A Bayesian Formulation of Coherent Point Drift

Osamu Hirose, *Member, IEEE*

Abstract—Coherent point drift is a well-known algorithm for solving point set registration problems, i.e., finding corresponding points between shapes represented as point sets. Despite its advantages over other state-of-the-art algorithms, theoretical and practical issues remain. Among theoretical issues, (1) it is unknown whether the algorithm always converges, and (2) the meaning of the parameters concerning motion coherence is unclear. Among practical issues, (3) the algorithm is relatively sensitive to target shape rotation, and (4) acceleration of the algorithm is restricted to the use of the Gaussian kernel. To overcome these issues and provide a different and more general perspective to the algorithm, we formulate coherent point drift in a Bayesian setting. The formulation brings the following consequences and advances to the field: convergence of the algorithm is guaranteed by variational Bayesian inference; the definition of motion coherence as a prior distribution provides a basis for interpretation of the parameters; rigid and non-rigid registration can be performed in a single algorithm, enhancing robustness against target rotation. We also propose an acceleration scheme for the algorithm that can be applied to non-Gaussian kernels and that provides greater efficiency than coherent point drift.

Index Terms—Non-rigid point set registration, coherent point drift, variational Bayesian inference, motion coherence, fast computation

1 Introduction

The goal of point set registration is to find pairs of corresponding points between shapes represented as point sets. Algorithms for finding shape correspondences have been actively studied in computer vision and computer graphics. Why do we need to find corresponding points between shapes? This is because shapes with correspondence information become a foundation to organize, reconstruct, and synthesize a broader class of shapes. We begin with a summary concerning some of the active research fields in which point set registration arises.

Shape Reconstruction. A goal of computer vision and graphics is to reconstruct the entire 3D surface of an object from range images, i.e., 2D images with depth information, captured at different camera positions. To this end, range images are typically converted into point sets, also called point clouds. A naïve conversion of range images embeds the corresponding point clouds into local coordinate systems. If every point cloud partially overlaps with some of the others, point set registration finds overlaps of the point clouds and aligns them in a global coordinate system.

Shape Retrieval. As a result of advances in computer graphics, a vast number of 3D shapes have been generated. One aim is to rapidly retrieve 3D shapes that resemble a query object from a database. One approach to performing this task is to define a measure of similarity between shapes and return a shape having a significant similarity with a query object. Here, we refer to shapes with and without correspondence information as structured and unstructured shapes, respectively. Defining an adequate measure of similarity is, however, a non-trivial task because typical 3D shapes are unstructured. Additionally, each shape model might be embedded in a local coordinate system. Let us assume that shapes are structured and located in a global coordinate system. Then, a naïve similarity measure based on the Euclidean distance, for example, can be a reasonable, computationally efficient similarity measure. Point set registration converts unstructured shapes into structured ones.

Shape Model Learning. If we have a class of structured shapes such as human faces, we can learn a shape model that summarizes shape variations in the class using a statistical learning technique, e.g., principal component analysis (PCA). This is because each element in a structured shape vector can be interpreted as a statistical variable with a specific meaning, e.g., the tip of a nose. Point set registration converts points in unstructured shapes into statistical variables. The shape model after the learning can be used for shape blending and shape completion, for example.

a) *Shape Blending.* If 3D shapes can be automatically synthesized, the efforts of graphic designers can be reduced significantly. A PCA-based shape model, for example, allows us to generate an infinite number of 3D shapes endowed with the characteristics of training shapes. Shape generation based on structured shapes is called shape blending.

b) *Shape Completion.* A 3D surface generated by the scan of a real object typically contains missing regions due to occlusion. The missing regions can be recovered naturally by fitting a PCA-based watertight shape model, i.e., a shape model without defective holes or gaps, to the scanned surface. The interpolation of missing regions in a scan is called shape completion.

Types of Registration Algorithms. Various registration algorithms have been developed to solve shape correspondence problems. Typically, registration algorithms are categorized as either rigid or non-rigid according to the transformation models that deform shapes and align them. Rigid registration algorithms find a map that preserves the distance between

• *The author is with the Institute of Science and Engineering, Kanazawa University, Kakuma, Kanazawa, Ishikawa 920-1192, Japan. E-mail: hirose@se.kanazawa-u.ac.jp.*

Manuscript received 1 Apr. 2019; revised 31 Jan. 2020; accepted 2 Feb. 2020. Date of publication 6 Feb. 2020; date of current version 3 June 2021. (Corresponding author: Osamu Hirose.) Recommended for acceptance by I. Shimshoni. Digital Object Identifier no. 10.1109/TPAMI.2020.2971687

every pair of points, i.e., a map defined as rotation and translation. The reconstruction of an entire 3D surface from partial scans typically requires rigid registration.

In contrast, non-rigid registration algorithms find a map that does not necessarily preserve the distance between points. Non-rigid registration algorithms can be divided into two groups involving either non-linear maps or linear maps, e.g., similarity transformations and affine transformations. We note that non-rigid registration algorithms based on similarity transformations are sometimes categorized into rigid registration algorithms owing to their simplicity. Shape model learning and shape retrieval typically involve non-rigid registration with a non-linear map.

Coherent Point Drift. Coherent point drift (CPD) [1], [2] is a collection of three different registration algorithms based on a Gaussian mixture model (GMM). Among them, the best-known algorithm is a non-rigid algorithm with a non-linear map, which is based on the motion coherence theory [3]. We hereafter refer to it as CPD unless otherwise specified. CPD is considered a state-of-the-art non-rigid registration algorithm because of its registration performance and scalability to large point sets [4]. A primary reason for its stability is its estimation of shape correspondence in a soft-matching manner; point-to-point correspondences are not assumed to be one-to-one. Additionally, the sequential update of the residual variance in the GMM automatically shrinks the set of candidate points corresponding to each source point during the optimization [5].

Our Contribution. We formulate CPD in a Bayesian setting; we replace the motion coherence theory, i.e., the theoretical basis of CPD, with Bayesian inference; we introduce motion coherence using a prior distribution of displacement vectors. We also derive an algorithm that overcomes several issues of CPD on the basis of variational Bayesian inference (VBI). The algorithm is endowed with the following characteristics:

- Convergence of the algorithm is guaranteed.
- The parameters regarding motion coherence can be interpreted intuitively.
- The algorithm combines the rigid CPD and the non-rigid CPD.
- The algorithm can be accelerated with non-Gaussian kernel functions.

In numerical studies, we evaluate the registration performance of the algorithm by comparing it with the state-of-the-art algorithms. The remainder of the paper is organized as follows: In Section 2, we review point set registration methods related to our study. In Section 3, we follow the derivation of the non-rigid CPD algorithm and summarize several issues to be addressed. In Section 4, we present our methodological contributions with a summary of VBI. In Section 5, we evaluate the registration performance of our algorithm. In Section 6, we conclude this study. Appendices and Supplementary Video 1 can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.2971687. The implementation of the algorithm is available at the author's GitHub repository: https://github.com/ohirose/bcpd.

2 Related Work

In this section, we review registration algorithms related to our work. As described in the previous section, point set registration algorithms are typically categorized into rigid and non-rigid registration algorithms. Among these algorithms, rigid registration algorithms have been intensively studied [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17]. Rigid registration can also be performed through 3D keypoint detection, local surface feature description, and surface matching. The algorithms related to this approach are broadly covered in a survey paper [18].

Various transformation models for non-rigid point set registration have been proposed. A majority of the non-rigid transformations are based on thin plate spline functions [19], [20], [21], [22], [23], Gaussian functions [1], [2], [24], [25], [26], [27], and local affine transformations [28], [29], [30], [31], [32]. These methods are classified according to the formulation of the point set registration problem: the minimization of a weighted squared loss function with penalty terms [20], [23], [28], [29], [30], [31], [32] and the likelihood maximization of Gaussian mixture models [1], [2], [9], [19], [21], [22], [24], [25], [26], [27]. Registration methods with such different formulations are closely related; however, there is a clear difference in the estimation of the residual variance during optimization [9].

The handling of outliers, i.e., points that are irrelevant to the actual object geometry, is a major issue of non-rigid registration. Various approaches have been proposed to deal with outliers, such as statistical analysis for distances between corresponding points [9], [33], [34], soft assignments [20], [35], trimming point sets through iterative random sampling [36], kernel correlation [37], explicit probabilistic modeling of outliers [1], [2], [25], [26], [27], and the use of robust estimators such as the L_2E estimator [21] and a scaled Geman–McClure estimator [38], [39].

Non-rigid registration techniques typically assume the smoothness of a displacement field, i.e., motion coherence, in which displacement vectors to define a shape deformation gradually become parallel as the distance between two points decreases. The smoothness is typically introduced by imposing local rigidity on the surface of a deformed shape [40], [41] and the penalty on a non-smooth displacement field [1], [2], [19], [20], [21], [22], [23], [25], [26], [27]. By the assumption of the smooth displacement field, non-rigid registration algorithms seek to find transformations with sufficient global flexibility while preserving the local topology of a point set. Besides the coherent moves based on the proximity, several types of coherent moves have been utilized to register shapes involving more specific deformations such as human face [42], [43], human hand shape and pose [44], and human body pose [28], [38], [45], [46], [47], [48], [49], [50], [51], [52].

3 Preliminary

In this section, we follow the derivation of the CPD algorithm [2] to clarify the formulation difference between CPD and our Bayesian variant. CPD is derived by first defining a GMM, which implicitly encodes the unknown correspondences between points, and then minimizing the negative log-likelihood function using the EM algorithm.

Notation 1. The goal of point set registration is to find the map $\mathcal{T}$ that transforms the geometric shape represented as a point set $Y = \{y_1, \ldots, y_M\}$ so that the transformed shape

matches the target shape represented as the other point set $X = \{x_1, \ldots, x_N\}$. Here, we refer to a point set to be deformed as a source point set, and refer to the other point set that remains fixed as a target point set. The set of $\mathcal{T}(y) - y$ for any $y \in \mathbb{R}^D$ is called a displacement field, as it defines the displacement of each point in Y. We denote the displacement field by $v(y)$ or v. Additionally, we refer to any point that is irrelevant to the actual object geometry as an outlier. Throughout this paper, we use the following notation:

- N, M – the numbers of points in target and source point sets, respectively.
- D – the dimensionality of the space in which a point set is embedded.
- $x_n = (x_{n1}, \ldots, x_{nD})^T \in \mathbb{R}^D$ – the nth point in a target point set $X = \{x_1, \ldots, x_N\}$.
- $y_m = (y_{m1}, \ldots, y_{mD})^T \in \mathbb{R}^D$ – the mth point in a source point set $Y = \{y_1, \ldots, y_M\}$.
- $\mathrm{Tr}(\cdot)$, $|\cdot|$, $\mathrm{d}(\cdot)$ – the trace of a matrix, the determinant of a matrix, and the operation that converts a vector into its corresponding diagonal matrix, respectively.
- I_D, I_M – the identity matrices of sizes D and M.
- 1_D, 1_M, 1_N – the vectors of all 1s of sizes D, M, and N.

We also use the symbols of the point sets X and Y as the matrix symbols, which are defined as $X = (x_1, \ldots, x_N)^T$ and $Y = (y_1, \ldots, y_M)^T$. The notation used throughout this paper is summarized in Appendix E, available online.

3.1 Gaussian Mixture Model for Point Set Registration

We begin with the definition of a GMM, which is the basis of CPD. In the framework of CPD, a set of points representing a deformed shape, i.e., $\mathcal{T}(Y)$, is assumed to be centroids of a GMM, and a target point x_n is a data point generated by the GMM. The mth component distribution of the GMM is defined as follows:

$$p(x_n|m;\theta_0) = |2\pi\sigma^2 I_D|^{-1/2} \exp\Big\{-\frac{1}{2\sigma^2}||x_n - \mathcal{T}(y_m)||^2\Big\},$$

where $\mathcal{T}(y_m) = y_m + v(y_m)$ is a transformation model to move source points, and $\theta_0 = (v, \sigma^2)$ is the set of parameters. With the outlier distribution $p_{\mathrm{out}}(x_n) = 1/N$, the GMM is defined as follows:

$$p(x_n;\theta_0) = \omega p_{\mathrm{out}}(x_n) + (1-\omega)\sum_{m=1}^{M}\frac{1}{M}p(x_n|m;\theta_0),$$

where ω is the prior probability that x_n is an outlier. By the construction of the probabilistic model, point set registration is replaced with the estimation of θ_0 based on minimization of the negative likelihood regarding the GMM.

3.2 EM Algorithm and the Q-Function

Owing to the lack of a closed-form expression for $\theta_0 = (v, \sigma^2)$ that minimizes the negative likelihood of the GMM, a sequential optimization technique called the EM algorithm is applied to the minimization. The CPD algorithm is obtained as the result of applying the EM algorithm, and it seeks to find a local minimum of the negative likelihood by iterating the following procedure: (1) computation of the matching probability between x_n and y_m, denoted by p_{mn}, and (2) minimization of the objective function, called the Q-function, given p_{mn}. The former and latter are called the E-step and M-step, respectively. Suppose $\theta_0' = (v', \sigma'^2)$ is the alternative set of parameters θ_0. From the theory of the EM algorithm, the Q-function is derived as follows:

$$Q(\theta_0, \theta_0') = \frac{\hat{N}D}{2}\ln\sigma^2 + \frac{1}{2\sigma^2}\sum_{n=1}^{N}\sum_{m=1}^{M}p_{mn}||x_n - (y_m + v(y_m))||^2,$$

where $\hat{N} = \sum_{n=1}^{N}\sum_{m=1}^{M}p_{mn} \leq N$ is the estimated number of current matching points. The posterior matching probability p_{mn} is derived as follows:

$$p_{mn} = \frac{\exp(-\frac{1}{2\sigma'^2}||x_n - (y_m + v'(y_m))||^2)}{c_0 + \sum_{m'=1}^{M}\exp(-\frac{1}{2\sigma'^2}||x_n - (y_{m'} + v'(y_{m'}))||^2)},$$

where $c_0 = \frac{\omega}{1-\omega}\frac{M}{N}|2\pi\sigma'^2 I_D|^{1/2}$. The matching probability p_{mn} is dependent only on θ_0', and therefore the update of p_{mn} and the minimization of the Q-function regarding θ_0 can be alternated. We note that the Q-function is considered as an error function of the shape deformation under the current alignment θ_0'. The E-step and M-step can therefore be interpreted as follows: the E-step aligns point sets X and Y in a soft-matching manner, whereas the M-step finds the shape that is closest to the target shape given the current alignment θ_0'.

