a target point sets, $s_{X,Y}$ is the score of angular similarity with Y as the reference point set and $s_{Y,X}$ is the score of angular similarity with X as the reference point set. $vecn^x$ and $vecn^y$ are the normal of X and Y point sets, N and M are the number of Y and X point sets.

$$SSIM_{pointcolour} = \frac{1}{N} \sum_{i=1}^{N} \left(1 - \frac{|F_X(q) - F_Y(p)|}{max\{|F_X(q)|, |F_Y(p)|\} + \varepsilon} \right) \quad (12)$$

where Y is the transformed point sets, X is a target point sets, F is the feature based on colour [AE20], each neighbourhood of Y is associated with a neighbourhood of X, by identifying for every point p of Y its nearest point q in X. ε equals the machine rounding error for floating point numbers, and N is the number of Y point sets.

As shown in Table 2, our method is more accurate than FB-Warp and BCPD, and the average error is lower by about 60% than the

Table 2: *Registration error (consecutive frames).*

	Alex	Boxing	Hat	Bunny	Pillow	Overall
RMSE (FB-Warp)	0.0170	0.0037	0.0208	0.0089	0.0526	0.0206
RMSE (BCPD)	0.0131	0.0037	0.0135	0.0181	0.0214	0.0140
RMSE (Ours)	**0.0110**	**0.0029**	**0.0092**	**0.0074**	**0.0083**	**0.0078**
AS (FB-Warp)	0.9508	**0.9923**	0.9113	0.9836	0.9577	0.9591
AS (BCPD)	0.8924	0.9105	0.8250	0.8680	0.6965	0.8385
AS (Ours)	**0.9820**	0.9839	**0.9940**	**0.9910**	**0.9964**	**0.9895**
SSIM (FB-Warp)	0.6246	**0.6711**	0.5859	0.5922	0.5120	0.5971
SSIM (BCPD)	0.6091	0.5905	0.5626	0.5740	0.4991	0.5671
SSIM (Ours)	**0.6557**	0.6571	**0.6605**	**0.6314**	**0.6360**	**0.6481**

The best results in each category are in bold.

FB-Warp. The results of the AS and the SSIM show that the average values of our method are higher by about 3.2 and 8.5%, respectively. FB-Warp can achieve better results for the Boxing data since the
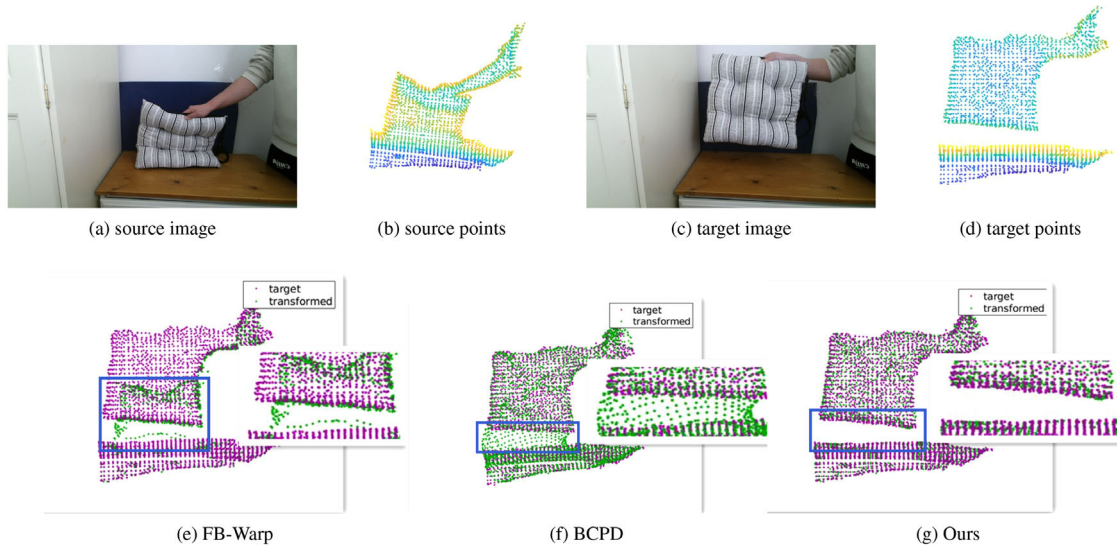
Figure 15: *The results of our data set with Pillow: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the results of registration from source points to target points, and the blue areas show the main differences: (e) uses the method of FB-Warp, (f) uses the method of BCPD without* Break and Splice *and (g) is our algorithm.*
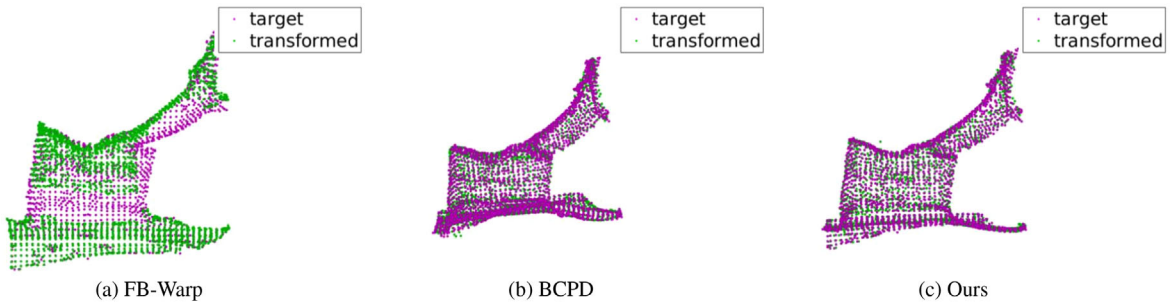

