Pattern Recognition 79 (2018) 452–466

Contents lists available at ScienceDirect

Pattern Recognition

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

Shape registration with directional data

Mairéad Grogan*, Rozenn Dahyot

School of Computer Science and Statistic, Trinity College Dublin, Ireland

ARTICLE INFO

Article history:
Received 29 August 2017
Revised 10 January 2018
Accepted 18 February 2018
Available online 19 February 2018

Keywords:
Shape registration
Directional information
Von Mises-Fisher
$\mathcal{L}_2$ registration

ABSTRACT

We propose several cost functions for registration of shapes encoded with Euclidean and/or non-Euclidean information (unit vectors). Our framework is assessed for estimation of both rigid and non-rigid transformations between the target and model shapes corresponding to 2D contours and 3D surfaces. The experimental results obtained confirm that using the combination of a point's position and unit normal vector in a cost function can enhance the registration results compared to state of the art methods.

1. Introduction

Directions, axes or rotations are described as unit vectors in $\mathbb{R}^d$ and are known collectively as directional data. In computer vision this type of data is often processed and includes surface normals and tangent vectors, orientations of image gradients, the direction of sound sources and GPS coordinate information [1,2]. Directional data can be viewed as points on the surface of a hypersphere $\mathbb{S}^d$, with angular directions observed in the real world frequently visualized on the circle or sphere. A lot of research has been concerned with successfully modelling and analysing this type of data, with distributions proposed by von Mises, Fisher and Watson [3] used in a range of applications including data clustering, segmentation and texture mapping [4].

In this paper we propose to use von Mises-Fisher kernels to model the normal vectors of the shape (i.e. 2D contours, and 3D surface). Registration is then performed by minimizing a distance between two Kernel Density Estimates encoding the target and model shapes. Section 2 reviews the related work and in Section 3, we propose several new cost functions for registration using normal information. Section 4 outlines some of the implementation details of our method and experimental results (Section 5) compare our approach to leading techniques for registration [5–7].

2. Related works

2.1. Euclidean distance between GMMs

Considering $p_1(x)$ and $p_2(x)$, two probability density functions (pdf) for the random vector $x \in \mathbb{R}^d$, the Euclidean $\mathcal{L}_2$ distance between p_1 and p_2 is defined as:

$$\mathcal{L}_2(p_1, p_2) = \int [p_1(x) - p_2(x)]^2 \, dx = \|p_1 - p_2\|^2 \tag{1}$$

Many divergences have been defined for p.d.f. [8] but $\mathcal{L}_2$ has the advantage of being explicit in the case of Gaussian Mixtures Models (GMM) and also robust to outliers when these GMMs are kernel density estimates (KDE) with Gaussian Kernels [5,9]. $\mathcal{L}_2$ can be rewritten as:

$$\mathcal{L}_2(p_1, p_2) = \|p_1 - p_2\|^2 = \|p_1\|^2 + \|p_2\|^2 - 2\langle p_1 | p_2 \rangle \tag{2}$$

and simplified to

$$\mathcal{L}_2 E(p_1, p_2) = \|p_1\|^2 - 2\langle p_1 | p_2 \rangle \tag{3}$$

when p_2 is chosen as the empirical p.d.f. (i.e. KDE with Dirac Kernels) fitted on a target point set [5,9]. Jian and Vemuri applied $\mathcal{L}_2$ for shape registration in $\mathbb{R}^2$ and $\mathbb{R}^3$, where a parameterized GMM $p_1(x|\theta)$ is fitted to a target shape model represented by GMM p_2 [5]: θ controls an affine or non rigid transformation (e.g. Thin Plate Spline) and the solution $\hat{\theta}$ is computed such that it minimizes the $\mathcal{L}_2$ distance. While Jian and Vemuri encode shapes as GMMs fitted to point clouds of contours ($\mathbb{R}^2$) or surfaces ($\mathbb{R}^3$), Arellano and Dahyot extended this shape registration approach into a multiple instance shape (ellipses) detection scheme, and also use directional information about the contour [10]. Similarly, in this paper we propose to consider directional information and we consider specific