3.3 Regularization and the Motion Coherence Theory

If no constraint is imposed on the displacement field $v(y)$, the minimization of the Q-function given θ_0' is ill-posed; for example, the Q-function is obviously minimized if displacement vectors satisfy $x_1 = y_m + v(y_m)$ for all m given a fixed σ^2. A regularization term to avoid trivial solutions is incorporated into the Q-function as follows:

$$Q'(\theta_0, \theta_0') = Q(\theta_0, \theta_0') + \lambda\int_{\mathbb{R}^D}\frac{|\tilde{v}(s)|^2}{\tilde{\mathcal{G}}(s)}ds, \tag{1}$$

where λ is a trade-off parameter, the tilde symbol indicates the Fourier transform, and $\tilde{\mathcal{G}}$ is a positive function that tends to zero as $||s|| \to \infty$. We note that $1/\tilde{\mathcal{G}}$ is a high-pass filter, and therefore the penalization of the displacement field is increased as it becomes rougher. Myronenko *et al.*, the authors of the CPD algorithm, used the Gaussian function $\mathcal{G}_\beta(z) = \exp(-\frac{||z||^2}{2\beta^2})$ as a choice of $\mathcal{G}$. One of the reasons for this choice is that the regularization term is equivalent to that of the motion coherence theory [3], which penalizes non-coherent moves of neighboring points. Therefore, this regularization implies that motion coherence is imposed on the source points to be moved for registration.

3.4 Minimization of the Q-Function

The remaining task to derive the CPD algorithm is to minimize $Q'(\theta_0, \theta_0')$ regarding $\theta_0 = (v, \sigma^2)$ given the current alignment $\theta_0' = (v', \sigma'^2)$. Owing to the nonlinear dependence between v and σ^2, a closed-form expression of their simultaneous updates is not available. Therefore, v and σ^2 are updated separately. To derive the equation for updating v,

we first assume that σ^2 is given. The alternative parameter set θ_0' defines the current soft-matching, and therefore, the minimization of Q' regarding v is interpreted as a function-fitting problem based on the regularization theory [54]. The solution of the problem is guaranteed to be

$$v(y) = \sum_{m=1}^{M} w_m \mathcal{G}(y - y_m) + \varphi(y),$$

where $w_m \in \mathbb{R}^D$ is a coefficient vector, and $\varphi(y)$ is a function that spans the null space of the functional $\int_{\mathbb{R}^D} \frac{|\tilde{v}(s)|^2}{\tilde{\mathcal{G}}(s)} ds$ [55]. Specifically, if the function $\mathcal{G}$ is positive definite, $\varphi(y)$ is guaranteed to be zero. The Gaussian function $\mathcal{G}_\beta$ as a choice of $\mathcal{G}$ guarantees $\varphi(y) = 0$ owing to the positive definiteness of $\mathcal{G}_\beta$. Suppose $\hat{Y} = (\hat{y}_1, \ldots, \hat{y}_M)^T \in \mathbb{R}^{M \times D}$ with $\hat{y}_m = y_m + v'(y_m) \in \mathbb{R}^D$ is the matrix that represents a current deformed shape given θ_0'. By using the general solution to solve the Euler-Lagrange equation related to Eq. (1), the equation for updating $v(y)$ is obtained through the coefficient matrix $W = (w_1, \ldots, w_M)^T \in \mathbb{R}^{M \times D}$ as follows:

$$W = (G + \lambda\sigma^2 \mathrm{d}(P1_N)^{-1})^{-1}(\mathrm{d}(P1_N)^{-1}PX - \hat{Y}),$$

where $P = (p_{mn}) \in [0,1]^{M \times N}$ and $G = (g_{mm'}^{(\beta)}) \in \mathbb{R}^{M \times M}$ with $g_{mm'}^{(\beta)} = \mathcal{G}_\beta(y_m - y_{m'})$. After the update of the displacement field $v(y)$, the residual variance σ^2 is updated. By equating the corresponding derivative of Q' to zero, the equation for updating σ^2 is obtained as follows:

$$\sigma^2 = \frac{1}{\hat{N}D} \mathrm{Tr}\{X^T \mathrm{d}(P^T 1_M) X - 2\hat{Y}^T PX + \hat{Y}^T \mathrm{d}(P1_N)\hat{Y}\}.$$

The CPD algorithm is designed to perform iterative updates of P, W, and σ^2 under a suitable initialization on the basis of the EM algorithm.

3.5 Several Issues

Despite the advantages of CPD over state-of-the-art algorithms, several issues remain in both theoretical and practical aspects. One theoretical issue of CPD is that it is unknown whether the algorithm always converges, as suggested by Myronenko *et al.* [2]. If we find $\hat{\theta}_0 = (\hat{v}, \hat{\sigma}^2)$ satisfying $Q'(\hat{\theta}_0, \theta_0') \leq Q'(\theta_0', \theta_0')$ at the minimization step, convergence is guaranteed from the theory of the EM algorithm. However, it is unknown whether the updates of v and σ^2 in the M-step satisfy the condition, because the equations for updating them are obtained by the partial minimization of Q' regarding either v or σ^2 given the other. Another theoretical issue of CPD is the difficulty in interpreting tuning parameters λ and β. Both parameters control the degree of motion coherence, i.e., the smoothness of a displacement field, and their difference is unclear. Therefore, the tuning parameters tend to be controlled in a non-intuitive manner. In a practical aspect, CPD is relatively sensitive to the rotation of a target. This is because (1) CPD captures the target's rotation using the non-rigid displacement term $v(y)$, and (2) the rotation of a source shape for registration is penalized by the regularization as well as its non-rigid deformation. Another practical issue is that the acceleration scheme of CPD is limited to the case of the Gaussian function, which will be reviewed in Section 4.6. Through a Bayesian formulation of CPD, we address both theoretical and practical issues.

4 Methods

We propose a Bayesian formulation of CPD and derive an algorithm called Bayesian coherent point drift (BCPD). In Section 4.1, we review VBI. In Section 4.2, we formulate point set registration in a Bayesian setting without the motion coherence theory. In Section 4.3, we derive the BCPD algorithm using VBI, which guarantees convergence of the algorithm. In Section 4.4, we summarize the difference between CPD and BCPD, and we show that CPD is recovered from BCPD. In Section 4.5, we provide a statistical view of motion coherence. In Section 4.6, we present an acceleration scheme of BCPD that can be applied to non-Gaussian kernel functions.

4.1 Variational Bayesian Inference

One important task of Bayesian inference is to estimate a set of unobserved variables, θ, from a set of observations, z. To this end, we need to compute the maximum mode of a posterior distribution $p(\theta|z)$ or the expectation of θ over $p(\theta|z)$, because they are reasonable estimates of θ. However, this task is often hampered by the difficulty in computing the estimates; e.g., the analytic form of the maximum mode is unavailable because of the multimodality of the posterior distribution, or the computational cost of evaluating the expectation is prohibitively large because of the existence of many discrete variables. VBI relaxes the difficulty by approximating $p(\theta|z)$ using an alternative distribution $q(\theta)$ with the property that the maximum mode or the expectation is easily computed. The distribution $q(\theta)$ itself is generally unknown; therefore, finding reasonable estimates of θ is replaced with finding the distribution $q(\theta)$ that approximates $p(\theta|z)$ as much as possible. Typically, VBI is defined as the minimization of the Kullback–Leibler (KL) divergence between $q(\theta)$ and $p(\theta|z)$, as follows:

$$\hat{q}(\theta) = \mathrm{argmin}_q \left\{ -\int q(\theta) \ln \left(\frac{p(\theta|z)}{q(\theta)} \right) d\theta \right\}. \tag{2}$$

The equivalent definition suitable for deriving the general solution is the maximization of the lower bound $L(q)$ for the log model evidence $\ln p(z)$, where $L(q)$ is defined as follows:

$$L(q) = \int q(\theta) \ln \left(\frac{p(z,\theta)}{q(\theta)} \right) d\theta,$$

i.e., the negative KL divergence between $q(\theta)$ and $p(z,\theta)$. If no constraint is imposed on $q(\theta)$, the computations of the expectation and the maximum mode remain intractable because the solution of this optimization problem is $\hat{q}(\theta) = p(\theta|z)$. Here, we suppose that θ can be divided into J groups, i.e., $\theta = (\theta_1, \ldots, \theta_J)$, and we denote the marginal distribution of θ_j with respect to $q(\theta)$ by $q_j(\theta_j)$. To find $q(\theta)$ that relaxes the difficulty in computing the estimates of θ, the posterior distribution $q(\theta)$ is constrained to be the product of its marginals, i.e., $q(\theta) = \prod_{j=1}^{J} q_j(\theta_j)$. This factorization works for splitting the original problem, Eq. (2), into its subproblems. Here, we assume that only q_i is unknown among the factorized distributions $\{q_1, \ldots, q_J\}$. Then, the

maximization of the lower bound $L(q)$ is replaced by the one that focuses on q_i

$$\hat{q}_i(\theta_i) = \mathrm{argmax}_{q_i} \int q_i(\theta_i) \ln \Big(\frac{\exp E_i[\ln p(z,\theta)]}{q_i(\theta_i)} \Big) d\theta_i, \tag{3}$$

where $E_i[\ln p(z,\theta)] = \int \ln p(z,\theta) \prod_{j(\neq i)}^{J} q_j(\theta_j) d\theta_j$. We note that the addition of a constant inside the integral does not change the solution. As the integral term forms a negative KL divergence, the general solution is obtained as follows:

$$\ln \hat{q}_i(\theta_i) = E_i[\ln p(z,\theta)] + const., \tag{4}$$

where the constant term corresponds to the normalization constant of $\hat{q}_i(\theta_i)$. Suppose $\tilde{q}(\theta) = \hat{q}_i(\theta_i) \prod_{j(\neq i)}^{J} q_j(\theta_j)$. Then, the inequality of the lower bounds $L(\tilde{q}) \geq L(q)$ always holds and is tight if and only if $\hat{q}_i(\theta_i) = q_i(\theta_i)$ for any θ_i. This is because the subproblem, Eq. (3), is replaced by the minimization of the KL divergence; the KL divergence $D_{\mathrm{KL}}(p_1||p_2)$ is non-negative, and becomes zero if and only if $p_1(x) = p_2(x)$ for any x [53]. Therefore, a solution for the optimization problem, Eq. (2), is obtained by the following procedure:

1) Initialize q_i for all $i \in \{1, \ldots, J\}$.
2) Cycle the update of q_i with the other q_j fixed for each $i \in \{1, \ldots, J\}$ using Eq. (4).
3) Repeat step 2 until convergence.

Convergence is guaranteed because $L(q)$ is monotonically increasing and bounded because of the inequalities $\ln p(z) \geq L(\hat{q}) \geq L(\tilde{q}) \geq L(q)$.

4.2 Bayesian Formulation of CPD

We formulate point set registration in a Bayesian-inference manner; that is, we first define a probabilistic model that generates a target point set X from a source point set Y with unobserved random variables θ, and then estimate θ given X and Y. In this section, we focus on the construction of the generative model of X. The key difference between our formulation and the original CPD formulation is that we define motion coherence using a prior distribution instead of the regularization term. We suppose that $\mathcal{T}(y_m)$ is the function that deforms the shape represented as Y to match it with the shape represented as X.

Assumptions. As in the definition of Myronenko *et al.*, we assume that a target point set X is generated under the following assumptions:

1) A target point x_n is selected as either a point forming a target shape or an outlier with the outlier probability ω.
2) If a target point x_n is selected as an outlier, x_n is generated from an outlier distribution $p_{\mathrm{out}}(x_n)$.
3) If x_n is a non-outlier, an index $m \in \{1, \ldots, M\}$, indicating that x_n corresponds to y_m, is sampled with a probability α_m, where $\sum_{m=1}^{M} \alpha_m = 1$.
4) Then, x_n is generated from a D-dimensional multivariate normal distribution with a mean vector $\mathcal{T}(y_m)$ and a covariance matrix $\sigma^2 I_D$.
5) A target point set X is generated by an N-times repetition of 1, 2, 3, and 4.

The original formulation of CPD constrains α_m to be $1/M$ for all $m \in \{1, \ldots, M\}$. On the basis of these assumptions, we define the generative model of X.

Notation 2. Here, we define the notation required for the definition of the generative model as follows:

- $x = (x_1^T, \ldots, x_N^T)^T \in \mathbb{R}^{DN}$ – the vector representation of a target point set $X = \{x_1, \ldots, x_N\}$.
- $y = (y_1^T, \ldots, y_M^T)^T \in \mathbb{R}^{DM}$ – the vector representation of a source point set $Y = \{y_1, \ldots, y_M\}$.
- $v = (v_1^T, \ldots, v_M^T)^T \in \mathbb{R}^{DM}$ – the vector representation of displacement vectors that characterize a non-rigid transformation of a source shape.
- $c = (c_1, \ldots, c_N)^T \in \{0, 1\}^N$ – the vector of indicator variables, where c_n takes 1 if the nth target point is a non-outlier, and is 0 otherwise.
- $e = (e_1, \ldots, e_N)^T \in \{1, \ldots, M\}^N$ – the vector of index variables representing that the nth target point corresponds to the mth source point if $e_n = m$.
- $\alpha = (\alpha_1, \ldots, \alpha_M)^T \in [0, 1]^M$ – the vector of probabilities that satisfies $\sum_{m=1}^{M} \alpha_m = 1$, where α_m represents the probability of an event $e_n = m$ for any n.
- $\rho = (s, R, t)$ – the set of random variables that defines the similarity transformation T.
- $\phi(z; \mu, S)$ – the multivariate normal distribution of z with a mean vector μ and a covariance matrix S.

We note that $y \in \mathbb{R}^{DM}$ and $v \in \mathbb{R}^{DM}$ are, hereafter, used as vector symbols that are different from those for defining the displacement field $v(y) : \mathbb{R}^D \to \mathbb{R}^D$ in Section 3.