Figure 16: *The results of connection registration on pillow data and the source point set and target point set in Figure 15 are exchanged (Figure 15d as the source point set): (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

deformation is slight between the adjacent frames. Our method can achieve the best results on all data for the large inter-frame motions (Table 3). At the same time, registration time is lower than FB-Warp, as shown in Table 4 (C means consecutive frames, and L means large inter-frame motions). In this table, the Partition times are the *Break and Splice*, and the registration times are the total times of two parts registration by BCPD.

4.3. Evaluation with Gaussian noise

In order to evaluate the robustness of the proposed method against noise, we perform experiments with Gaussian noise in the source points and target points, respectively. In this section, Figure 11b is source points and Figure 11d is target points. For the first test, we sample noise from Gaussian distribution for each point in the source point cloud, where the mean and the standard deviation are $mean(source points)$ and $\alpha * std(source points)$ ($\alpha \in$

Table 3: *Registration error (large inter-frame motions).*

	Alex	Boxing	Hat	Bunny	Pillow	Overall
RMSE (FB-Warp)	0.7694	0.0626	0.8681	0.3337	0.3341	0.4736
RMSE (BCPD)	0.4151	0.1375	0.6034	0.1647	0.5427	0.3727
RMSE (Ours)	**0.1506**	**0.0486**	**0.1752**	**0.0995**	**0.1464**	**0.1241**
AS (FB-Warp)	0.7303	0.7711	0.8426	0.7932	0.7215	0.77174
AS (BCPD)	0.8142	0.8462	0.8374	0.7932	0.7605	0.8103
AS (Ours)	**0.8377**	**0.8632**	**0.8611**	**0.8407**	**0.8443**	**0.8494**
SSIM (FB-Warp)	0.4776	0.6277	0.6531	0.6148	0.5412	0.5829
SSIM (BCPD)	0.6071	0.6412	0.6210	0.6392	0.5910	0.6199
SSIM (Ours)	**0.6136**	**0.6458**	**0.6689**	**0.6698**	**0.6249**	**0.6446**

The best results in each category are in bold.

{0.002, 0.004, 0.006}). During testing, we compare FB-Warp and BCPD with noise in source data for each algorithm. Figure 17 shows the results before registration, and Figure 18 shows the results of
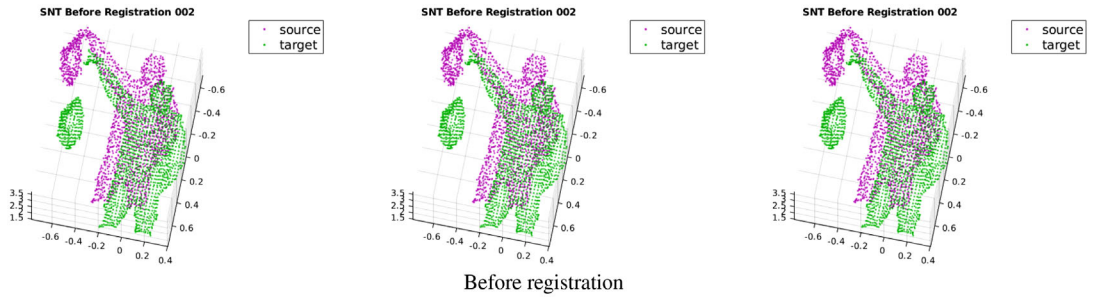
Before registration