* Corresponding author.
E-mail addresses: mgrogan@tcd.ie (M. Grogan), Rozenn.Dahyot@tcd.ie (R. Dahyot).

https://doi.org/10.1016/j.patcog.2018.02.021
0031-3203/

kernels suited for directional data as opposed to the Gaussian Kernel used by Arellano and Dahyot. Alternatively, Fan et al. propose to use GMMs to model the convex hull of 3D shapes, and minimise the $\mathcal{L}_2$ distance between these distributions to register 3D shapes [11]. Another application for $\mathcal{L}_2$ registration in the image domain is to compute the colour transfer function to change the colour feel of images and videos [12, 13, 14, 31]. In this context, $\mathcal{L}_2$ registration is shown to outperform other popular schemes including these designed on the optimal transport framework [14].

Other registration techniques include maximum likelihood techniques, in which one point set is represented by a GMM and the other by a mixture of delta functions, which is equivalent to minimising the KL divergence between the two mixtures. In [6], Myronenko and Song also impose that the GMM centroids move coherently, preserving the structure of the point clouds. The Iterative Closest Point algorithm is another popular registration technique which alternates between estimating closest-point correspondences and a rigid transformation. However, it can become trapped in local minima and requires a good initialisation, and many methods have been proposed as improvements [8,15]. Ying et al. propose to extend the ICP algorithm so that it can account for scale differences between 3D shapes using an SVD decomposition approach [16] and later estimate affine and non-linear transformations by applying a Lie group parametrization method to globally align the shapes and avoid non-degenerate solutions [17,18]. Yang et al. [7] propose to combine ICP with a branch-and-bound scheme which efficiently searches the rigid 3D motion space. They also derive novel upper and lower bounds for the ICP error function and provide a globally optimal solution to the 3D rigid registration problem. Du et al. propose to make the ICP algorithm more robust by using a Gaussian model when computing the distance between the point sets, reducing the effect of noise on the results [19]. Rather than using fuzzy correspondences [5,6] other techniques estimate explicit correspondences between the point sets using shape descriptors such as shape context [20] and the local shape and neighbourhood structure of the shapes [21,22]. Yang et al. [23] propose to combine global and local structural differences in a global and local mixture distance (GLMD) based method for non-rigid registration. Their iterative two step process alternately estimates the correspondences and computes the transformation.

2.2. Directional data

We consider the d-dimensional unit random vector u such that $\|u\| = 1$ ($u \in \mathbb{S}^{d-1}$ with $\mathbb{S}^{d-1}$ the hypersphere in $\mathbb{R}^d$). Several distributions exist for random unit vectors and some are presented in this section. Applications of such distributions include RGB-D image segmentation [4] and structure from motion in 360 video [24] amongst others [1,2].

2.2.1. Von Mises-Fisher kernel

The von Mises-Fisher distribution is one of the most commonly used distributions for directional data and has properties similar to those of a multivariate Gaussian in $\mathbb{R}^d$. The von Mises-Fisher probability density function for a random unit vector $u \in \mathbb{S}^{d-1}$ is defined as:

$$vMF(u;\mu,\kappa) = C_d(\kappa)\ \exp\left(\kappa\ \mu^T u\right) \tag{4}$$

with parameters $\kappa \geq 0$ and $\|\mu\| = 1$ and the normalising constant C_d is defined as:

$$C_d(\kappa) = \frac{1}{\int_{\mathbb{S}^{d-1}} \exp\left(\kappa\ \mu^T u\right)\ du} = \frac{\kappa^{\frac{d}{2}-1}}{(2\pi)^{\frac{d}{2}}\ \mathcal{I}_{\frac{d}{2}-1}(\kappa)} \tag{5}$$