Transformation Model. We begin with the definition of the non-rigid transformation model $\mathcal{T}$ that deforms a source shape. Unlike in CPD, we define it as a combination of a similarity transformation T and a non-rigid transformation as follows:

$$\mathcal{T}(y_m) = T(y_m + v_m) = sR(y_m + v_m) + t,$$

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

Gaussian Mixture Model. We define a GMM that generates a point in a target point set and plays a role in a likelihood function for solving registration problems. Under assumption 4, we define the distribution of x_n for an event $e_n = m$ as a multivariate normal distribution as follows:

$$\begin{aligned} &\phi(x_n; \mathcal{T}(y_m), \sigma^2 I_D) \\ &\quad = |2\pi\sigma^2 I_D|^{-1/2} \exp\Big\{ -\frac{1}{2\sigma^2} ||x_n - \mathcal{T}(y_m)||^2 \Big\}, \end{aligned}$$

where $|\cdot|$ represents the determinant of a square matrix. Unlike in the original CPD paper, we define a GMM with an explicit definition of unknown correspondence, i.e., c_n and e_n. Under assumptions 1, 2, 3, and 4, we define the joint distribution of (x_n, e_n, c_n) given $(y, v, \alpha, \rho, \sigma^2)$ as a combination of a two-component mixture distribution and an M-component mixture distribution as follows:

$$\begin{aligned} &p(x_n, e_n, c_n | y, v, \alpha, \rho, \sigma^2) \\ &\quad = \{\omega p_{\mathrm{out}}(x_n)\}^{1-c_n} \left\{ (1-\omega) \prod_{m=1}^{M} (\alpha_m \phi_{mn})^{\delta_m(e_n)} \right\}^{c_n}, \end{aligned}$$

where ϕ_{mn} is an abbreviation of $\phi(x_n; \mathcal{T}(y_m), \sigma^2 I_D)$ and $\delta_m(e_n)$ is the indicator function, with a value of 1 if $e_n = m$ and 0 otherwise. Notation with respect to corresponding points is shown in Fig. 1.

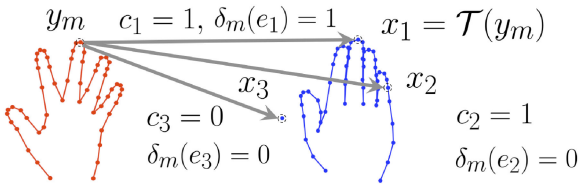


Fig. 1. Notation with respect to corresponding points. The point sets colored red and blue represent the source point set Y and the target point set X, respectively. The variable $c_n \in \{0, 1\}$ indicates whether or not x_n is a non-outlier, whereas the variable $e_n \in \{1, \ldots, M\}$ specifies the index of a point in Y that corresponds to x_n. Target points x_1, x_2, and x_3 represent the point that corresponds to y_m, a non-outlier that does not correspond to y_m, and an outlier, respectively.

Prior Distributions. We define motion coherence as a prior distribution of displacement vectors. Suppose $G = (g_{mm'}) \in \mathbb{R}^{M \times M}$ with $g_{mm'} = \mathcal{K}(y_m, y_{m'})$ is a positive-definite matrix, where $\mathcal{K}(\cdot, \cdot)$ is a positive-definite kernel. We define the prior distribution of v as follows:

$$p(v|y) = \phi(v; 0, \lambda^{-1} G \otimes I_D), \tag{5}$$

where λ is a positive constant and $\otimes$ denotes the Kronecker product. The prior distribution encodes motion coherence because displacement vectors v_m and $v_{m'}$ correlate with each other if the affinity value between y_m and $y_{m'}$, i.e., $\lambda^{-1} g_{mm'}$, is sufficiently large. Additionally, we define $p(\alpha)$ as a Dirichlet distribution

$$p(\alpha) = \mathrm{Dir}(\alpha|\kappa 1_M),$$

where $\kappa > 0$ is the parameter that controls the shape of the Dirichlet distribution and 1_M is the vector of all 1 s of size M. For the simplicity of the generative model, we do not introduce prior distributions over the source point set y, the residual variance σ^2, or the set of variables ρ that defines a similarity transformation T.

Full Joint Distribution. Incorporating the prior distributions into the GMM with assumption 5, we define the full joint distribution $p(x, y, \theta)$ as follows:

$$p(x, y, \theta) \propto p(v|y) p(\alpha) \prod_{n=1}^{N} p(x_n, c_n, e_n | y, v, \alpha, \rho, \sigma^2),$$

where $\theta = (v, \alpha, c, e, \rho, \sigma^2)$ is the set of latent variables that mediate the generation of x given y. This definition of the joint distribution replaces a point set registration with a probabilistic density estimation. In the next section, we derive an algorithm for estimating a reasonable θ using VBI.

4.3 Derivation of the BCPD Algorithm

In this section, we derive a registration algorithm based on the Bayesian formulation described in Section 4.2. One approach to solving a registration problem is to compute a reasonable estimate of θ, such as the expectation of $\theta = (v, \alpha, c, e, \rho, \sigma^2)$ over $p(\theta|x, y)$, or the maximum mode of $p(\theta|x, y)$. The exact computation of the estimate is, however, intractable for large M and N; the number of evaluations of $p(\theta|x, y)$ is roughly $(M+1)^N$ times when a naïve method is applied. Therefore, we use VBI as described in Section 4.1. To relax the difficulty in computing the estimate, we approximate $p(\theta|x, y)$ as a distribution $q(\theta)$ that is endowed with the factorization

$$q(\theta) = q_1(v, \alpha) q_2(c, e) q_3(\rho, \sigma^2).$$

We also assume $q_3(\rho, \sigma^2) = \delta(\rho, \sigma^2)$ for simplification, where the symbol δ with no subscript represents a Dirac delta function, i.e., q_3 is the distribution with a point mass at (ρ, σ^2). This assumption means that q_3 is characterized by the first moment only, and the second and higher moments can be represented as the product of the first moment. Through VBI, the maximum mode or the expectation regarding q will be available after performing iterative updates of q_1, q_2, and q_3 until convergence. Furthermore, VBI guarantees convergence of the algorithm.

Notation 3. Here, we list the useful symbols for describing the closed-form expressions of $\hat{q}_1$, $\hat{q}_2$, and $\hat{q}_3$ as follows:

- $G = (g_{mm'}) \in \mathbb{R}^{M \times M}$ with $g_{mm'} = \mathcal{K}(y_m, y_{m'})$ – the Gram matrix to define motion coherence, where $\mathcal{K}(\cdot, \cdot)$ is a positive-definite kernel.
- $P = (p_{mn}) \in [0, 1]^{M \times N}$ – the probability matrix where $p_{mn} = E[c_n \delta_m(e_n)]$ represents the posterior probability that x_n corresponds to y_m.
- $\nu = (\nu_1, \ldots, \nu_M)^T \in \mathbb{R}^M$ with $\nu_m = \sum_{n=1}^{N} p_{mn}$ – the estimated numbers of target points matched with each source point.
- $\nu' = (\nu'_1, \ldots, \nu'_N)^T \in [0, 1]^N$ with $\nu'_n = \sum_{m=1}^{M} p_{mn}$ – the vector of probabilities where ν'_n represents the posterior probability that x_n is a non-outlier.
- $\hat{N} = \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn} \leq N$ – the estimated number of matching points between X and Y.
- ϕ_{mn} – the abbreviation of $\phi(x_n; \mathcal{T}(y_m), \sigma^2 I_D)$.

For convenience, we simplify the notation concerning the Kronecker product by attaching a tilde symbol to a matrix or a vector as follows:

$$\tilde{G} = G \otimes I_D, \quad \tilde{\nu} = \nu \otimes 1_D,$$
$$\tilde{P} = P \otimes I_D, \quad \tilde{\nu}' = \nu' \otimes 1_D.$$

As with the above notation, we denote the augmented form of the similarity transformation by $\tilde{T}$, i.e.,

$$\tilde{T}(y) = s(I_M \otimes R) y + (1_M \otimes t).$$

Hereafter, we provide the closed-form expressions of $\hat{q}_1$, $\hat{q}_2$, and $\hat{q}_3$ without derivations, which are available in Appendices A, B, and C, available in the online supplemental material.

4.3.1 Update of $q_1(v, \alpha)$

The update of $q_1(v, \alpha)$ corresponds to the update of the nonlinear shape deformation. Suppose $q_2(c, e)$ and $q_3(\rho, \sigma^2)$ are given. From Eq. (4), the closed-form expression of $\hat{q}_1(v, \alpha)$ can be obtained as the product of a Dirichlet distribution and a multivariate normal distribution as follows:

Proposition 1. *The approximated posterior distribution $\hat{q}_1(v, \alpha)$ is factorized into its marginals, i.e., $\hat{q}_1(v, \alpha) = \hat{q}_\alpha(\alpha)\hat{q}_v(v)$. Furthermore, $\hat{q}_\alpha(\alpha)$ is a Dirichlet distribution and $\hat{q}_v(v)$ is a multivariate normal distribution, which are defined as follows:*

$$\hat{q}_\alpha(\alpha) = \mathrm{Dir}(\alpha|\kappa 1_M + \nu),$$
$$\hat{q}_v(v) = \phi\Big(v; \frac{s^2}{\sigma^2} \tilde{\Sigma}\, \mathrm{d}(\tilde{\nu})(\tilde{T}^{-1}(\hat{x}) - y), \tilde{\Sigma}\Big),$$

where $\hat{x} = \mathrm{d}(\tilde{\nu})^{-1}\tilde{P}x \in \mathbb{R}^{DM}$ is the shape for which $\tilde{T}^{-1}(\hat{x})$ is interpreted as the inverse alignment from x to y, and $\tilde{\Sigma} = \Sigma \otimes I_D \in \mathbb{R}^{DM\times DM}$ with $\Sigma = (\lambda G^{-1} + \frac{s^2}{\sigma^2}\mathrm{d}(\nu))^{-1} \in \mathbb{R}^{M\times M}$ is the posterior covariance matrix of displacement vector v.

For reference, we list the vectors of size DM, which are related to $q_v(v)$ and used hereafter, as follows:

$$\hat{x} = (\hat{x}_1^T, \ldots, \hat{x}_M^T)^T = \mathrm{d}(\tilde{\nu})^{-1}\tilde{P}x,$$
$$\hat{v} = (\hat{v}_1^T, \ldots, \hat{v}_M^T)^T = \frac{s^2}{\sigma^2}\tilde{\Sigma}\mathrm{d}(\tilde{\nu})(\tilde{T}^{-1}(\hat{x}) - y),$$
$$\hat{u} = (\hat{u}_1^T, \ldots, \hat{u}_M^T)^T = y + \hat{v},$$
$$\hat{y} = (\hat{y}_1^T, \ldots, \hat{y}_M^T)^T = \tilde{T}(y + \hat{v}).$$

This proposition provides some insight into motion coherence. Because $T^{-1}(\hat{x}_m) - y_m$ is interpreted as the residual for the mth source point, the meaning of the mean vector $\hat{v} = \frac{s^2}{\sigma^2}\tilde{\Sigma}\mathrm{d}(\tilde{\nu})(\tilde{T}^{-1}(\hat{x}) - y)$ can be intuitively explained as follows:

- If λ is sufficiently large, $\hat{v}_m$ is approximated as $\hat{v}_m \approx \frac{s^2}{\lambda\sigma^2}\sum_{m'=1}^{M} g_{mm'}\ \nu_{m'}(T^{-1}(\hat{x}_{m'}) - y_{m'})$, i.e., $\hat{v}_m$ is computed by smoothing the weighted residual vector $\nu_m(T^{-1}(\hat{x}_m) - y_m)$ with basis functions $g_{m1}, \ldots, g_{mM}$.
- In contrast, if λ is close to zero, $\hat{v}_m$ is approximated as $\hat{v}_m \approx T^{-1}(\hat{x}_m) - y_m$, i.e., $\hat{v}_m$ is estimated as the residual vector $T^{-1}(\hat{x}_m) - y_m$ without smoothing.

In Section 4.5, we further discuss motion coherence from a statistical aspect.

Remark. The proposition allows us to compute $\langle\alpha_m\rangle = \exp(E[\ln\alpha_m])$ and $\langle\phi_{mn}\rangle = \exp(E[\ln\phi_{mn}])$, which are required to update $q_2(c,e)$. By using standard results regarding the Dirichlet distribution and Remark 1 in Appendix D, available in the online supplemental material, $\langle\alpha_m\rangle$ and $\langle\phi_{mn}\rangle$ are updated as follows:

$$\langle\alpha_m\rangle = \exp\big\{\psi(\kappa + \nu_m) - \psi(\kappa M + \hat{N})\big\}, \tag{6}$$

$$\langle\phi_{mn}\rangle = \phi(x_n; \hat{y}_m, \sigma^2 I_D)\exp\Big\{-\frac{s^2}{2\sigma^2}\mathrm{Tr}(\sigma_m^2 I_D)\Big\}, \tag{7}$$

where $\psi(\cdot)$ is the digamma function, also known as the ψ-function, and σ_m^2 is the mth diagonal element of Σ.

4.3.2 Update of $q_2(c,e)$

Next, we proceed to the update of $q_2(c,e)$ that encodes shape correspondence between X and Y. Suppose $q_1(v,\alpha)$ and $q_3(\rho,\sigma^2)$ are given. From Eq. (4), we obtain the closed-form expression of $\hat{q}_2(c,e)$ as follows:

Proposition 2. *The approximated posterior distribution $\hat{q}_2(c,e)$ is a combination of a Bernoulli distribution and a categorical distribution, and is defined as follows:*

$$\hat{q}_2(c,e) = \prod_{n=1}^{N}(1-\nu'_n)^{1-c_n}\left\{\nu'_n\prod_{m=1}^{M}\Big(\frac{p_{mn}}{\nu'_n}\Big)^{\delta_m(e_n)}\right\}^{c_n},$$

where p_{mn} and ν'_n are defined as

$$p_{mn} = \frac{(1-\omega)\langle\alpha_m\rangle\langle\phi_{mn}\rangle}{\omega p_{\mathrm{out}}(x_n) + (1-\omega)\sum_{m'=1}^{M}\langle\alpha_{m'}\rangle\langle\phi_{m'n}\rangle}, \tag{8}$$

and $\nu'_n = \sum_{m=1}^{M} p_{mn}$.

From this proposition, we see that $\hat{q}_2(c,e)$ is factorized into $\hat{q}_2(c,e) = \prod_{n=1}^{N}\hat{q}_2^{(n)}(c_n,e_n)$. Additionally, the proposition provides the following observations, some of which are consistent with the description in Notation 3:

- The definition of p_{mn} in Proposition 2 is consistent with the one in Notation 3 because of $E[c_n\delta_m(e_n)] = \hat{q}_2^{(n)}(c_n = 1, e_n = m) = p_{mn}$.
- The posterior marginal distribution of c_n is the Bernoulli distribution with the probability ν'_n, and thus, the posterior probability of x_n being a non-outlier is ν'_n.
- The posterior conditional distribution of e_n given $c_n = 1$ is the categorical distribution with M categoriy probabilites $\{p_{mn}/\nu'_n\}_{m=1}^{M}$.
- The number of target points matching with y_m can be estimated as ν_m because of $E[\sum_{n=1}^{N} c_n\delta_m(e_n)] = \nu_m$.
- The number of matching points between the target and source point sets, X and Y, can be estimated as $\hat{N}$ because of $E[\sum_{n=1}^{N}\sum_{m=1}^{M} c_n\delta_m(e_n)] = \hat{N}$.