Figure 17: *The source point set with noise and target point set without noise (SNT) are for different α, and figures from left to right are 0.002, 0.004 and 0.006.*
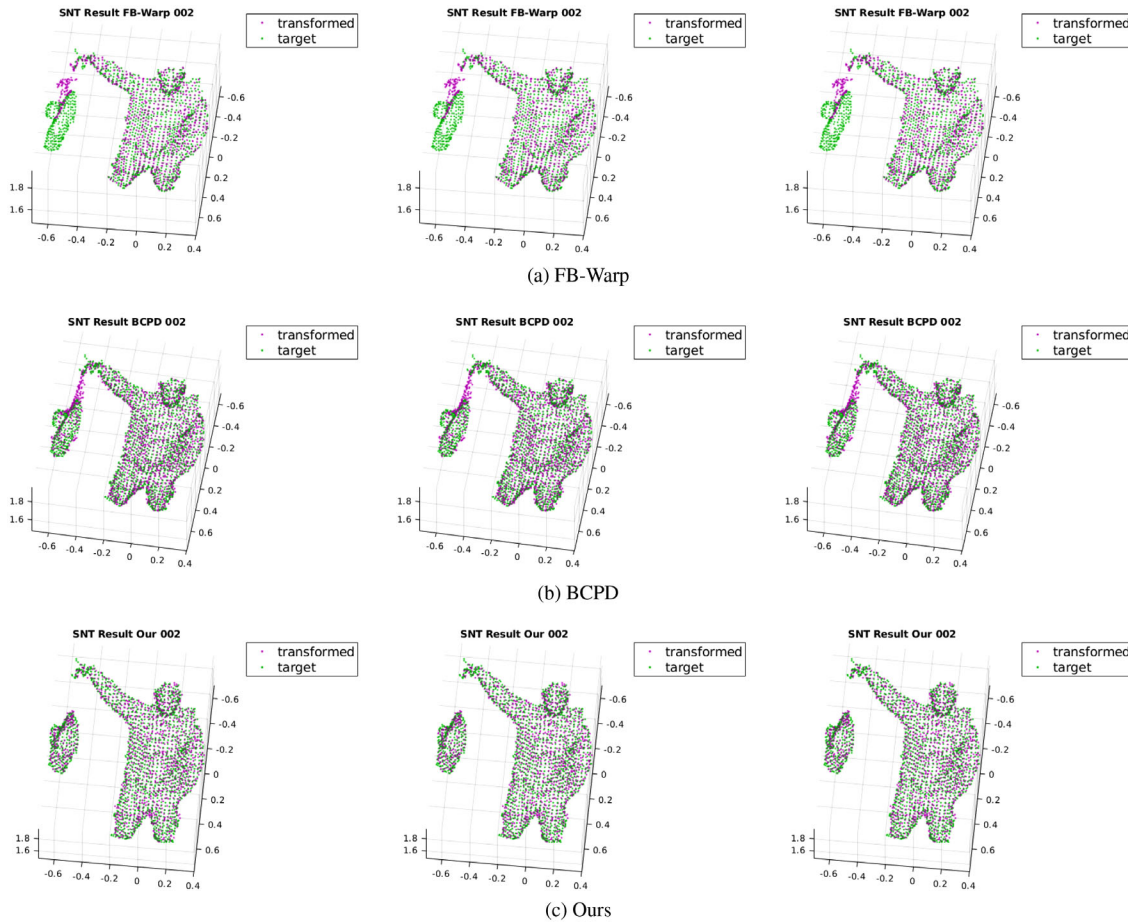

(a) FB-Warp

(b) BCPD

(c) Ours

Figure 18: *The results of registration on Hat data with noise: (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*

registration. Our method results better than FB-Warp and BCPD when α is 0.002 and 0.004. FB-Warp performs the worst, with many inaccurate points on the hat. However, when the α is 0.006, there are many wrong points between hand and hat for all methods except for Figure 20c STN result (source point set is original and target point set is with Gaussian noise).

In addition, we sample noise in the target point cloud (Figure 19), where the mean and the standard deviation are $mean(target\,points)$ and $\alpha * std(target\,points)$ ($\alpha \in \{0.002, 0.004, 0.006\}$). Figure 20 shows the results of registration. At last, the source and target points are sampled Gaussian noise (Figure 21). Figure 22 shows the results of registration. We get similar registration results, and our method
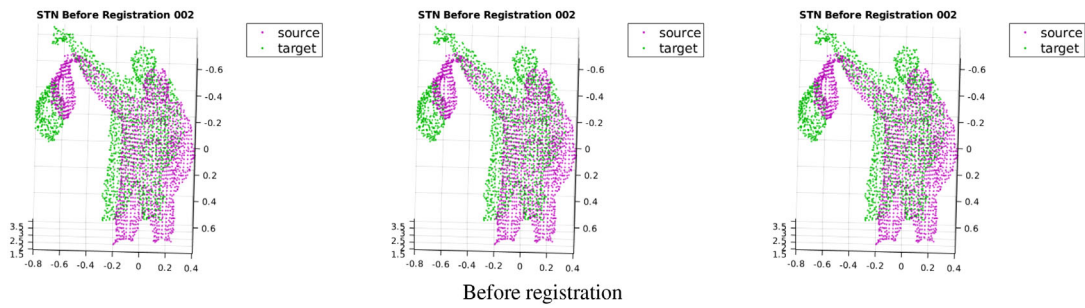
B. Bao & H. Fu / EG Author Guidelines

Before registration

Figure 19: *The source point set without noise and target point set with noise (STN) are for different α, and figures from left to right are 0.002, 0.004 and 0.006.*

Table 4: *Registration time (s).*

	Alex	Boxing	Hat	Bunny	Pillow	Overall
FB-Warp-C	50.2430	**15.8930**	53.7950	36.7800	76.4530	46.6328
FB-Warp-L	45.6756	**12.8875**	42.8859	25.5088	53.2297	36.0975
Ours-C	**13.6605**	18.3537	**21.6762**	**14.1161**	**28.3915**	19.2496
Ours-L	**10.5121**	15.0353	**12.2921**	**12.1661**	**9.8807**	**11.9773**
Partition-C	1.5958	4.8619	1.1519	0.0761	0.8829	1.7137
Partition-L	0.1990	4.0456	1.1071	1.1031	1.0238	1.4957
Registration-C	12.0647	13.4918	20.5243	14.0400	27.5086	17.5259
Registration-L	10.3131	10.9897	11.0590	11.1890	8.8569	10.4815

The best results in each category are in bold.

based on Break and Splice is robust to Gaussian noise to some degree. At the same time, we also compute the RMSE to evaluate different methods, and the results show in Table 5. Our method can lower errors in different situations.