with $\mathcal{I}_{\frac{d}{2}-1}$ the modified Bessel function of order $\frac{d}{2}-1$. The von Mises-Fisher distribution is parametrised by the mean vector μ and the concentration parameter κ, so called because it determines how strongly the distribution is concentrated about the mean vector. The von Mises-Fisher distribution is rotationally symmetric about μ, with high values of κ creating a distribution highly concentrated about μ, and low values of κ creating an almost uniform distribution on $\mathbb{S}^{d-1}$. The von Mises-Fisher distribution with parameters κ and μ is noted $vMF(\mu, \kappa)$ for simplification. For dimension $d = 3$, u is a unit vector in $\mathbb{R}^3$ and belongs to the sphere $\mathbb{S}^2$, and the normalising constant in the von Mises-Fisher distribution is [4]:

$$C_3(\kappa) = \frac{\kappa}{4\pi\ \sinh(\kappa)} \tag{6}$$

For $d \neq 3$, the value $C_d(\kappa)$ is not directly available but can be computed using numerical integration.

2.2.2. Watson distribution

The Watson distribution is also used to model axially symmetric directional data and is defined as follows:

$$W_d(u;\mu,\kappa) = M_d(\kappa)\exp\left(\kappa\ (\mu^T u)^2\right) \tag{7}$$

with the normalising constant:

$$M_d(\kappa) = \frac{1}{\int_{\mathbb{S}^{d-1}} \exp\left(\kappa\ (\mu^T u)^2\right)\ du} \tag{8}$$

This can be computed as $M_d(\kappa) = M(\frac{1}{2}, \frac{d}{2}, \kappa)$, the confluent hypergeometric function also known as the Kummer function, which is not directly available but can be approximated. Like the von Mises-Fisher distribution, the Watson distribution is also rotationally symmetric about μ and the value of κ determines the shape of the distribution. For a geometric description of the Watson distribution see [4, 31].

3. $\mathcal{L}_2$ with directional data

Given two sets of observations $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\cdots,n_1}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,\cdots,n_2}$ for the random vectors $(x,u) \in \mathbb{R}^{d_x} \times \mathbb{S}^{d_u}$, we encode the model and target shapes using KDEs and register them by minimising the $\mathcal{L}_2$ distance between them. Here we assume that the shapes differ by some transformation ϕ, controlled by the parameter θ, which registers S_1 to S_2 and creates a new shape with observations $\tilde{S}_1 = \{(\tilde{x}_1^{(i)}, \tilde{u}_1^{(i)})\}_{i=1,\cdots,n_1}$ that maps to S_2.

Probability density functions are modelled using sets $\tilde{S}_1$ and S_2 providing two possible distributions denoted p_1 and p_2 for the random vector $x \in \mathbb{R}^{d_x}$, $u \in \mathbb{S}^{d_u}$, and $(x,u) \in \mathbb{R}^{d_x} \times \mathbb{S}^{d_u}$. The $\mathcal{L}_2$ distance between p_1 and p_2 can then be computed as $\mathcal{L}_2(p_1,p_2) = \|p_1 - p_2\|^2 = \|p_1\|^2 + \|p_2\|^2 - 2\langle p_1|p_2\rangle$ which is equivalent to maximizing the scalar product $\langle p_1|p_2\rangle$ when ϕ is a rigid mapping [5].

3.1. Pdf modelling for $x \in \mathbb{R}^{d_x}$

Jian and Vemuri used a KDE with a Gaussian kernel $\mathcal{N}(x;\tilde{x}_1^{(i)}, h)$ fitted to each point $\tilde{x}_1^{(i)}$ in $\tilde{S}_1$ [5]:

$$p_1(x) = \frac{1}{n_1}\sum_{i=1}^{n_1} \mathcal{N}(x;\tilde{x}_1^{(i)}, h_1) \tag{9}$$

Likewise the second set of points $\{x_2^{(j)}\}$ is modelled using the Gaussian kernels:

$$p_2(x) = \frac{1}{n_2}\sum_{j=1}^{n_2} \mathcal{N}(x;x_2^{(i)}, h_2) \tag{10}$$