This proposition guarantees that the update of $P = (p_{mn})$ with Eq. (8) increases the lower bound. The terms relating to P, i.e., $\nu = P1_N$, $\nu' = P^T1_M$, $\hat{N} = \nu^T1_M$, are updated in turn. We also see that the expectations required by the update of $\hat{q}_2$ are summarized into $\langle\alpha_m\rangle$ and $\langle\phi_{mn}\rangle$, which are computed by Eqs. (6) and (7).

4.3.3 Update of $q_3(\rho,\sigma^2)$

The update of $q_3(\rho,\sigma^2)$ corresponds to the update of the similarity transformation T, parameterized by $\rho = (s,R,t)$. Suppose $q_1(v,\alpha)$ and $q_2(c,e)$ are given. From the assumption that q_3 is a Dirac delta function, q_3 is characterized only by its mode. Therefore, maximizing the lower bound $L(q)$ without resorting to Eq. (4) allows us to find the mode of $q_3(\rho,\sigma^2)$ that increases the lower bound. Let us define the following notation:

$$\bar{x} = \frac{1}{\hat{N}}\sum_{m=1}^{M}\nu_m\hat{x}_m,\quad \bar{\sigma}^2 = \frac{1}{\hat{N}}\sum_{m=1}^{M}\nu_m\sigma_m^2,$$
$$\bar{u} = \frac{1}{\hat{N}}\sum_{m=1}^{M}\nu_m\hat{u}_m,$$
$$S_{xu} = \frac{1}{\hat{N}}\sum_{m=1}^{M}\nu_m(\hat{x}_m - \bar{x})(\hat{u}_m - \bar{u})^T,$$
$$S_{uu} = \frac{1}{\hat{N}}\sum_{m=1}^{M}\nu_m(\hat{u}_m - \bar{u})(\hat{u}_m - \bar{u})^T + \bar{\sigma}^2 I_D,$$

where σ_m^2 is the mth diagonal element of Σ. Using this notation, we obtain the following proposition:

Proposition 3. *Suppose the approximated posterior distribution $q_3(\rho,\sigma^2)$ is a Dirac delta function. Given $q_1(v,\alpha)$ and $q_2(c,e)$, the lower bound is maximized by the following equations:*

$$\hat{R} = \Phi\mathrm{d}(1,\ldots,1,|\Phi\Psi^T|)\Psi^T,$$
$$\hat{s} = \mathrm{Tr}\big(\hat{R}^T S_{xu}\big)/\mathrm{Tr}\big(S_{uu}\big),$$
$$\hat{t} = \bar{x} - \hat{s}\hat{R}\bar{u},$$
$$\hat{\sigma}^2 = \frac{1}{\hat{N}D}\big\{x^T\mathrm{d}(\tilde{\nu}')x - 2x^T\tilde{P}^T\hat{y} + \hat{y}^T\mathrm{d}(\tilde{\nu})\hat{y}\big\} + \hat{s}^2\bar{\sigma}^2,$$

where Φ and Ψ are the orthogonal matrices of size $D \times D$ obtained by the singular value decomposition of S_{xu}, i.e., $S_{xu} = \Phi S'_{xu} \Psi^T$ with the diagonal matrix of singular values, $S'_{xu} \in \mathbb{R}^{D \times D}$.

We note that the update of σ^2 is important as it gradually shrinks the radius to define a candidate set of target points that corresponds to each source point [5].

4.3.4 Outlier Distribution

Myronenko *et al.* used the uniform distribution defined as $p_{\text{out}}(x_n) = 1/N$. The distribution does not satisfy one of the conditions concerning a probabilistic density function; the integral of $p_{\text{out}}(x_n)$ over $\mathbb{R}^D$ is not one; the integral approaches zero for any ω as N becomes larger if the nonzero region of $p_{\text{out}}(x_n)$ is bounded, otherwise the integral becomes infinity. Suppose V is the volume of the minimum bounding box that covers the target point set X. To avoid the normalization issue, we use an estimate of the D-dimensional uniform distribution: $p_{\text{out}}(x_n) = 1/V$ if x_n is in the bounding box and $p_{\text{out}}(x_n) = 0$ otherwise.

4.3.5 Initialization

VBI requires the initialization of the expectations for the random variables, as described in Section 4.1. We initialize the expectations in a non-informative manner; we set $\hat{v} = 0$, $\langle \alpha_m \rangle = 1/M$, $s = 1$, $R = I_D$, and $t = 0$. The initial guess of residual variance σ^2 controls the randomness of point matching during the early stage of the optimization. This is suggested by the fact that p_{mn} approaches $1/M$ for any m and n if the outlier probability ω is set to zero and the residual variance σ^2 in Eqs. (7) and (8) goes to infinity. CPD uses $\sigma^2_{\text{ini}} = \frac{1}{NMD} \sum_{n=1}^{N} \sum_{m=1}^{M} ||x_n - y_m||^2$ as an initial guess. For our Bayesian variant of CPD, the use of σ^2_{ini} is often too informative, especially when the target shape is largely rotated, and the solution often falls into local optima. Therefore, we use $\gamma \sigma^2_{\text{ini}}$ as our initial guess of σ^2, where γ is a positive constant. As γ increases, point matching during the early stage of the optimization becomes random, and a moderately large γ often stabilizes registration. The BCPD algorithm is summarized in Fig. 2.

4.4 Relation to the Original CPD

In this section, we show that CPD is recovered from BCPD. First, we summarize the differences between CPD and BCPD. The differences are listed as follows:

- The transformation model $\mathcal{T}(y_m)$ is a combination of non-rigid and similarity transformations, where the non-rigidity is derived from the displacement vector v.
- The conditional distribution $p(v|y) = \phi(v; 0, \lambda^{-1}\tilde{G})$ and the prior distribution $p(\alpha) = \text{Dir}(\alpha|\kappa 1_M)$ are imposed on the displacement vector v and the mixing coefficients α, respectively.
- The optimization is based on VBI instead of the EM algorithm used in CPD.
- The initial guess of σ^2 is parameterized by γ, which controls the randomness of point matching during the early stage of the optimization.

Algorithm: Bayesian Coherent Point Drift

· Input: $x \in \mathbb{R}^{DN}$, $y \in \mathbb{R}^{DM}$, ω, λ, κ, γ.

· Output:

$$\hat{y} = \tilde{T}(y + \hat{v}) = s(I_M \otimes R)(y + \hat{v}) + (1_M \otimes t).$$

· Initialization:

$$\hat{y} = y,\ \hat{v} = 0,\ \Sigma = I_M,\ s = 1,\ R = I_D,\ t = 0,$$
$$\langle \alpha_m \rangle = \tfrac{1}{M},\ \sigma^2 = \tfrac{\gamma}{NMD} \textstyle\sum_{n=1}^{N} \sum_{m=1}^{M} ||x_n - y_m||^2,$$
$$G = (g_{mm'}) \text{ with } g_{mm'} = \mathcal{K}(y_m, y_{m'}).$$

· Optimization: Repeat until convergence.

- Update $P = (p_{mn})$ and related terms.

$$\langle \phi_{mn} \rangle = \phi(x_n; \hat{y}_m, \sigma^2 I_D) \exp\{-\tfrac{s^2}{2\sigma^2} \text{Tr}(\sigma_m^2 I_D)\},$$
$$p_{mn} = \frac{(1-\omega)\langle \alpha_m \rangle \langle \phi_{mn} \rangle}{\omega p_{\text{out}}(x_n) + (1-\omega) \sum_{m'=1}^{M} \langle \alpha_{m'} \rangle \langle \phi_{m'n} \rangle},$$
$$\nu = P 1_N,\ \nu' = P^T 1_M,\ \hat{N} = \nu^T 1_M,\ \hat{x} = \text{d}(\tilde{\nu})^{-1} \tilde{P} x.$$

- Update Σ, $\hat{v}$, $\hat{u}$, and $\langle \alpha_m \rangle$ for all m.

$$\Sigma^{-1} = \lambda G^{-1} + \tfrac{s^2}{\sigma^2} \text{d}(\nu),\ \hat{v} = \tfrac{s^2}{\sigma^2} \tilde{\Sigma} \text{d}(\tilde{\nu})(\tilde{T}^{-1}(\hat{x}) - y),$$
$$\hat{u} = y + \hat{v},\ \langle \alpha_m \rangle = \exp\{\psi(\kappa + \nu_m) - \psi(\kappa M + \hat{N})\}.$$

- Update s, R, t, σ^2, $\hat{y}$ and related terms.

$$\bar{x} = \tfrac{1}{\hat{N}} \textstyle\sum_{m=1}^{M} \nu_m \hat{x}_m,\ \ \bar{\sigma}^2 = \tfrac{1}{\hat{N}} \sum_{m=1}^{M} \nu_m \sigma_m^2,$$
$$\bar{u} = \tfrac{1}{\hat{N}} \textstyle\sum_{m=1}^{M} \nu_m \hat{u}_m,$$
$$S_{xu} = \tfrac{1}{\hat{N}} \textstyle\sum_{m=1}^{M} \nu_m (\hat{x}_m - \bar{x})(\hat{u}_m - \bar{u})^T,$$
$$S_{uu} = \tfrac{1}{\hat{N}} \textstyle\sum_{m=1}^{M} \nu_m (\hat{u}_m - \bar{u})(\hat{u}_m - \bar{u})^T + \bar{\sigma}^2 I_D,$$
$$\Phi S'_{xu} \Psi^T = \text{svd}(S_{xu}),\ R = \Phi \text{d}(1, \cdots, 1, |\Phi\Psi|) \Psi^T,$$
$$s = \text{Tr}(R S_{xu}) / \text{Tr}(S_{uu}),\ t = \bar{x} - sR\bar{u},\ \hat{y} = \tilde{T}(y + \hat{v}),$$
$$\sigma^2 = \tfrac{1}{\hat{N}D} \{x^T \text{d}(\tilde{\nu}') x - 2x^T \tilde{P}^T \hat{y} + \hat{y}^T \text{d}(\tilde{\nu}) \hat{y}\} + s^2 \bar{\sigma}^2.$$

Fig. 2. BCPD algorithm. The tilde symbol attached to a matrix denotes the Kronecker product between the matrix and I_D, e.g., $\tilde{\Sigma} = \Sigma \otimes I_D$ and $\tilde{P} = P \otimes I_D$, whereas the tilde symbol attached to a vector denotes the Kronecker product between the vector and 1_D, e.g., $\tilde{\nu} = \nu \otimes 1_D$ and $\tilde{\nu}' = \nu' \otimes 1_D$. The symbol ψ represents the digamma function. The mth diagonal element of Σ is denoted by σ_m^2. The singular value decomposition is denoted by "svd." The determinant and trace of a square matrix are denoted by $|\cdot|$ and $\text{Tr}(\cdot)$, respectively. Unlike CPD, BCPD simultaneously estimates the variables of non-rigid and similarity transformations. The acceleration of computations G, Σ, and P is discussed in Section 4.6.

The CPD algorithm can be recovered by removing these differences, i.e., by imposing the following constraints on our Bayesian formulation:

1) The parameter κ is set to infinity.
2) The similarity transformation T is replaced by the identity function.
3) The posterior distribution $q_v(v)$ is assumed to be a Dirac delta function.
4) The outlier distribution is defined as $p_{\text{out}}(x_n) = 1/N$.
5) The parameter γ is set to 1.

Condition 1 constrains all mixing coefficients to be equivalent, i.e., $\langle \alpha_m \rangle = 1/M$ for all m, and recovers the definition of CPD with respect to the mixing coefficients. Condition 2 removes the update of $\rho = (s, R, t)$ and replaces T and T^{-1} with the identity function. Condition 3 means that the second moment of $q_v(v)$, denoted by $E[vv^T]$, is equivalent to the product of the first moment, i.e., $\hat{v}\hat{v}^T$. The condition drops the terms $\exp\{-\frac{s^2}{2\sigma^2} \text{Tr}(\sigma_m^2 I_D)\}$, $\bar{\sigma}^2 I_D$, and $s^2\bar{\sigma}^2$ from the

equations for updating $\langle \phi_{mn} \rangle$, S_{uu}, and σ^2, respectively. Condition 4 replaces $p_{\text{out}}(x_n)$ by $1/N$. Therefore, the conditions provide the same equation for updating p_{mn} as in the E-step in the CPD algorithm

$$p_{mn} = \frac{\exp\{-\frac{1}{2\sigma^2}||x_n - (y_m + \hat{v}_m)||^2\}}{c_0 + \sum_{m'=1}^{M} \exp\{-\frac{1}{2\sigma^2}||x_n - (y_{m'} + \hat{v}_{m'})||^2\}}, \tag{9}$$

where $c_0 = \frac{\omega}{1-\omega}\frac{M}{N}|2\pi\sigma^2 I_D|^{1/2}$. Let us denote the expectation of a displacement vector $\hat{v}$ by the product between $\tilde{G} = G \otimes I_D \in \mathbb{R}^{DM \times DM}$ and vector $w \in \mathbb{R}^{DM}$, i.e., $\hat{v} = \tilde{G}w$. From the definitions of $\hat{v}$ and $\tilde{\Sigma}$, we obtain the same equations for updating w and σ^2 as in the M-step in the CPD algorithm

$$w = (\tilde{G} + \lambda\sigma^2 \mathrm{d}(\tilde{v})^{-1})^{-1}(\hat{x} - y),$$
$$\sigma^2 = \frac{1}{\hat{N}D}\left\{x^T \mathrm{d}(\tilde{v}')x - 2x^T\tilde{P}^T\hat{y} + \hat{y}^T\mathrm{d}(\tilde{v})\hat{y}\right\}.$$

Therefore, the CPD algorithm is recovered by imposing the conditions on the Bayesian formulation of CPD, i.e., BCPD is a generalization of CPD. Here, we note some details regarding the algorithms:

- Convergence of CPD, as well as BCPD, is guaranteed by VBI.
- If we replace condition 2 with $v = 0$, BCPD becomes the rigid CPD, meaning that the non-rigid CPD and the rigid CPD are combined as a single algorithm.