4.4. Discussion

Our method works well, as expected, in dealing with separations and connections in dynamic scenes for point set registrations. Especially in the separation event, our method achieves excellent results. Although the BCPD handles the connection event (Figure 16b), it fails to register the target point set (Figure 15f). In addition, our *Break and Splice* framework achieves lower errors and fast computing time. In addition, there are no available public data sets with various viewpoints and special topological changes. Thus, we experiment with significant view changes. As shown in the following (Figures 23a and 23c are images, Figures 23b and 23d are point sets),

we have acquired two-point clouds by moving the camera about 45°from left to right as a case of significant view change. We have applied our method to this case, as shown in the result of labelled source points in Figure 23e. Our method still works for large view changes. This is because we always find a part of the source and target points set under a common coordinate system after merging the two-point clouds by *Cluster*. Based on this part, we can find one of the labels by *Refine*, and the rest of the point cloud repeats this process (Cluster and Refine) until there is only one label in the merging point cloud. Therefore, our method will not be a failure without the initial alignment.

Meanwhile, our refinement is sensitive to the parameter γ. If the value of the parameter γ is not desirable, the source point will not be refined, which will cause fewer source points to be matched with the target points at the final registration stage. However, BCPD, as an advanced non-rigid registration method, can handle the registration with an inconsistent number of point clouds. The unsatisfied result is that the dense and sparse distribution of the transformed point cloud is different from that of the target point cloud.

5. Conclusion

We present a novel non-rigid point cloud registration framework that handles separation and connection topology changes. The *Break and Splice* framework allows clustering and refinement of point sets to overcome distribution irregularities of the point sets, which can improve the accuracy of non-rigid registration efficiently. Experiment results have shown that our method aligns two-point sets with these topology changes more effectively than the state-of-the-art approaches.
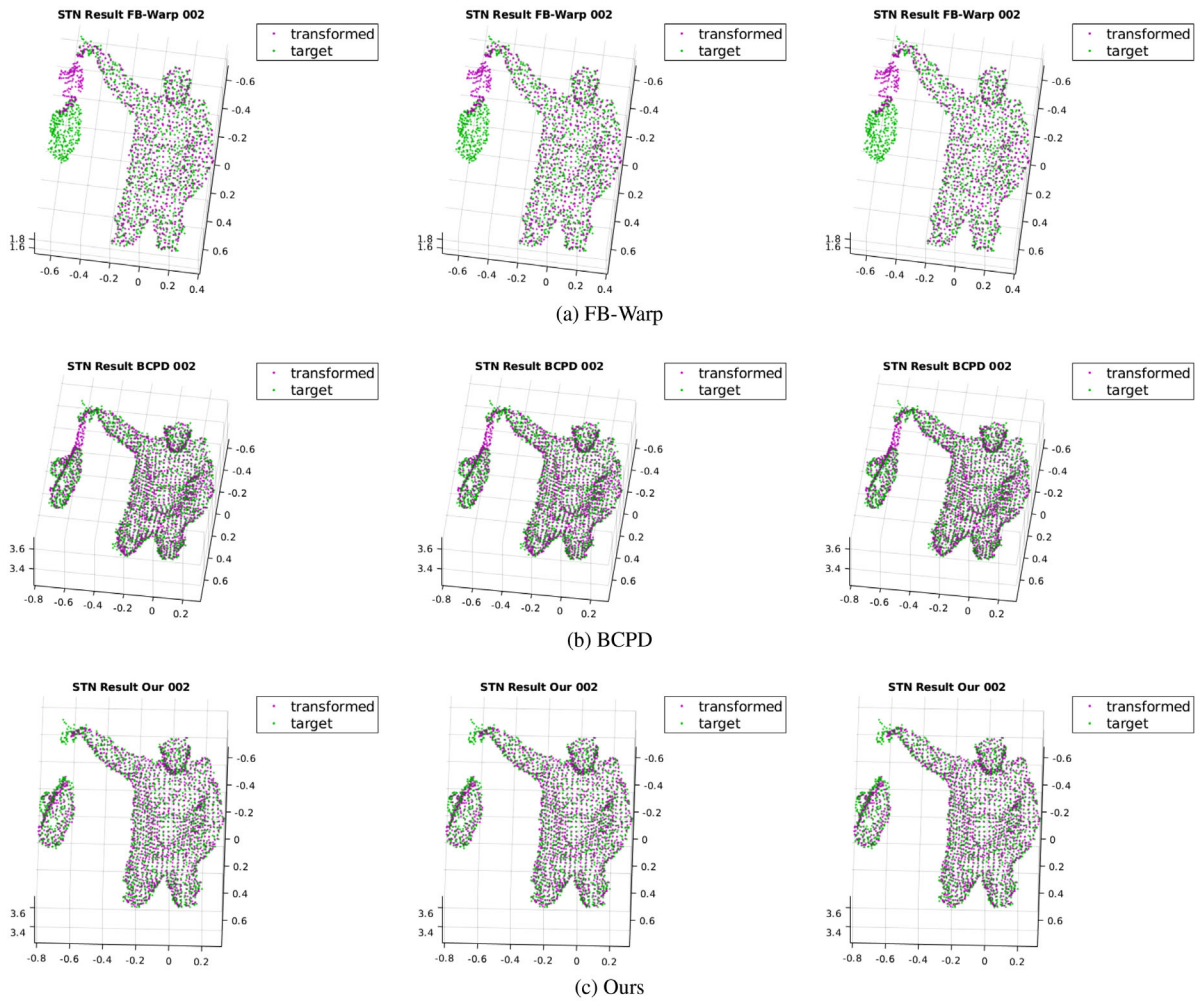
Table 5: *Registration error with Gaussian noise.*