All Gaussians are chosen spherical in shape and have uniform scale and using the scalar product between two Gaussian kernels (cf. Table A.4), the following cost function is proposed to estimate θ:

$$\mathcal{C}^x = \frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1}\frac{1}{\sqrt{4h_1^2\pi}}\exp\left(\frac{-\|\tilde{x}_1^{(j)}-\tilde{x}_1^{(i)}\|^2}{4h_1^2}\right) - \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1}\frac{1}{\sqrt{2(h_1^2+h_2^2)\pi}}\exp\left(\frac{-\|x_2^{(j)}-\tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right) \quad (11)$$

This cost function computes $\|p_1\|^2 - 2\langle p_1|p_2\rangle$, and minimising $\mathcal{C}^x$ is equivalent to minimising the $\mathcal{L}_2$ distance between the GMMs p_1 and p_2, and was previously proposed for shape registration by Jian and Vemuri [5]. Note that if we had instead fitted a Dirac kernel to each of the points in $\{x_2^{(j)}\}$ we would have obtained the following cost function:

$$\mathcal{C}_\delta^x = \frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1}\frac{1}{\sqrt{4h_1^2\pi}}\exp\left(\frac{-\|\tilde{x}_1^{(j)}-\tilde{x}_1^{(i)}\|^2}{4h_1^2}\right) - \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1}\frac{1}{\sqrt{2h_1^2\pi}}\exp\left(\frac{-\|x_2^{(j)}-\tilde{x}_1^{(i)}\|^2}{2h_1^2}\right) \quad (12)$$

which is equivalent to $\mathcal{C}^x$ when $h_2 = 0$.

3.2. Pdf modelling for $u \in \mathbb{S}^{d_u}$

To model the first set of normals $\{u_1^{(i)}\}$ we propose a KDE with a von Mises-Fisher kernel $vMF(u;\tilde{u}_1^{(i)},\kappa)$ fitted to each normal $\tilde{u}_1^{(i)}$ in $\tilde{S}_1$:

$$p_1(u) = \frac{1}{n_1}\sum_{i=1}^{n_1} vMF(u;\tilde{u}_1^{(i)},\kappa_1) \quad (13)$$

We propose to model the second set of normal vectors $\{u_2^{(j)}\}$ using either the empirical distribution:

$$p_2(u) = \frac{1}{n_2}\sum_{j=1}^{n_2}\delta(u-u_2^{(j)}) \quad (14)$$

or using the von Mises-Fisher distribution:

$$p_2(u) = \frac{1}{n_2}\sum_{j=1}^{n_2} vMF(u;u_2^{(j)},\kappa_2). \quad (15)$$

Using the definitions for $\langle p_1|p_2\rangle$ as given in Table A.4, two cost functions used to estimate θ by minimising $\|p_1\|^2 - 2\langle p_1|p_2\rangle$ can then be defined as follows:

$$\mathcal{C}_\delta^u = \left(\frac{C_{d_u}(\kappa_1)}{n_1}\right)^2\sum_{i=1}^{n_1}\sum_{j=1}^{n_1}C_{d_u}^{-1}\left(\|\kappa_1\tilde{u}_1^{(i)}+\kappa_1\tilde{u}_1^{(j)}\|\right) - \frac{2C_{d_u}(\kappa_1)}{n_1 n_2}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\exp(\kappa_1\tilde{u}_1^{(i)T}u_2^{(j)}), \quad (16)$$

based on the modelling for p_2 defined in Eq. (14), and

$$\mathcal{C}^u = \left(\frac{C_{d_u}(\kappa_1)}{n_1}\right)^2\sum_{i=1}^{n_1}\sum_{j=1}^{n_1}C_{d_u}^{-1}\left(\|\kappa_1\tilde{u}_1^{(i)}+\kappa_1\tilde{u}_1^{(j)}\|\right) - \frac{2\,C_{d_u}(\kappa_1)C_{d_u}(\kappa_2)}{n_1 n_2}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}C_{d_u}^{-1}\left(\|\kappa_1\tilde{u}_1^{(i)}+\kappa_2 u_2^{(j)}\|\right) \quad (17)$$

based on the modelling for p_2 defined in Eq. (15).