We also note that Eq. (9) suggests a connection between the CPD algorithm and the iterative closest point (ICP) algorithm [7], which is one of the best-known registration algorithms. We then describe the connection to provide an insight into BCPD.

Relation to ICP. We show that ICP is a special case of BCPD regarding the estimation of corresponding points. ICP solves the registration problem by repeatedly finding the point in the current deformed shape that is closest to each point in the target shape, and updating the deformed shape using the current pairs of closest points. From the description in Section 3.2, we see that the first and second steps of ICP are associated with the E-step and M-step of CPD, respectively. To describe a connection between BCPD and ICP, we focus on the E-step of CPD, i.e., the computation of matching probabilities. From Eq. (9), the matching probability p_{mn} becomes a so-called softmax function for a fixed n if ω is zero and all points in Y are not identical to one another. Here, we assume that σ^2 is infinitesimal. Under these conditions, we have $p_{mn} = 1$ if y_m is the closest to x_n among $\{y_1, \ldots, y_M\}$ and $p_{m'n} = 0$ for all $m' \neq m$. This corresponds to the first step of ICP, i.e., finding the closest point in Y for x_n. Therefore, we see that the first step of ICP is a special case of the E-step in the CPD algorithm, which means that BCPD is a generalization of ICP concerning the correspondence estimation.

4.5 Statistical View of Motion Coherence

The Bayesian formulation in Section 4.2 provides a clear difference between tuning parameters that control motion coherence. Here, we show that the directional correlation between displacement vectors is dependent only on the Gram matrix G regardless of λ. As shown in Fig. 3, if G is the Gaussian affinity matrix parameterized by β, i.e., $G = (g_{mm'}^{(\beta)})$ with $g_{mm'}^{(\beta)} = \exp(-\frac{1}{2\beta^2}||y_m - y_{m'}||^2)$, we will see that

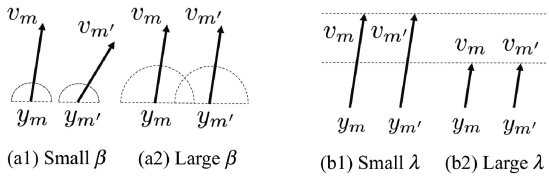


Fig. 3. Interpretation of tuning parameters β and λ when the Gram matrix G is Gaussian. (a) β controls the directional correlation between displacement vectors. (b) λ controls the expected length of displacement vectors.

1) β controls the directional correlation between displacement vectors, and
2) λ controls the expected length of displacement vectors.

We begin with the case that G is non-Gaussian. From the definition of $p(v|y)$, i.e., Eq. (5), the prior covariance matrix between v_m and $v_{m'}$ is defined as

$$\mathrm{Cov}(v_m, v_{m'}) = \lambda^{-1} g_{mm'} I_D, \tag{10}$$

for all $m,\ m' \in \{1, \ldots, M\}$. From this equation, the Pearson correlation between v_{md} and $v_{m'd}$ is obtained as

$$\mathrm{Corr}(v_{md}, v_{m'd}) = g_{mm'}/\sqrt{g_{mm}g_{m'm'}}.$$

The correlation is not dependent on the index d, and thereby defines a measure of directional correlation between displacement vectors. Intuitively, the correlation represents the deviation of the angle between displacement vectors; the angle becomes zero without randomness if the correlation is 1 whereas the angle becomes random if the correlation is 0. We note that the correlation is not dependent on λ; it does not contribute to the deviation of the angle between displacement vectors. We also note that the correlation always becomes 1 if two points y_m and $y_{m'}$ are the same and the corresponding $g_{mm'}$ is not zero.

Next, we assume that G is Gaussian. We begin with the role of β. From Eq. (10), the Pearson correlation between the dth elements of v_m and v'_m is obtained as follows:

$$\mathrm{Corr}(v_{md}, v_{m'd}) = g_{mm'}^{(\beta)}, \tag{11}$$

for all d because $g_{mm}^{(\beta)}$ is always 1 for all m. As discussed previously, the correlation represents a measure of directional correlation between v_m and $v_{m'}$. The correlation is dependent only on β, regardless of λ. Therefore, β is interpreted as a parameter that controls the directional correlation between displacement vectors. From this equation, we also see that the correlation increases as the distance between y_m and $y_{m'}$ decreases, and it becomes 1 if the distance is 0. This means that the displacement field for registering point sets is smooth, i.e., displacement vectors gradually become parallel as the distance between two points decreases.

We then proceed to the role of λ when G is Gaussian. From Eq. (10), we see that $\lambda||v_m||^2$ follows the χ^2-distribution with the degree of freedom D. Therefore, the square root of the expected squared norm of v_m, denoted by l_m, is obtained as follows:

$$l_m = \sqrt{\lambda^{-1} D}, \tag{12}$$

which is not dependent on m. From this equation, we see that λ is the parameter that controls the expected length of displacement vectors regardless of β. We note that Eqs. (11) and (12) hold for a non-Gaussian kernel if the kernel function parameterized by β is normalized, i.e., $g_{mm}^{(\beta)} = 1$ for all m. We finally note that BCPD falls into a rigid registration technique for sufficiently large λ because l_m becomes zero for all m if λ goes to infinity.

4.6 Acceleration of the BCPD Algorithm

In this section, we present an acceleration scheme for the BCPD algorithm, which can be applied even if the Gram matrix G is non-Gaussian. The bottleneck in BCPD lies in the computations of the Gram matrix G, the posterior matching probability P, and the covariance matrix $\Sigma = (\lambda G^{-1} + \frac{s^2}{\sigma^2}\mathrm{d}(\nu))^{-1}$, whose computational costs are $O(M^2)$, $O(MN)$, and $O(M^3)$, respectively. The basis of our acceleration scheme is the combination of the Nyström method [56] and the KD tree search [57], which allow the computations of G, P, and Σ in at most $O((M+N)\log(M+N))$ time. We note that the initialization of σ^2 is not a bottleneck as it can be computed in $O(M+N)$ by changing the order of the summation and multiplication.

By using the fast Gauss transform (FGT) [58], the original CPD accelerates the eigendecomposition of G at the initialization step and the computation of P at the E-step. Owing to the acceleration, CPD is scalable to large point sets, yet (1) the eigendecomposition of G is still time-consuming [59] and (2) the computation of P requires the evaluations of truncated Gaussian distributions near convergence [2], the computational cost of which is $O(MN)$. Dupej *et al.* relaxed (1) by computing the approximate eigendecomposition of G using the Nyström method and the improved fast Gauss transform (IFGT) [59]. The IFGT is an improved version of the FGT, and it efficiently approximates the sum of Gaussian functions in high-dimensional spaces [60]. Lu *et al.* relaxed (2) by reducing the number of EM iterations using the squared iterative EM algorithm, combined with a variant of IFGT [61]. A problem of using the FGT and IFGT is that the Gram matrix G must be Gaussian. To address this issue, we accelerate the computation of G using only the Nyström method. In addition, to relax (2), we accelerate the computation of P by the Nyström method during the early stage of the optimization and the KD tree search near convergence, which do not require $O(MN)$ computation.

4.6.1 Fast Computation of G and Σ

The Nyström method generates a low-rank approximation of a Gram matrix using random sampling in $O(M)$ time, and it can also be applied to the approximated computation of the rank-restricted eigendecomposition in $O(M)$ time [56]. Suppose K is the number of random samples. We also suppose that the approximated eigendecomposition of G is denoted by $G \approx Q\Lambda Q^T$ where $Q \in \mathbb{R}^{M\times K}$ is a column-orthonormal matrix and $\Lambda \in \mathbb{R}^{K\times K}$ is a diagonal matrix, the mth element of which is the mth approximated eigenvalue of G. The use of the eigendecomposition and the Woodbury identity replaces the computation of Σ as follows:

$$\Sigma \approx \frac{1}{\lambda} Q\Lambda \left\{ I_K - S\left(\frac{\lambda\sigma^2}{s^2}\Lambda^{-1} + S \right)^{-1} \right\} Q^T,$$

where $S = Q^T \mathrm{d}(\nu) Q$. Here, we note that the evaluations of all elements in the approximated Σ, whose computational cost is $O(M^2)$, are not required; the required computations regarding Σ are the displacement vector $\hat{v} = \frac{s^2}{\sigma^2}\tilde{\Sigma}\mathrm{d}(\tilde{\nu})(\tilde{T}^{-1}(\hat{x}) - y)$ and the diagonal elements of Σ for computing $\langle \phi_{mn} \rangle$, S_{uu}, and $\bar{\sigma}^2$. The computational costs of evaluating all of them are $O(M)$ if we assume that K is a constant.

4.6.2 Fast Computation of P

The first step to accelerating the evaluation of P is to replace it with the evaluation of the Gaussian affinity matrix between Y and X, denoted by $K_{YX} \in \mathbb{R}^{M\times N}$, where the mnth element of K_{YX} is defined as $\exp\{-\frac{1}{2\sigma^2}||x_n - \hat{y}_m||^2\}$. Suppose $x_{(d)} = (x_{1d}, \ldots, x_{Nd})^T \in \mathbb{R}^N$ is the vector constructed from the dth elements of $x_1, \ldots, x_N$. Then, the terms involving P are replaced by those involving K_{YX} on the basis of the following observations:

1) All computations regarding P can be expressed as one of the products $\tilde{P}x$, $\nu' = P^T 1_M$, or $\nu = P1_N$.
2) The vector $\tilde{P}x$ can be computed through the products $Px_{(1)}, \ldots, Px_{(D)}$.
3) The computations of $Px_{(d)}$, $P^T 1_M$, and $P1_N$ can be replaced by the products involving K_{YX}.

The first observation is shown in Fig. 2. The third observation is obtained as follows:

$$\begin{aligned} q &= 1_N \circ' (K_{YX}^T b + c_0' 1_N), \\ P^T 1_M &= 1_N - c_0' q, \\ P1_N &= b \circ (K_{YX} q), \\ Px_{(d)} &= b \circ \{K_{YX}(q \circ x_{(d)})\}, \end{aligned}$$

where $\circ$ and $\circ'$ represent elementwise product and division, respectively, $c_0' = \frac{\omega}{1-\omega} p_{\mathrm{out}}(x_n) |2\pi\sigma^2 I_D|^{1/2}$ is a constant, and $b \in \mathbb{R}^M$ represents a vector, the mth element of which is $b_m = \langle \alpha_m \rangle \exp\{-\frac{s^2}{2\sigma^2}\mathrm{Tr}(\sigma_m^2 I_D)\}$. Therefore, if the computation of terms involving K_{YX} is accelerated, the evaluation of terms involving P is also accelerated. We accelerate those separately for large σ and small σ.

a) *For Large σ.* The matrix–vector products involving K_{YX} can be approximated in $O(M+N)$ time by using the Nyström method. Suppose V is a point set constructed by the random sampling of points from the union between X and Y, and we suppose J is the number of elements in V. The Nyström method guarantees

$$K_{YX} \approx K_{YV} K_{VV}^{-1} K_{XV}^T,$$

where $K_{YV} \in \mathbb{R}^{M\times J}$, $K_{XV} \in \mathbb{R}^{N\times J}$, and $K_{VV} \in \mathbb{R}^{J\times J}$ are the Gaussian affinity matrices between Y and V, between X and V, and for V itself, respectively. Therefore, if D and J are assumed to be constants that are much less than M and N,

TABLE 1
Computational Costs for the Evaluation of the Bottleneck Terms Involving G, Σ, and P

Method	G	Σ	P
Naïve	$O(M^2)$	$O(M^3)$	$O(MN)$
Nyström	$O(M)$	$O(M)$	$O(M+N)$
KD tree	–	–	$O((M+N)\log(M+N))$

The computations of terms involving G and Σ are accelerated by the Nyström method throughout the optimization, whereas those involving P are accelerated by the Nyström method for large σ and the KD tree search for small σ.

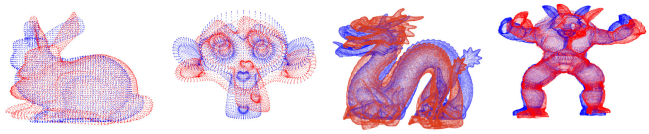


Fig. 4. Datasets used for numerical evaluations: bunny, monkey, dragon, and armadillo. Each point set colored red was constructed by deforming the point set colored blue in a non-linear manner.

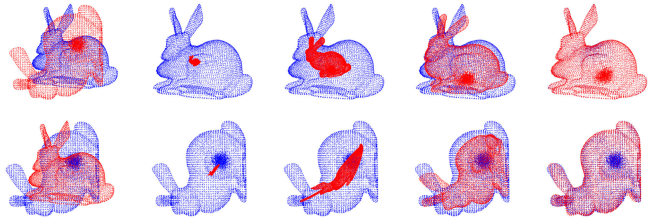


Fig. 5. Optimization trajectories for bunny data with rotation and clustered outliers. The points in the target and deformed shapes are colored blue and red, respectively. The leftmost column shows the initial point sets, and the optimization proceeds from left to right.

Fig. 6. Optimization trajectories for the armadillo data with rotation using the inverse multiquadric kernel. Even if the kernel is non-Gaussian, BCPD is scalable to point sets containing more than 100,000 points.

the approximation of the products involving K_{YX} can be computed in $O(M+N)$ time.

b) *For Small σ*. One issue of using the Nyström method is the inaccuracy that originates from the approximation, which often hampers convergence. To avoid the convergence issue without sacrificing computing time, we replace the Nyström method for computing P with the KD tree search method if σ^2 is sufficiently small. This approach is reasonable because, if the optimization approaches convergence, σ^2 is typically small and almost all elements in K_{YX} become nearly zero. The nonzero elements of K_{XY} can be found by the use of the KD tree search in $O((M+N)\log(M+N))$ time. BCPD with the use of the KD tree search often registers point sets as accurately as BCPD with exact computations while reducing the computational costs. The computational costs for the bottleneck computations are summarized in Table 1.

5 Experiments

In this section, we evaluate the registration performance of BCPD. In Section 5.1, we describe the datasets used for numerical evaluations. In Section 5.2, we evaluate the accuracy of non-rigid registration methods. In Section 5.3, we evaluate the registration performance of the accelerated BCPD. In Section 5.4, we apply BCPD to datasets with non-artificial deformations. In Section 5.5, we evaluate the accuracy of rigid registration methods.

5.1 Datasets and Demonstrations

We begin with the datasets used in Sections 5.2 and 5.3, shown in Fig. 4. We also provide several demonstrations.