	SNT			STN			SNTN		
	FB-Warp	BCPD	Ours	FB-Warp	BCPD	Ours	FB-Warp	BCPD	Ours
0.002	0.1543	0.1358	**0.0267**	0.1546	0.1229	**0.1344**	0.1296	0.1261	**0.0274**
0.004	0.1395	0.1224	**0.0325**	0.1470	0.1268	**0.1325**	0.1334	0.1241	**0.0339**
0.006	0.1660	0.1292	**0.0448**	0.1494	0.1228	**0.1202**	0.1703	0.1269	**0.0484**

The best results in each category are in bold.

Figure 20: *The results of registration on Hat data with noise: (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*
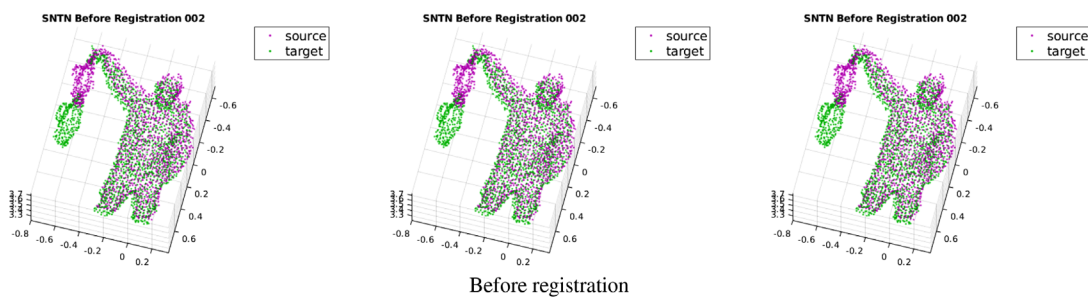

Figure 21: *The source point set with noise and target point set with noise (SNTN) are for different α, and figures from left to right are 0.002, 0.004 and 0.006.*

Currently, the framework does not take into account RGB information. Thus, no texture information is included in the results. In future work, we will focus on developing a method using RGB information to obtain rich textures. Another issue is that our approach requires parts of the two-point clouds to be found in a common co-ordinate system, which must include the parts of the source point set and target point set simultaneously. If this condition is not met, for example, in the case of large deformation, some large camera movements may cause a failure on the label onto *Y* in Figure 2. Especially our method cannot handle more clusters or self-occlusion
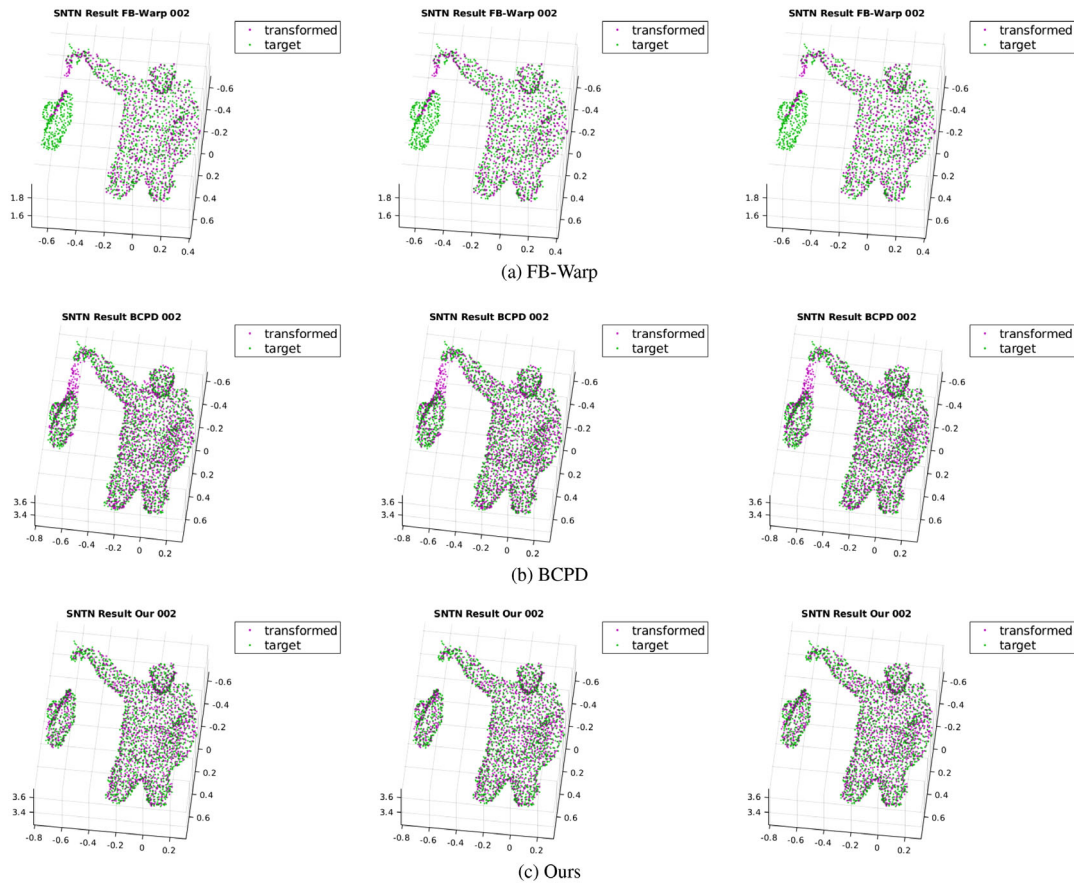
B. Bao & H. Fu / EG Author Guidelines