Since both terms in $\mathcal{C}^u$ depend on the normalising constant $C_{d_u}(\kappa)$, the computation of $\mathcal{C}^u$ requires numerical integration when $d_u \neq 3$. On the other hand, when ϕ_θ is a rigid transformation $\mathcal{C}_\delta^u$ can be simplified as

$$\widehat{\theta} = \arg\max_\theta\left\{\mathcal{C}_\delta^u = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\exp(\kappa_1\tilde{u}_1^{(i)T}u_2^{(j)})\right\} \quad (18)$$

and therefore can be easily computed $\forall d_u$. This is one of the main advantages of using the Dirac distribution to model one set of normal vectors.

3.3. Pdf modelling for $(x,u) \in \mathbb{R}^{d_x}\times\mathbb{S}^{d_u}$

We investigate in this section a cost function which accounts for both the normal vectors and point positions of the shapes in the modelling. For the transformed observations a KDE with Gaussian kernels fitted to each point $\tilde{x}_1^{(i)}$ and a *vMF* kernel fitted to each normal vector $\tilde{u}_1^{(i)}$ is modelled as follows:

$$p_1(x,u) = \frac{1}{n_1}\sum_{i=1}^{n_1} vMF(u;\tilde{u}_1^{(i)},\kappa_1)\ \mathcal{N}(x;\tilde{x}_1^{(i)},h_1) \quad (19)$$

For the second set of observations we again propose two methods for modelling the point and normal vectors. First, we propose to fit a dirac Delta kernel to each normal vector $u_2^{(j)}$ and a Gaussian kernel to each point $x_2^{(j)}$ to create a KDE of the form:

$$p_2(x,u) = \frac{1}{n_2}\sum_{j=1}^{n_2}\delta(u-u_2^{(j)})\ \mathcal{N}(x;x_2^{(j)},h_2). \quad (20)$$

We also propose an alternate KDE with vMF kernels fitted to the normal vectors $\{u_2^{(j)}\}$ as in Eq. (19):

$$p_2(x,u) = \frac{1}{n_2}\sum_{j=1}^{n_2} vMF(u;u_2^{(j)},\kappa_2)\ \mathcal{N}(x;x_2^{(j)},h_2). \quad (21)$$

Then the parameter θ is estimated by minimizing one of the following cost functions:

$$\mathcal{C}_\delta^{x,u} = \frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1}\langle vMF(\tilde{u}_1^{(i)},\kappa_1)|vMF(\tilde{u}_1^{(j)},\kappa_1)\rangle \times\langle\mathcal{N}(\tilde{x}_1^{(i)},h_1)|\mathcal{N}(\tilde{x}_1^{(j)},h_1)\rangle - \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1}\langle vMF(\tilde{u}_1^{(i)},\kappa_1)|\delta(u_2^{(j)})\rangle \times\langle\mathcal{N}(\tilde{x}_1^{(i)},h_1)|\mathcal{N}(x_2^{(j)},h_2)\rangle. \quad (22)$$

based on the modelling proposed in Eq. (20), or

$$\mathcal{C}^{x,u} = \frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1}\langle vMF(\tilde{u}_1^{(i)},\kappa_1)|vMF(\tilde{u}_1^{(j)},\kappa_1)\rangle \times\langle\mathcal{N}(\tilde{x}_1^{(i)},h_1)|\mathcal{N}(\tilde{x}_1^{(j)},h_1)\rangle - \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1}\langle vMF(\tilde{u}_1^{(i)},\kappa_1)|vMF(u_2^{(j)},\kappa_2)\rangle \times\langle\mathcal{N}(\tilde{x}_1^{(i)},h_1)|\mathcal{N}(x_2^{(j)},h_2)\rangle. \quad (23)$$