Datasets. The first dataset is the bunny dataset, which is included in the CPD software. The dataset is composed of two bunny shapes: one is the same shape as that distributed in the 3D Scanning Repository (http://graphics.stanford.edu/data/3Dscanrep), and the other is deformed in a non-rigid manner. We downloaded the armadillo and dragon shapes from the 3D Scanning Library website, and we obtained the monkey shape from Blender. For each shape, we created the corresponding deformed shape using a non-rigid deformation tool. The numbers of points in the bunny, monkey, dragon, and armadillo data were 8,171, 7,958, 437,645, and 106,289, respectively. We note that the ground truth of corresponding points is known for the datasets, and thus exact registration errors can be evaluated.

Demonstrations. Figs. 5 and 6 show demonstrations of BCPD using the bunny and armadillo datasets with artificial disturbance. We used Gaussian and inverse multiquadric kernel functions for the bunny and armadillo datasets, respectively. Even when the kernel was non-Gaussian, BCPD successfully registered the armadillo dataset, which has more than 100,000 points. We summarized several demonstrations of BCPD in Supplementary Video 1, available online.

5.2 Comparison of Registration Accuracy

In this section, we report the registration accuracy regarding non-rigid registration methods: CPD [2], TPS-RPM [20], GMM-REG [21], and BCPD.

Generation of Artificial Disturbance. To evaluate robustness against rotation, outliers, and clustered outliers, we generated target point sets with artificial disturbance by repeating the following procedure:

- Five hundred points are randomly subsampled from both source and target point sets in the bunny, monkey, and dragon datasets so that the registration methods with no acceleration scheme can handle generated point sets.
- The subsampled target point sets are modified by one of the following operations: rotation, the addition of outliers that follow a uniform distribution, and the addition of clustered outliers that follow a Gaussian distribution.

We changed the angle of rotation, the number of outliers, and the number of clustered outliers as described hereafter.

a) *Rotation*. We changed the rotation angles from -120 to 120 degrees at intervals of five degree. The number of generated target point sets was 49 in total for each dataset.

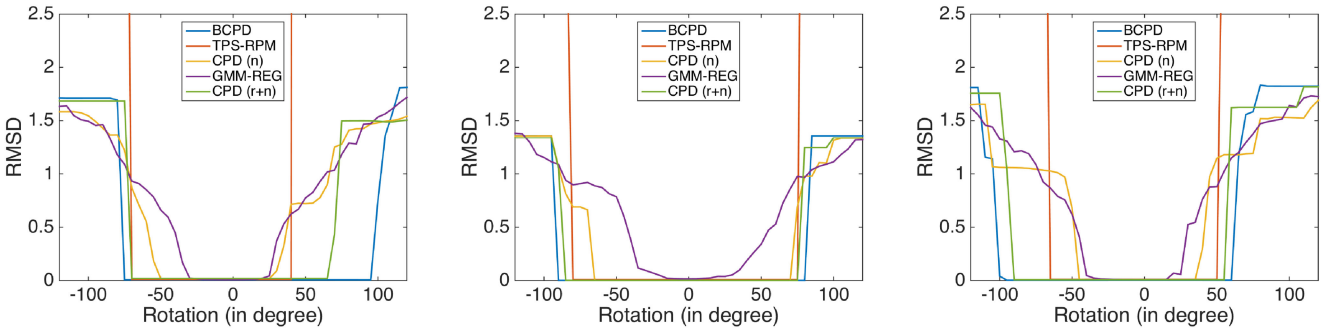


Fig. 7. Evaluation of robustness against the rotations of target point sets: bunny (left), monkey (middle), and dragon (right). The non-rigid CPD with or without the pre-alignment by the rigid CPD is denoted by "CPD (r+n)" or "CPD (n)," respectively.

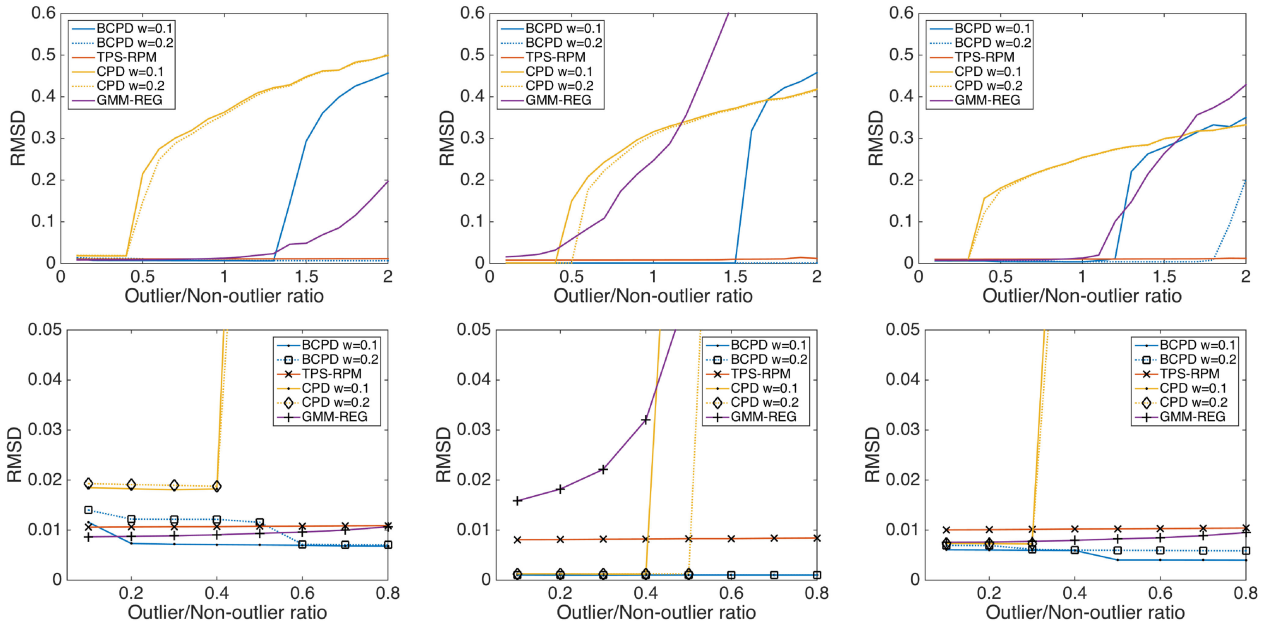


Fig. 8. Evaluation of robustness against outliers: bunny (left), monkey (middle), and dragon (right). The figures in the bottom are enlarged views of the top figures. The registration error of a method for each outlier/non-outlier ratio was measured by the median of RMSDs among 100 trials.

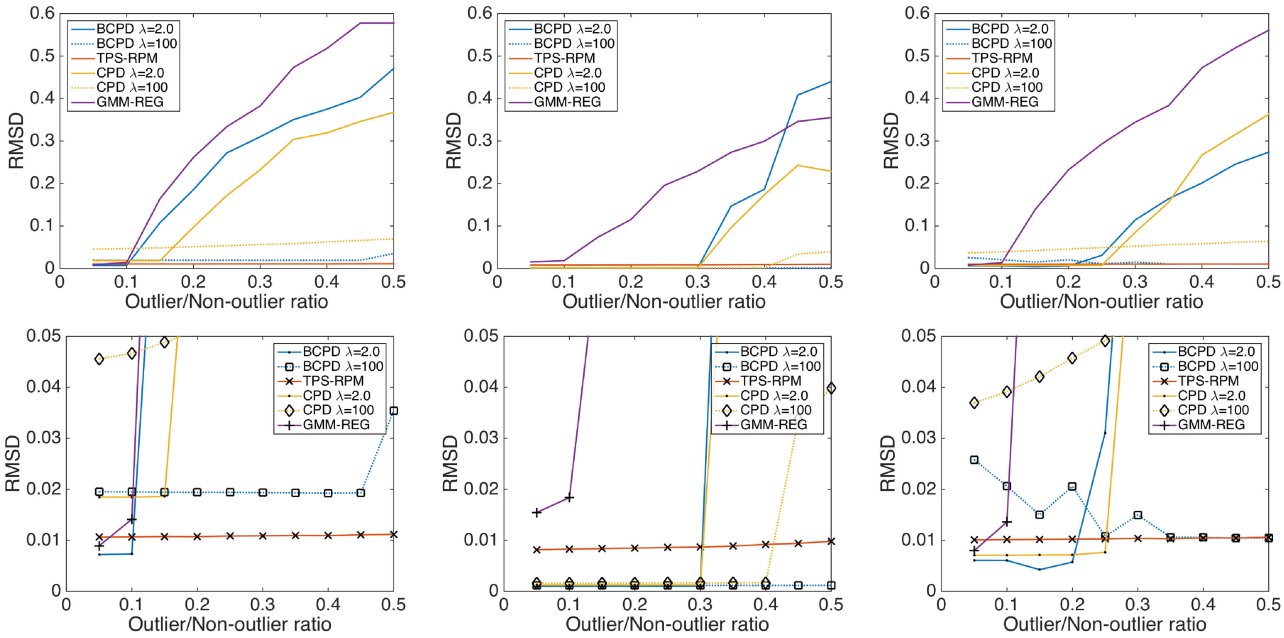


Fig. 9. Evaluation of robustness against clustered outliers: bunny (left), monkey (middle), and dragon (right). The figures in the bottom are enlarged views of the top figures. The registration error of a method for each outlier/non-outlier ratio was measured by the median of RMSDs among 100 trials.

b) *Addition of outliers*. We generated outliers that follow a uniform distribution with the bounding box surrounding a target point set. The length of an edge of the bounding box along the dth axis was set to $1.2l_d$, where $l_d = \max_n\{x_{nd}\}_{n=1}^N - \min_n\{x_{nd}\}_{n=1}^N$. We changed the number of outliers from 50 to 1,000 at intervals of 50, i.e., from 0.1 to 2.0 at intervals of 0.1 in outlier/non-outlier ratio. For each ratio, we repeated the generation 100 times. The number of generated target point sets was 2,000 in total for each dataset.

c) *Addition of clustered outliers*. We generated clustered outliers based on the isotropic Gaussian distribution with standard deviation 0.1 after normalizing a target point set. The center of the distribution was randomly selected as a point in the target point set. We changed the number of outliers from 25 to 250 at intervals of 25, i.e., from 0.05 to 0.5 at intervals of 0.05 in outlier/non-outlier ratio. For each ratio, we repeated the generation 100 times. The number of generated target point sets was 1,000 in total for each dataset.

Parameters. We used the Gaussian kernel and the shared parameters $(\beta, \lambda, \omega) = (2.0, 2.0, 0.1)$ for BCPD and CPD. To observe the effect of parameters, we added the results of $\omega = 0.2$ and $\lambda = 100$ for point sets with outliers and clustered outliers, respectively. The acceleration methods were disabled for both methods. The remaining parameters of BCPD, i.e., γ and κ, were set to 5.0 and infinity, respectively. The maximum number of iterations for BCPD was set to 500. The remaining CPD parameters were set to the following values:

$$\texttt{opt.method} = '\texttt{nonrigid}',\ \texttt{opt.max\_it} = 100,$$
$$\texttt{opt.normalize} = 1,\ \texttt{opt.tol} = \texttt{1e} - 10.$$

For GMM-REG, we used the following parameters:

$$\texttt{method} = '\texttt{TPS\_L2}',\ \texttt{normalize} = 1,\ \texttt{level} = 3,$$
$$\texttt{sigma} = [0.5, 0.2, 0.02],\ \texttt{lambda} = 0,$$
$$\texttt{max\_function\_evals} = [20, 20, 100].$$

For TPS-RPM, we used `T_finalfac=500`, `frac=1`, and `T_init=1.5`.

Evaluation. We used the root-mean-squared distance (RMSD) to measure registration errors. Except for the datasets with rotation, we computed the median RMSD among 100 trials because the generation of outliers and clustered outliers were dependent on random numbers.

Results. Here, we summarize the results of the evaluation.

a) *Robustness against rotation*. Fig. 7 shows the result of the performance evaluation regarding the robustness against target rotation. We added the evaluation of the non-rigid CPD after pre-alignment by the rigid CPD to compare BCPD with a naïve combination of non-rigid and rigid registrations. For all datasets, the range of angles within which BCPD correctly registered was the largest among the ranges within which the competitors did; BCPD outperformed the naïve combination of the rigid CPD and non-rigid CPD, suggesting an advantage of the simultaneous optimization of rigid and non-rigid transformations.

b) *Robustness against outliers*. Fig. 8 shows the result of the performance evaluation regarding the robustness against outliers that follow a uniform distribution. BCPD outperformed CPD for both $\omega = 0.1$ and $\omega = 0.2$ despite the similarity between the algorithms. This suggests that the difference between the outlier distributions of BCPD and CPD affected their performances. TPS-RPM was more robust against outliers than BCPD with $\omega = 0.1$; however, BCPD was comparable with TPS-RPM when $\omega = 0.2$.

c) *Robustness against clustered outliers*. Fig. 9 shows the result of the performance evaluation regarding the robustness against clustered outliers. The registration accuracy of BCPD and CPD were moderately similar for both $\lambda = 2.0$ and $\lambda = 100$ compared with case b). TPS-RPM was more robust against clustered outliers than BCPD with $\lambda = 2.0$; however, BCPD was comparable with TPS-RPM when $\lambda = 100$.

5.3 Performance of the Accelerated BCPD

In this section, we evaluate the registration performance of the accelerated BCPD to show that the acceleration maintains registration accuracy and reduces computing times.

5.3.1 Acceleration Parameters Versus Registration Accuracy

We evaluated the registration accuracy of BCPD for K and J, i.e., the parameters of the Nyström method for accelerating the computation of $\{G, \Sigma\}$ and P, respectively. Further, we evaluated the registration accuracy of the accelerated CPD algorithm for the comparison between them.

Dataset and Parameters. We used the bunny dataset, as shown in Fig. 4. We normalized the dataset before registration so that the mean and variance of the elements in a point set vector were 0 and 1, respectively. For both methods, we used the Gaussian kernel and $(\beta, \lambda, \omega) = (2, 2, 0)$. For BCPD, we used $(\gamma, \kappa) = (1, \infty)$, where $\gamma = 1$ and $\kappa \to \infty$ lead to the same assumption on the initial σ^2 and the mixing coefficients as for CPD. The maximum number of iterations was set to 500. We switched the Nyström method to the KD tree search with search radius $\min(0.15, 7\sigma)$ at $\sigma = 0.2$ only if we chose the KD tree search option. To accelerate CPD, we used the following parameters:

$$\texttt{opt.method} = '\texttt{nonrigid\_lowrank}', \texttt{opt.fgt} = 2,$$
$$\texttt{opt.eigfgt} = 1, \texttt{opt.viz} = 0, \texttt{opt.corresp} = 0,$$
$$\texttt{opt.normalize} = 1, \texttt{opt.tol} = \texttt{1e} - 10,$$
$$\texttt{opt.max\_it} = 100,$$

where the first three options specify non-rigid registration with the use of the fast Gauss transform, the eigendecomposition of G, and the computation of truncated Gaussian distributions.