(a) FB-Warp

(b) BCPD

(c) Ours

Figure 22: *The results of registration on Hat data with noise: (a) uses the method of FB-Warp, (b) uses the method of BCPD without* Break and Splice *and (c) is our algorithm.*
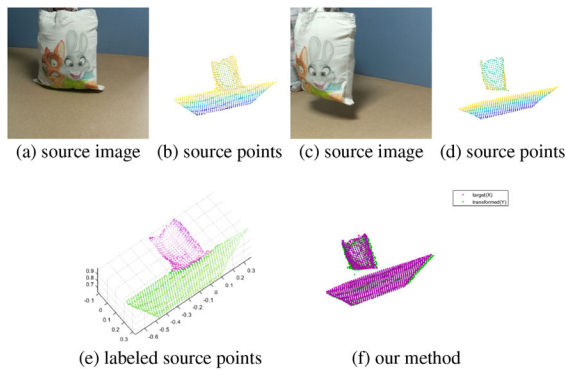

(a) source image   (b) source points   (c) source image   (d) source points

(e) labeled source points   (f) our method

Figure 23: *The result of registration about the large view changes: (a) and (c) are colour images. (b) and (d) are their corresponding point sets. The second row shows the results of assigning labels in source points (e) and registration from source points to target points (f).*

data due to getting inaccurate labels, which are based on 2D boundary extraction. For example, Figure 24 shows a dragon behind the bunny. When the bunny is separated from the table, it is difficult to
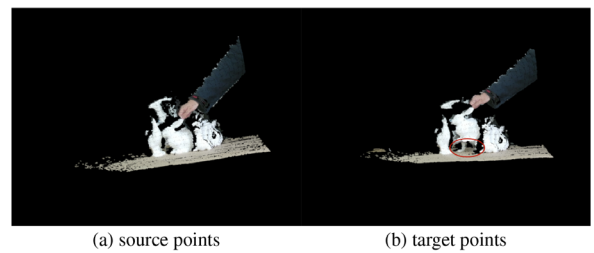

(a) source points   (b) target points

Figure 24: *An example for the cluster. (a) is a source point set, and (b) is a target point set that a bunny is separated from the table.*

find the boundary of the bunny since the bunny and the leg of the dragon will be regarded as an object. In the future, we will explore a new framework to solve the large changes of the camera with the deformation of the object and focus on the extraction of boundaries for clusters or self-occlusion data.

Acknowledgements

References

[ACP03] Allen B., Curless B., Popović Z.: The space of human body shapes: reconstruction and parameterization from range scans. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 587–594.

[AE18] Alexiou E., Ebrahimi T.: Point cloud quality assessment metric based on angular similarity. In *Proceedings of the 2018 IEEE International Conference on Multimedia and Expo (ICME)* (2018), IEEE, pp. 1–6.

[AE20] Alexiou E., Ebrahimi T.: Towards a point cloud structural similarity metric. In *Proceedings of the 2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (2020), IEEE, pp. 1–6.

[AHB87] Arun K. S., Huang T. S., Blostein S. D.: Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence 5* (1987), 698–700.

[ARV07] Amberg B., Romdhani S., Vetter T.: Optimal step nonrigid icp algorithms for surface registration. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (2007), IEEE, pp. 1–8.

[Awr16] Awrangjeb M.: Using point cloud data to identify, trace, and regularize the outlines of buildings. *International Journal of Remote Sensing 37*, 3 (2016), 551–579.

[Ben75] Bentley J. L.: Multidimensional binary search trees used for associative searching. *Communications of the ACM 18*, 9 (1975), 509–517.

[BYG17] Bai L., Yang X., Gao H.: Nonrigid point set registration by preserving local connectivity. *IEEE Transactions on Cybernetics 48*, 3 (2017), 826–835.

[CR03] Chui H., Rangarajan A.: A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding 89*, 2-3 (2003), 114–141.

[CSK05] Chetverikov D., Stepanov D., Krsek P.: Robust Euclidean alignment of 3D point sets: the trimmed iterative closest point algorithm. *Image and Vision Computing 23*, 3 (2005), 299–309.

[CTJ*18] Chen L., Tang W., John N. W., Wan T. R., Zhang J. J.: Slam-based dense surface reconstruction in monocular minimally invasive surgery and its application to augmented reality. *Computer Methods and Programs in Biomedicine 158* (2018), 135–146.

[DKD*16] Dou M., Khamis S., Degtyarev Y., Davidson P., Fanello S. R., Kowdle A., Escolano S. O., Rhemann C., Kim D., Taylor J., Kohli P., Tankovich V., Izadi S.: Fusion4D: Real-time performance capture of challenging scenes. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 1–13.