based on the modelling proposed in Eq. (21). Using the appropriate scalar product definitions given in Table A.4, the proposed cost functions can be written explicitly as:

$$\mathcal{C}_\delta^{x,u} = \frac{C_d(\kappa_1)C_d(\kappa_1)}{n_1 n_1\sqrt{4h_1^2\pi}}\times\sum_{i=1}^{n_1}\sum_{j=1}^{n_1}C_d^{-1}\left(\|\kappa_1\tilde{u}_1^{(i)}+\kappa_1\tilde{u}_1^{(j)}\|\right) \times\exp\left(\frac{-\|\tilde{x}_1^{(j)}-\tilde{x}_1^{(i)}\|^2}{4h_1^2}\right) - \frac{2C_d(\kappa_1)}{n_1 n_2\sqrt{2(h_1^2+h_2^2)\pi}}\times\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\exp(\kappa_1\ {u_2^{(j)}}^T\tilde{u}_1^{(i)}) \times\exp\left(\frac{-\|x_2^{(j)}-\tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right). \quad (24)$$

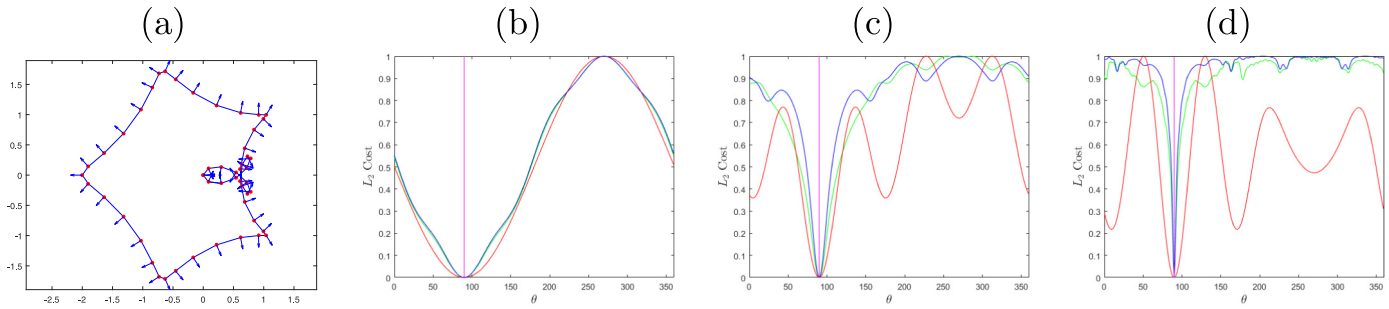


Fig. 1. In (a) the parametric curve (blue) sampled at 50 locations (red) with normal vectors (shown in blue) is S_1, and was rotated by an angle of $\theta_{GT} = 90^\circ$ to generate S_2. In (b), (c) and (d) we show the effect that our simulated annealing strategy has on the cost functions computed using S_1 and S_2 as θ ranges from 1° to 360°. To avoid local minima, h is gradually decreased and κ is gradually increased until $h = h_{final}$ and $\kappa = \kappa_{final}$. Green: $\mathcal{C}^x$; Red: $\mathcal{C}^u$; Blue: $\mathcal{C}^{x,u}$; Pink: θ_{GT}. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and

$$\mathcal{C}^{x,u} = \frac{C_{d_u}(\kappa_1)C_{d_u}(\kappa_1)}{n_1 n_1 \sqrt{4h_1^2\pi}} \times \sum_{i=1}^{n_1}\sum_{j=1}^{n_1} C_{d_u}^{-1}\left(\|\kappa_1 \tilde{u}_1^{(i)} + \kappa_1 \tilde{u}_1^{(j)}\|\right) \times \exp\left(\frac{-\|\tilde{x}_1^{(j)} - \tilde{x}_1^{(i)}\|^2}{2h_1^2}\right) - 2\frac{C_{d_u}(\kappa_1)C_{d_u}(\kappa_2)}{n_1 n_2 \sqrt{2(h_1^2+h_2^2)\pi}} \times \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} C_{d_u}^{-1}\left(\|\kappa_1 \tilde{u}_1^{(i)} + \kappa_2 u_2^{(j)}\|\right) \times \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right). \quad (25)$$