Results. First, we investigated the influence of J, i.e., the number of Nyström points for accelerating P, on the registration performance. We accelerated the computations of G and Σ by setting $K = 70$ because the direct inversion of Σ was severely time-consuming and memory-inefficient. We summarized the result in Fig. 10; the second figure is an enlarged view of the first one. The RMSDs with the Nyström method decreased as J increased; however, the RMSD was relatively large even if $J = 600$, suggesting the need for more accurate computations near convergence. Even if the use of the KD tree search was allowed, it was not activated for $J = 50, 100$, and 150 because the corresponding σs were always greater than the cutoff, i.e., 0.2. For $J \geq 200$, the KD tree search was always activated. The corresponding RMSDs were very close

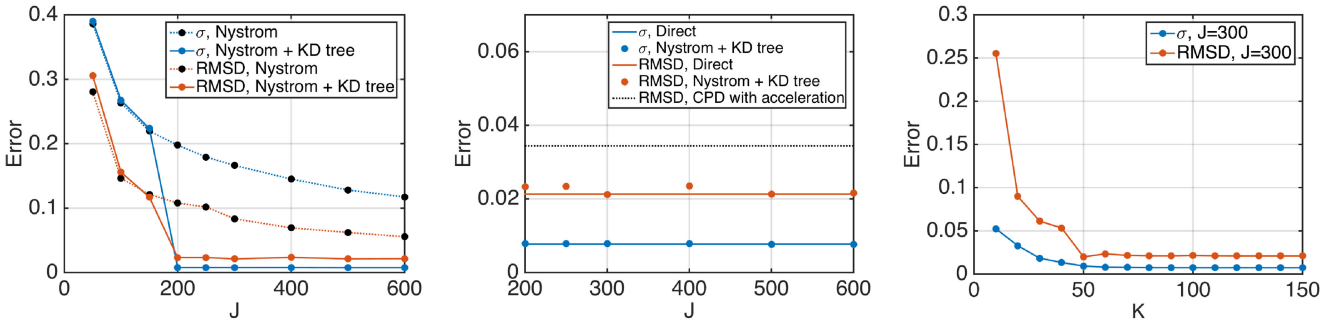


Fig. 10. Registration error versus the numbers of Nyström points, J and K, used for approximating P and G, respectively. The registration error was measured by the root-mean-squared distance (RMSD) using the bunny dataset. The symbol σ represents the square root of the residual variance after the optimization, i.e., an estimate of registration error. The second figure is an enlarged view of the first figure, and it additionally includes RMSDs computed by the BCPD that exactly calculated P and the accelerated CPD.

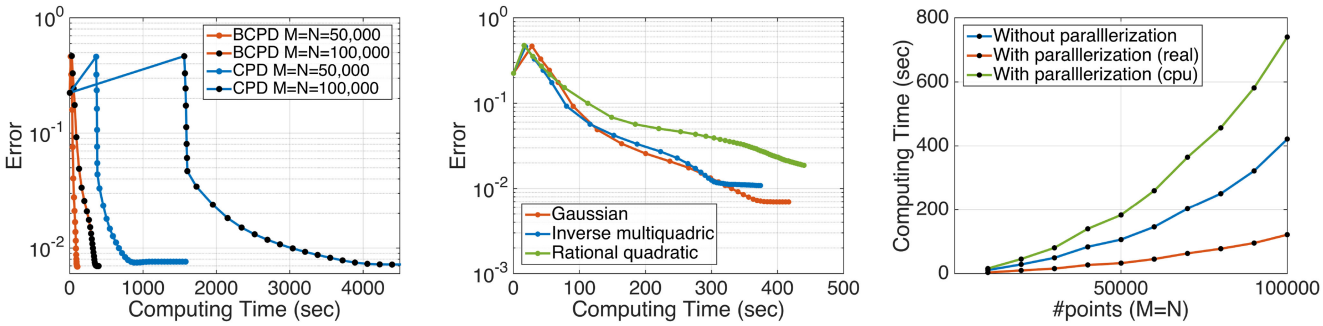


Fig. 11. Evolution of computing time and registration error (RMSD) for the armadillo dataset: CPD and BCPD (left) and the BCPD with different kernel functions (middle). A dot represents an odd number of optimization iterations. Right: Number of points vs. computing time until convergence with or without parallelization.

to the RMSD with the direct computation of P and were below 0.0344, the RMSD of the accelerated CPD. These results suggest that the accelerated calculation of P was sufficiently accurate for the bunny dataset.

The third figure in Fig. 10 shows the influence of K on registration performance. We calculated RMSDs in the same way as in the previous case; however, we fixed J at 300 and varied K from 10 to 150. The minimum RMSD was obtained at $K = 50$, suggesting that the approximation of G with $K = 50$ is sufficiently accurate for the bunny dataset.

5.3.2 Computing Time Versus Registration Accuracy

We evaluated the computing time and registration accuracy of BCPD and CPD; the latter was the only method scalable to point sets with 10^5 points among the competitors in Section 5.2. We used the implementation developed by Myronenko *et al.* because the faster implementations of CPD [59], [61] were not publicly available.

Dataset and Parameters. Using the armadillo dataset shown in Fig. 4, we generated target and source point sets by randomly extracting 10,000, 20,000, ..., 100,000 points. We set the parameters shared by BCPD and CPD to $(\beta, \lambda, \omega) = (3, 20, 0)$. For BCPD, we used the acceleration parameters $(J, K) = (300, 100)$, and we set the remaining parameters of BCPD and CPD to the same values as in Section 5.3.1.

Computational Environment. We used a MacBook Pro (15-inch Retina display, Early 2013, OS X El Capitan 10.11.6) with a 2.4 GHz Intel Core i7 CPU and 16 GB RAM as our computational environment. We implemented the BCPD algorithm in C and used GCC 6.0 as a compiler. We also implemented a parallelization option using OpenMP. We note that CPU time is roughly the same as wall-clock time if the parallelization is disabled.

Results. We first disabled the parallelization option implemented in our software for a fair comparison. The left panel in Fig. 11 shows the evolution of computing time and registration accuracy. BCPD achieved less computing time and slightly less registration error than CPD did. Especially, for the point sets containing 10^5 points, CPD repeated the E-step and M-step 100 times, i.e., the maximum number of iterations, without satisfying its convergence condition. The corresponding computing time and RMSD were 9389.8 s and 0.0079, respectively. BCPD repeated 62 cycles of VBI until convergence, and the corresponding computing time and RMSD were 398.0 s and 0.0070, respectively.

We then measured computing times of BCPD for different kernel functions with parallelization disabled. Using the point sets composed of 10^5 points, we compared Gaussian, inverse multiquadric, and rational quadratic kernels. All of the kernel functions are known to be positive definite [62]. The definitions of the kernel functions are listed in Table 2. The second panel of Fig. 11 shows the evolution of computing time and registration accuracy until convergence. As shown in the figure, the difference in computing times for the three kernels was considerably small. This suggests that our acceleration scheme can be applied even if the kernel function is non-Gaussian.

Finally, we activated the parallelization option of our software. We measured computing times of BCPD, changing the number of points in both target and source point

TABLE 2
Kernel Functions Used for the Evaluation of Computing Times

Kernel	$\mathcal{K}(y_m, y_{m'})$
Gaussian	$\exp\left(-\frac{1}{2\beta^2}\|\|y_m - y_{m'}\|\|^2\right)$
Inverse multiquadric	$\frac{1}{\sqrt{\|\|y_m - y_{m'}\|\|^2 + \beta^2}}$
Rational quadratic	$1 - \frac{\|\|y_m - y_{m'}\|\|^2}{\|\|y_m - y_{m'}\|\|^2 + \beta^2}$

sets from 10^4 to 10^5 at intervals of 10^4. The result is shown in the third panel of Fig. 11. CPU times increased owing to the multithreading overhead caused by parallelization; however, wall-clock times decreased by at least 50 percent. These results suggest that BCPD has an advantage over CPD in terms of computing time.

5.4 Real Data Examples

In this section, we report numerical experiments performed using the space-time faces [63] and SHREC'19 human-matching [64] datasets to demonstrate that BCPD can handle non-artificial deformations.

5.4.1 Performance Evaluation Using Space-Time Faces

We evaluate the registration performance of BCPD using a dataset of space-time faces, composed of a 3D+t sequence of facial expressions with the ground truth of corresponding points. The number of points in each face is 23,728. We used the ith and the $(i+1)$th faces as the source and target point sets for $i = 1, \ldots, 80$. An example of source and target point sets is shown in Fig. 12. We evaluated the registration accuracy of BCPD and CPD because GMM-REG and TPS-RPM were not scalable enough to register point sets in this dataset. We set the shared parameters of CPD and BCPD to the same values, i.e., $(\beta, \lambda, \omega) = (0.3, 10^4, 0)$, and the remaining parameters of BCPD were set to $(\gamma, \kappa) = (0.1, \infty)$. The acceleration parameters of CPD were set to the same values as those in Section 5.3.1, whereas those of BCPD were set to $(J, K) = (300, 150)$, and the Nyström method was switched to the KD tree search with search radius $\min(0.15, 7\sigma)$ if $\sigma < 0.2$. Registration errors were measured using the RMSD from the ground truth. The right panel of Fig. 12 shows the boxplot of RMSDs after taking the logarithm. The RMSDs of BCPD were smaller than those of CPD, although the difference in RMSDs was quite small.

5.4.2 Application to SHREC'19 Data

Here, we provide an example of the "shape transfer" as an application of BCPD. The dataset was taken from a SHREC'19 track, called "matching humans with different connectivity" [64]. Among 44 human body shapes, we used shape no. 1 as a target point set and no. 42 as a source point set, as shown in Fig. 13. After resampling 10,000 points for both point sets using voxel grid filtering, we applied the BCPD with the Gaussian kernel; we used the registration parameters $(\lambda, \beta, \omega, \gamma, \kappa) = (2, 2, 0, 10, \infty)$ and the same acceleration parameters as those in Section 5.3.1. Shape (c) in the figure shows shape no. 42 after the registration, clearly approaching shape no. 1.

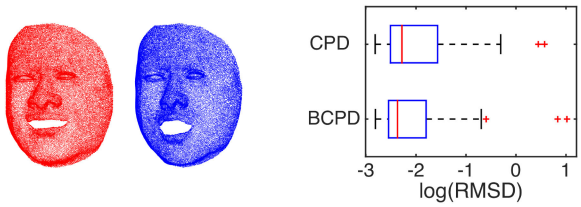


Fig. 12. Performance evaluation using the datasets of space-time face. An example of target and source point sets (left) and the registration accuracy of CPD and BCPD for 80 pairs of point sets in the dataset (right).

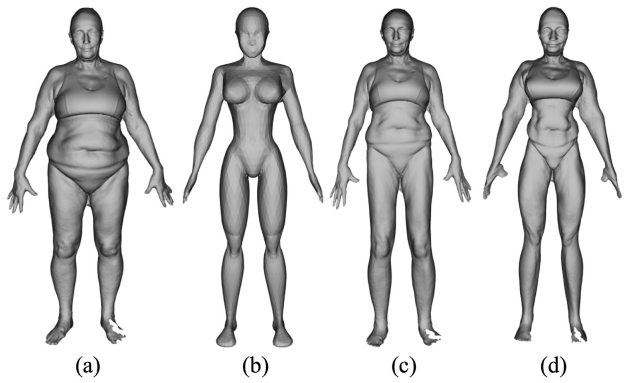


Fig. 13. "Shape transfer" using BCPD. (a) Source shape taken from SHREC'19 dataset. (b) Target shape taken from SHREC'19 dataset. (c) Shape after the first registration. (d) Shape after the second registration.

The rightmost shape of Fig. 13 shows the shape after the second registration; we applied BCPD to shapes (b) and (c), where we set (c) as the source shape for registration. We used the registration parameters $(\lambda, \beta, \omega, \gamma, \kappa) = (2, 1.2, 0, 0.1, \infty)$; that is, we relaxed the motion coherence compared with the first registration, and we initialized σ^2 so that the initial matching was a "moderately hard" one. The deformed shape became closer to shape no.1 as a whole, although some parts of the body, e.g., the hands and arms, deformed unnaturally. This suggests that a tightly fit shape can be obtained by the second registration with weak motion coherence and a small γ.

As described above, we applied BCPD twice with different registration parameters. This was because the single execution of the BCPD algorithm failed to transfer shape (a) to shape (b) under the weak motion coherence corresponding to the second registration. This suggests that running the algorithm twice contributed to preventing a local optimum that would have resulted in an incorrect registration.

5.5 Rigid Registration

BCPD can be applied to rigid registration problems owing to the similarity transformation incorporated into its transformation model, as shown in Fig. 14. We evaluated the performance of rigid registrations for BCPD by comparing it with the methods implementing rigid registration techniques: CPD [2], GMM-REG [21], and Go-ICP [14].

Datasets. The first dataset we used was the ASL dataset [65]. Among the eight types of datasets, we used four types of data: Apartment, Stairs, Mountain Plain, and Wood in Summer. The second dataset we used was the set of 3D surfaces with partial overlaps [66] included in the UWA dataset.

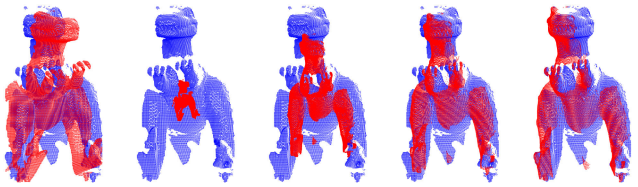


Fig. 14. BCPD solves rigid registration problems under an appropriate set of parameters owing to the similarity transformation incorporated into its transformation model.