[DWB06] Durrant-Whyte H., Bailey T.: Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine 13*, 2 (2006), 99–110.

[Fer73] Ferguson T. S.: A Bayesian analysis of some nonparametric problems. *The Annals of Statistics 1*, 2 (1973), 209–230.

[GTRS16] Golyanik V., Taetz B., Reis G., Stricker D.: Extended coherent point drift algorithm with correspondence priors and optimal subsampling. In *Proceedings of the 2016 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2016), IEEE, pp. 1–9.

[GXY*17] Guo K., Xu F., Yu T., Liu X., Dai Q., Liu Y.: Real-time geometry, albedo, and motion reconstruction using a single RGB-D camera. *ACM Transactions on Graphics (ToG) 36*, 4 (2017), 1.

[HAWG08] Huang Q.-X., Adams B., Wicke M., Guibas L. J.: Non-rigid registration under isometric deformations. *Computer Graphics Forum 27* (2008), 1449–1457.

[Hir21] Hirose O.: A Bayesian formulation of coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence 43*, 7 (2021), 2269–2286. https://doi.org/10.1109/TPAMI.2020.2971687

[IZN*16] Innmann M., Zollhöfer M., Nießner M., Theobalt C., Stamminger M.: VolumeDeform: Real-time volumetric non-rigid reconstruction. In *Proceedings of the European Conference on Computer Vision* (2016), Springer, pp. 362–379.

[LG20] Li C., Guo X.: Topology-change-aware volumetric fusion for dynamic scene reconstruction. In *Proceedings of the European Conference on Computer Vision* (2020), Springer, pp. 258–274.

[MFHM19] Meguelati K., Fontez B., Hilgert N., Masseglia F.: Dirichlet process mixture models made scalable and effective by means of massive distribution. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (2019), pp. 502–509.

[MQZ*15] Ma J., Qiu W., Zhao J., Ma Y., Yuille A. L., Tu Z.: Robust $l\_{2}e$ estimation of transformation for non-rigid registration. *IEEE Transactions on Signal Processing 63*, 5 (2015), 1115–1129.

[MS10] Myronenko A., Song X.: Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 12 (2010), 2262–2275.

[MSCP*07] Myronenko A., Song X., Carreira-Perpinán M. A.: Non-rigid point set registration: Coherent point drift. *Advances in Neural Information Processing Systems 19* (2007), 1009.

[Nea00] Neal R. M.: Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics 9*, 2 (2000), 249–265.

[NFS15] Newcombe R. A., Fox D., Seitz S. M.: DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 343–352.

[SBB17] Shah S. A. A., Bennamoun M., Boussaid F.: Keypoints-based surface representation for 3D modeling and 3D object recognition. *Pattern Recognition 64* (2017), 29–38.

[SBCI17] Slavcheva M., Baust M., Cremers D., Ilic S.: Killing-Fusion: Non-rigid 3D reconstruction without correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 1386–1395.

[SBI18] Slavcheva M., Baust M., Ilic S.: SobolevFusion: 3D reconstruction of scenes undergoing free non-rigid motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2646–2655.

[TCL*12] Tam G. K., Cheng Z.-Q., Lai Y.-K., Langbein F. C., Liu Y., Marshall D., Martin R. R., Sun X.-F., Rosin P. L.: Registration of 3D point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics 19*, 7 (2012), 1199–1217.

[WS01] Williams C., Seeger M.: Using the Nyström method to speed up kernel machines. In *Proceedings of the 14th Annual Conference on Neural Information Processing Systems* (2001), pp. 682–688.

[WSMG*16] Whelan T., Salas-Moreno R. F., Glocker B., Davison A. J., Leutenegger S.: Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research 35*, 14 (2016), 1697–1716.

[YG89] Yuille A. L., Grzywacz N. M.: A mathematical analysis of the motion coherence theory. *International Journal of Computer Vision 3*, 2 (1989), 155–175.

[YOF15] Yang Y., Ong S. H., Foong K. W. C.: A robust global and local mixture distance based non-rigid point set registration. *Pattern Recognition 48*, 1 (2015), 156–173.

[ZFA21] Zampogiannis K., Fermüller C., Aloimonos Y.: Topology-aware non-rigid point cloud registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence 43*, 3 (2021), 1056–1069. https://doi.org/10.1109/TPAMI.2019.2940655.

[ZSG*18] Zollhöfer M., Stotko P., Görlitz A., Theobalt C., Nießner M., Klein R., Kolb A.: State of the art on 3D reconstruction with RGB-D cameras. *Computer Graphics Forum 37* (2018), 625–652.

Appendix

Suppose $\mathbf{P}$ is a mixture of K Gaussian distributions (K is unknown). For simplicity, we note $\mathbf{p}_i$ as the ith point in $\mathbf{P}$. $\mathbf{c} = \{c_1, \dots c_{M+N}\}(c_i \in \{1, \dots, K\})$ is the indicator variables, and $c_i = k$ indicates that point p_i is generated from the k_{th} Gaussian distribu-

tion. π_k is defined to represent the weight of the kth Gaussian component, where $\pi_k \geq 0$, $k = \{1, \dots K\}$, and $\sum_{k=1}^{K} \pi_k = 1$.