Although both terms in $\mathcal{C}^{x,u}$ depend on the normalizing constant $C_{d_u}(\kappa)$, in $\mathcal{C}_\delta^{x,u}$ the term $\langle p_1|p_2\rangle$ is independent of this constant and can be computed for any dimension d_u. Therefore when ϕ is a rigid transformation, θ can be estimated for any dimension d_u by maximizing the cost function:

$$\mathcal{C}_\delta^{x,u} = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \exp(\kappa_1\ {u_2^{(j)}}^T \tilde{u}_1^{(i)}) \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right) \quad (26)$$

4. Algorithm and analysis

In this section we outline some of the implementation details of our algorithm when it was applied to registering two shapes S_1 and S_2 with point sets $\{x_1^{(i)}\}$ and $\{x_2^{(j)}\}$ and unit normal vectors $\{u_1^{(i)}\}$ and $\{u_2^{(j)}\}$ respectively.

4.1. Transformation function ϕ

To test the proposed cost functions, we considered shapes differing by both a rigid and non-rigid transformation ϕ. As a translation only affects the observations $\{x^{(i)}\}$ and not the normal vectors $\{u^{(i)}\}$, the cost functions $\mathcal{C}^u$ and $\mathcal{C}_\delta^u$ are invariant to translation. To ensure all cost functions are evaluated equally, when estimating a rigid transformation we omit a translation and only consider data differing by a rotation (Fig. 1).

For shapes differing by a non-rigid deformation, we estimate a Thin Plate Spline transformation and do not include a regularisation term to control the non-linearity of the transformation. However this can be added by the user if necessary. The N control points c_j used to control the TPS transformations are chosen uniformly on a grid spanning the bounding box of the model shape.

4.2. Algorithm

Given two point sets $\{x_1^{(i)}\}_{i=1,\dots n_1}$ and $\{x_2^{(j)}\}_{j=1,\dots n_2}$ representing the model and target shapes, our strategy for estimating the transformation $\phi(x, \theta)$ is summarised in Algorithm 1.

Algorithm 1 Our strategy for estimating the transformation $\phi(x, \theta)$.

Require: $\hat{\theta}$ initialised so that $\phi(x, \hat{\theta}) = x$ (identity function)
Require: κ_{init}, κ_{final} and h_{init}, h_{final} for $\mathcal{C}^{x,u}$.
Require: h_{step}, κ_{step}
Require: Computation of unit normal vectors $\{u_1^{(i)}\}_{i=1,\dots n_1}$ and $\{u_2^{(j)}\}_{j=1,\dots n_2}$ from $\{x_1^{(i)}\}_{i=1,\dots n_1}$ and $\{x_2^{(j)}\}_{j=1,\dots n_2}$.
Choose m points $\{x_1^{(i)}\}_{i=1,\dots m}$ and $\{x_2^{(j)}\}_{j=1,\dots m}$ and their associated unit normal vectors $\{u_1^{(i)}\}_{i=1,\dots m}$ and $\{u_2^{(j)}\}_{j=1,\dots m}$ for processing.
Start $h = h_{init}$ and $\kappa = \kappa_{init}$
repeat
  $\hat{\theta} \leftarrow \arg\min_\theta \mathcal{C}(\theta)$
  $h \leftarrow h_{step} \times h$
  $\kappa \leftarrow \kappa_{step} \times \kappa$
until Convergence $h < h_{final}$ and $\kappa > \kappa_{final}$ **return** $\hat{\theta}$

In all of our experiments we let $h_1 = h_2 = h$ and $\kappa_1 = \kappa_2 = \kappa$. To avoid local minima, we implement a simulated annealing strategy by gradually decreasing h and increasing κ. The values chosen for all parameters can be found in the supplementary material. To reduce computation time we do not process all points in the shapes S_1 and S_2, but instead sample m points and their associated unit normal vectors from both.