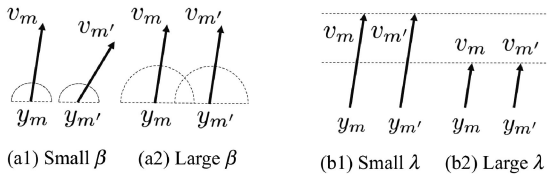


Fig. 15. Comparisons of rigid registrations using the ASL data (left) and the UWA data (right). The y-axis represents the rate of correct registrations using 40 pairs of point sets. A trial is defined as being a success if the angular difference from the ground truth is less than a value specified by the x-axis. For the UWA dataset, the ground truth was defined based on the results of Go-ICP, owing to its almost perfect performance finding partial overlaps in the dataset.

We used all four objects in the dataset: Chef, Parasaurolophus, T. Rex, and Chicken. We note that these datasets were acquired with different sensors, having been acquired using the Hokuyo UTM-30LX Scanning Rangefinder and the Konica Minolta Vivid 910 3D Scanner.

Pre-Processing. To reduce computing time for CPD and GMM-REG, we decreased the number of points in each point set in both datasets by approximately 2,000 using the voxel grid filter implemented in GMM-REG. For CPD and GMM-REG, we used the internal normalization methods implemented in each software. For BCPD, we used the same normalization method as CPD. For Go-ICP, we reduced the number of points in each target point set by 1,000 using its internal downsampling scheme; we normalized the point sets to be inside $[-1, 1]^3$ by following its manual, because the internal normalization scheme was not available in the software.

Parameters. For BCPD, we used the parameters $(\lambda, \omega, \gamma, \kappa) = (10^9, 0.1, 10, \infty)$; we used a large λ because BCPD becomes a rigid registration technique if λ goes to infinity. To accelerate BCPD, we used the Nyström method with $(J, K) = (300, 70)$ and switched it to the KD tree search with search radius $\min(0.3, 7\sigma)$ if $\sigma \leq 0.3$. We used the Gaussian kernel with $\beta = 2.0$ to run the BCPD algorithm, although the non-rigid deformation v was almost ignorable owing to the large λ. For CPD, we used the following parameters:

$$\texttt{opt.omega} = 0.1, \texttt{opt.method} = \texttt{'rigid'},$$
$$\texttt{opt.tol} = \texttt{1e} - 10, \texttt{opt.normalize} = 1,$$
$$\texttt{opt.fgt} = 1, \texttt{opt.max\_it} = 100.$$

For GMM-REG, we used the parameters

$$\texttt{level} = 2, \texttt{sigma} = [0.5, 0.2], \texttt{lambda} = 0,$$
$$\texttt{max\_function\_evals} = [20, 20].$$

For Go-ICP, the parameter domain to explore was set to be $[-\pi, \pi]^3 \times [-0.5, 0.5]^3$. The trimming rate and the convergence threshold were set to be 0.1 and 0.001, respectively.

Evaluation. We registered the ith and the $(i+1)$th point sets for $i = 1, \ldots, 10$ for each of the eight datasets because all pairs of the point sets were partially overlapped. We evaluated the goodness of a rigid registration by the angle error against the ground truth. The angular difference in degree between rotation matrices R_1 and R_2 in $\mathbb{R}^3$, denoted by ϵ, is computed by the following formula: $\epsilon = \frac{180}{\pi} \arccos(\frac{1}{2}\{\mathrm{Tr}\ (R_1 R_2^T) - 1\})$. The registration accuracy of a method was defined as the ratio of correct registrations to 40 registration problems. A trial was defined as being a success if the angular difference from the ground truth was less than a cutoff value. We changed the cutoff values from 0.25 to 5.0 at intervals of 0.25.

Results. Fig. 15 shows the results for the ASL dataset (left) and the UWA dataset (right). For the ASL dataset, BCPD outperformed the competitors if the cutoff value to define the successful registration was less than one degree. The accuracy of GMM-REG was 1.0 if the cutoff was greater than three degrees. The registration accuracy for BCPD and CPD was clearly different despite the similarity of their algorithms, suggesting that the difference in their acceleration schemes and outlier distributions affected their registration performance. For the UWA dataset, we observed that the registration performance of Go-ICP was almost perfect. Therefore, we used the result of Go-ICP as the ground truth to compare the remaining methods. The right panel in Fig. 15 shows the results for the UWA dataset. BCPD outperformed CPD and GMM-REG, suggesting that the registration performance of BCPD was the closest to that of Go-ICP for the UWA dataset.

6 Conclusion

In this paper, we formulated CPD in a Bayesian setting; we introduced motion coherence using the prior distribution of displacement vectors rather than using the motion coherence theory. We also derived BCPD registration algorithm, which is interpreted as a generalization of the CPD algorithm, using VBI. The Bayesian formulation provided a clear difference between tuning parameters controlling the motion coherence. We also proposed an acceleration scheme that can be applied to non-Gaussian kernels.

In the numerical studies, we showed that our acceleration method successfully reduced computing time without losing registration accuracy. We compared BCPD with well-known registration methods for both non-rigid and rigid registration problems. The results showed that BCPD was at least comparable to the competitors for all experiments, suggesting its usefulness as a general-purpose registration technique.

Acknowledgments

The author would like to deeply thank the anonymous referees who provided helpful, constructive, and detailed comments on earlier versions of the manuscript. This work was supported by JSPS KAKENHI Grant Number 17K12712.

REFERENCES

[1] A. Myronenko, X. Song, and M. Á. Carreira-Perpiñán, "Non-rigid point set registration: Coherent point drift," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2006, pp. 1009–1016.

[2] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2262–2275, Dec. 2010.

[3] A. L. Yuille and N. M. Grzywacz, "A mathematical analysis of the motion coherence theory," *Int. J. Comput. Vis.*, vol. 3, pp. 155–175, 1989.

[4] B. Maiseli, Y. Gu, and H. Gao, "Recent developments and trends in point set registration methods," *J. Vis. Commun. Image Representation*, vol. 46, pp. 95–106, 2017.

[5] O. Hirose, "Dependent landmark drift: Robust point set registration with a Gaussian mixture model and a statistical shape model," pp. 1–25, 2017, *arXiv 1711.06588v3*.

[6] L. G. Brown, "A survey of image registration techniques," *ACM Comput. Surv.*, vol. 24, no. 4, pp. 325–376, 1992.

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

[8] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proc. Int. Conf. 3-D Digit. Imag. Model.*, 2001, pp. 145–152.

[9] S. Granger and X. Pennec, "Multi-scale EM-ICP: A fast and robust approach for surface registration," in *Proc. Eur. Conf. Comput. Vis.*, 2002, pp. 418–432.

[10] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image Vis. Comput.*, vol. 21, no. 14, pp. 1145–1153, 2003.

[11] N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas, "Registration of point cloud data from a geometric optimization perspective," in *Proc. Eurograph./ACM SIGGRAPH Symp. Geometry Process.*, 2004, pp. 22–31.

[12] A. Makadia, A. Patterson, and K. Daniilidis, "Fully automatic registration of 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 1297–1304.

[13] Y. Khoo and A. Kapoor, "Non-iterative rigid 2D/3D point-set registration using semidefinite programming," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 2956–2970, Jul. 2016.

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

[15] V. Golyanik, B. Taetz, and D. Stricker, "Joint pre-alignment and robust rigid point set registration," in *Proc. IEEE Int. Conf. Image Process.*, 2016, pp. 4503–4507.

[16] J. Vongkulbhisal, F. De LaTorre, and J. P. Costeira, "Discriminative optimization: Theory and applications to point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3975–3983.

[17] J. Vongkulbhisal, B. I. Ugalde, F. De la Torre, and J. P. Costeira, "Inverse composition discriminative optimization for point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2993–3001.

[18] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "3D object recognition in cluttered scenes with local surface features: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2270–2287, Nov. 2014.

[19] H. Chui and A. Rangarajan, "A feature registration framework using mixture models," in *Proc. IEEE Workshop Math. Methods Biomed. Image Anal.*, 2000, pp. 190–197.

[20] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Comput. Vis. Image Understanding*, vol. 89, no. 2/3, pp. 114–141, 2003.

[21] B. Jian and B. C. Vemuri, "Robust point set registration using Gaussian mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1633–1645, Aug. 2011.

[22] J. Chen, J. Ma, C. Yang, L. Ma, and S. Zheng, "Non-rigid point set registration via coherent spatial mapping," *Signal Process.*, vol. 106, pp. 62–72, 2015.

[23] Y. Yang, S. H. Ong, and K. W. C. Foong, "A robust global and local mixture distance based non-rigid point set registration," *Pattern Recognit.*, vol. 48, no. 1, pp. 156–173, 2015.

[24] I. Kolesov, J. Lee, G. Sharp, P. Vela, and A. Tannenbaum, "A stochastic approach to diffeomorphic point set registration with landmark constraints," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 238–251, Feb. 2016.

[25] J. Ma, J. Zhao, and A. L. Yuille, "Non-rigid point set registration by preserving global and local structures," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 53–64, Jan. 2016.

[26] V. Golyanik, B. Taetz, G. Reis, and D. Stricker, "Extended coherent point drift algorithm with correspondence priors and optimal subsampling," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2016, pp. 1–9.

[27] J. Ma, J. Wu, J. Zhao, J. Jiang, H. Zhou, and Q. Z. Sheng, "Nonrigid point set registration with robust transformation learning under manifold regularization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3584–3597, Dec. 2019.

[28] B. Allen, B. Curless, and Z. Popović, "The space of human body shapes: Reconstruction and parameterization from range scans," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 587–594, 2003.

[29] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 399–405, 2004.

[30] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid ICP Algorithms for surface registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[31] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 80:1–80:7, 2007.

[32] H. Li, R. W. Sumner, and M. Pauly, "Global correspondence optimization for non-rigid registration of depth scans," *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1421–1430, 2008.

[33] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. J. Comput. Vis.*, vol. 13, no. 2, pp. 119–152, 1994.

[34] C. V. Stewart, C. L. Tsai, and B. Roysam, "The dual-bootstrap iterative closest point algorithm with application to retinal image registration," *IEEE Trans. Med. Imag.*, vol. 22, no. 11, pp. 1379–1394, Nov. 2003.

[35] A. Rangarajan, H. Chui, and F. L. Bookstein, "The softassign procrustes matching algorithm," *Lecture Notes Comput. Sci.*, vol. 1230, pp. 29–42, 1997.

[36] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust Euclidean alignment of 3D point sets: The trimmed iterative closest point algorithm," *Image Vis. Comput.*, vol. 23, no. 3, pp. 299–309, 2005.

[37] Y. Tsin and T. Kanade, "A correlation-based approach to robust point set registration," in *Proc. Eur. Conf. Comput. Vis.*, 2004, pp. 558–569.

[38] D. A. Hirshberg, M. Loper, E. Rachlin, and M. J. Black, "Coregistration: Simultaneous alignment and modeling of articulated 3D shape," *Lecture Notes Comput. Sci.*, vol. 7577, pp. 242–255, 2012.

[39] Q. Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 766–782.

[40] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *Proc. 27th Annu. Conf. Comput. Graph. Interactive Techn.*, 2000, pp. 157–164.

[41] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, 2005.

[42] D. Cosker, E. Krumhuber, and A. Hilton, "A FACS valid 3D dynamic action unit database with applications to 3D dynamic morphable facial modeling," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 2296–2303.

[43] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, "Learning a model of facial shape and expression from 4D scans," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 194:1–194:17, 2017.

[44] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: Modeling and capturing hands and bodies together," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 245:1–245:17, 2017.

[45] V. Kraevoy and A. Sheffer, "Template-based mesh completion," in *Proc. Eurograph. Symp. Geometry Process.*, 2005, pp. 13–22.

[46] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "SCAPE: Shape completion and animation of people," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 408–416, 2005.

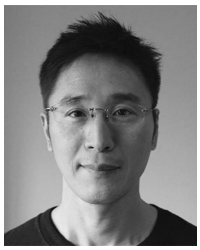
[47] D. Mateus, R. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer, "Articulated shape matching using Laplacian eigenfunctions and unsupervised point registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[48] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.-P. Seidel, "A statistical model of human pose and body shape," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 337–346, 2009.

[49] R. Horaud, F. Forbes, M. Yguel, G. Dewaele, and J. Zhang, "Rigid and articulated point registration with expectation conditional maximization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 587–602, Mar. 2011.

[50] F. Bogo, J. Romero, M. Loper, and M. J. Black, "FAUST: Dataset and evaluation for 3D mesh registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3794–3801.

[51] S. Ge and G. Fan, "Non-rigid articulated point set registration with local structure preservation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 126–133.
[52] S. Ge and G. Fan, "Non-rigid articulated point set registration for human pose estimation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2015, pp. 94–101.
[53] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.
[54] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural network architectures," *Neural Comput.*, vol. 7, pp. 219–269, 1995.
[55] G. Wahba, *Spline Models for Observational Data*. Philadelphia, PA, USA: SIAM, 1990.
[56] C. K. I. Williams and M. W. Seeger, "Using the Nyström method to speed up kernel machines," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2001, pp. 682–688.
[57] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.
[58] L. Greengard and J. Strain, "The fast Gauss transform," *SIAM J. Sci. Statist. Comput.*, vol. 12, no. 1, pp. 79–94, 1991.
[59] J. Dupej, V. Krajíček, and J. Pelikán, "Low-rank matrix approximations for coherent point drift," *Pattern Recognit. Lett.*, vol. 52, pp. 53–58, 2014.
[60] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis, "Improved fast Gauss transform and efficient kernel density estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2003, pp. 664–671.
[61] M. Lu, J. Zhao, Y. Guo, and Y. Ma, "Accelerated coherent point drift for automatic three-dimensional point cloud registration," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 2, pp. 162–166, Feb. 2016.
[62] M. Genton, "Classes of kernels for machine learning: A statistics perspective," *J. Mach. Learn. Res.*, vol. 2, pp. 299–312, 2001.
[63] L. Zhang, N. Snavely, B. Curless, and S. M. Seitz, "Spacetime faces: High resolution capture for modeling and animation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 548–558, 2004.
[64] S. Melzi, R. Marin, E. Rodolà, and U. Castellani, 2019. [Online]. Available: http://profs.scienze.univr.it/marin/shrec19/
[65] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms," *Int. J. Robot. Res.*, vol. 31, no. 14, pp. 1705–1711, 2012.
[66] A. S. Mian, M. Bennamoun, and R. Owens, "Three-dimensional model-based object recognition and segmentation in cluttered scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1584–1601, Oct. 2006.



Osamu Hirose (Member, IEEE) is currently an assistant professor with the Institute of Science and Engineering, Kanazawa University. His recent research interests focus on developing machine learning techniques for automating the analysis of digital images and movies, especially those related to biology. His research interests also include 3D shape modeling and registration.