The Gaussian mixture model with K components may be written as

$$p(\mathbf{p}_i|\theta_1, \dots, \theta_K) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{p}_i|\boldsymbol{\mu}_k, \mathbf{S}_k) \tag{A.1}$$

where $\theta_k = \{\boldsymbol{\mu}_k, \mathbf{S}_k, \pi_k\}$ is the set of parameters for component k. $\boldsymbol{\mu}_k$ is the mean vector for kth Gaussian component, and $\mathbf{S}_k$ is its precision (inverse covariance matrix). We set the prior joint distribution on the component parameters as *normal-Wishart* distribution.

The conditional posterior class probabilities derived by the Dirichlet Process Gaussian Mixture Model (DPGMM) are

for the kth component with $n_{-i,k} > 0$:

$$p(c_i = k|\mathbf{c}_{-i}, \boldsymbol{\mu}_k, \mathbf{S}_k, \alpha)$$
$$\propto \frac{n_{-i,k}}{M + N - 1 + \alpha} \mathcal{N}(\mathbf{p}_i|\boldsymbol{\mu}_k, \mathbf{S}_k) \tag{A.2}$$

for new Gaussian component:

$$p(c_i \neq c_{i'} \text{ for all } i \neq i'|\mathbf{c}_{-i}, \boldsymbol{\xi}, \rho, \beta, \mathbf{W}, \alpha)$$
$$\propto \frac{\alpha}{M + N - 1 + \alpha}$$
$$\times \int p(\mathbf{p}_i|\boldsymbol{\mu}, \mathbf{S}) p(\boldsymbol{\mu}, \mathbf{S}|\boldsymbol{\xi}, \rho, \beta, \mathbf{W}) \, \mathrm{d}\boldsymbol{\mu} \, \mathrm{d}\mathbf{S}$$
$$\propto \frac{\alpha}{M + N - 1 + \alpha} t_{\beta_n - D + 1} \left(\boldsymbol{\xi}_*, \frac{\mathbf{W}_*(\rho_n + 1)}{\rho_n(\beta_n - D + 1)} \right) \tag{A.3}$$

where α is the concentration parameter of the Dirichlet Process, $\alpha > 0$. $\boldsymbol{\xi} \in \mathbb{R}^D$, ρ, β, and $\mathbf{W} \in \mathbb{R}^{D \times D}$ are hyperparameters common to all mixture components. n_k is the occupation number, which indicates the number of points assigned to the k_{th} Gaussian component. $-i$ indicates all indices except for i, and $n_{-i,k}$ is the number of points, excluding $\mathbf{p}_i$, that are associated with the k_{th} Gaussian component. t is the *Student's t-distribution* with $\beta_n - D + 1$ degrees of freedom.

$$\beta_n = \beta + n_k \tag{A.4}$$

$$\rho_n = \rho + n_k \tag{A.5}$$

$$\boldsymbol{\xi}_* = \frac{\rho \boldsymbol{\xi} + \sum_{i:c_i=k} \mathbf{p}_i}{\rho + n_k} \tag{A.6}$$

$$\mathbf{W}_* = \mathbf{W} + \rho \boldsymbol{\xi} \boldsymbol{\xi}^T + \sum_{i:c_i=k} \mathbf{p}_i \mathbf{p}_i^T - (\rho + n_k) \boldsymbol{\xi}_* \boldsymbol{\xi}_*^T \tag{A.7}$$

Collapsed Gibbs Sampling method is used for the inference on the model above. For a detailed sampling process, please refer to Neal [Nea00] for a detailed sampling process. The DPGMM model can be expressed as follows. The maximum conditional posterior class probability determines the clustering to which each point belongs.

$$\mathbf{p}_i|c_i \sim \mathcal{N}(\boldsymbol{\mu}_{c_i}, \mathbf{S}_{c_i}) \tag{A.8}$$

$$c_i|\boldsymbol{\pi} \sim Please) \tag{A.9}$$

$$\boldsymbol{\pi}|\alpha \sim Dir\left(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\right) \tag{A.10}$$

$$(\boldsymbol{\mu}_k|\mathbf{S}_k, \boldsymbol{\xi}, \rho) \sim \mathcal{N}\left(\boldsymbol{\xi}, (\rho\mathbf{S}_k)^{-1}\right) \tag{A.11}$$

$$(\mathbf{S}_k|\beta, \mathbf{W}) \sim \mathcal{W}(\beta, \mathbf{W}) \tag{A.12}$$

$$(\boldsymbol{\mu}_k, \mathbf{S}_k) \sim \mathcal{NW}(\boldsymbol{\xi}, \rho, \beta, \mathbf{W}) \tag{A.13}$$

$$(n_1, \ldots n_K) \sim Multi(\pi_1, \ldots, \pi_K) \tag{A.14}$$

where α is the concentration parameter of the Dirichlet process, $\alpha > 0$. $\boldsymbol{\xi} \in \mathbb{R}^D$, ρ, β and $\mathbf{W} \in \mathbb{R}^{D \times D}$ are hyperparameters common to all mixture components. n_k is the occupation number, which indicates the number of points assigned to the k_{th} Gaussian component. *Cat* is the *Categorical* distribution. *Dir* and *Multi* represent *Dirichlet* distribution and *Multinomial* distribution, respectively.