4.3. Normal vector computation

We use several methods to compute the normal vectors $\{u^{(i)}\}$ at the points $\{x^{(i)}\}$. When testing our cost functions on 2D data, we use parametric curves and compute the normal vectors analytically. For 3D shapes in the form of meshes, we compute the normal vectors at a given vertex $x^{(i)}$ as the average of the normal vectors of each face connected to $x^{(i)}$. We also compute the normal vectors without exploiting the connectivity of a vertex, instead fitting a plane to it's nearest neighbours to compute the normal vector [25,26].

4.4. Computation complexity

The computational complexity of all cost functions depends on the number of points n_1 and n_2 in the shapes, and is of order $\mathcal{O}(n_1 \times n_2)$. Choosing n point correspondences reduces this to $\mathcal{O}(n)$. The computation time needed by the gradient ascent technique depends on the dimension of the latent space, which is determined by the transformation being estimated and the dimension d_x of the space in which the shapes are defined. We do not provide analytical gradients to the gradient ascent algorithm when testing any of the proposed cost functions, and instead use numerical methods. While analytical gradients can be computed for $\mathcal{C}^x$ [5], and for all cost function when estimating a rotation, computing gradients for $\mathcal{C}^{x,u}$ or $\mathcal{C}^{x,u}_\delta$ when estimating a TPS transformation is not trivial, and we found that using numerical approximation was preferable.

4.5. Correspondences

In some cases, when estimating a non-rigid transformation, correspondences are used to reduce computational complexity and improve the registration result. When n correspondences are chosen, the double sum $\sum_i^{n_1}\sum_j^{n_2}$ in all cost functions is reduced to $\sum_i^n$. To compute correspondences we used the method proposed by Yang et al. [23]. First, a global distance between the point sets is computed followed by a local distance measuring the difference in neighbourhood structure. The local and global distances are combined to estimate a set of point correspondences. Yang et al. also incorporate an annealing scheme which is designed to slowly change the cost minimisation from local to global, which we incorporate into our simulated annealing strategy.

4.6. Comparisons

To evaluate our algorithm we compared our results to several techniques [5–7,23]. The parameters used for each are given in the supplementary material. To ensure that a fair comparison between Jian and Vemuri's cost function $\mathcal{C}^x$ and our proposed cost functions was presented, we altered some of the optimisation steps in the code provided by Jian and Vemuri[1] so that they coincided with those implemented with our proposed cost functions eg. the same simulated annealing framework was used for $\mathcal{C}^x$, $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}_\delta$ and $\mathcal{C}^{x,u}$ and the changes made enhanced the results achievable by $\mathcal{C}^x$.

4.7. Evaluation

In all experiments we chose the model and target point sets to be of equal size with $n_1 = n_2 = n$. The ground truth point correspondences between the model and target shapes $\{x_1^{(i)}, x_2^{(i)}\}_{i=1,\dots n}$ are also known and we evaluate the results of all algorithms by computing the mean square error (MSE) between corresponding points in the transformed model ($\{\tilde{x}_1^{(i)}\}$) and target ($\{x_2^{(i)}\}$) point sets. Note that all n points in the target shape and transformed model are used to compute the MSE, not just the subsample of size m used to estimate $\phi(x, \hat{\theta})$.

4.8. Experimental set up

Our cost functions are evaluated experimentally (c.f. Sections 5 and 6) with the following settings:

- For rigid transformation, the scalar product $\langle p_1 | p_2 \rangle$ is the cost function that is maximized as $\|p_1\|$ does not change in this case (In Sections 5.1 and 6.1).
- For non-rigid transformation, $\|p_1\|^2 - 2\langle p_1 | p_2 \rangle$ is minimized to estimate θ (c.f. Sections 5.2 and 6.2).

In Section 6.3 we give details about the computational cost of our algorithm.

5. Experimental results 2D

When considering the von Mises kernel as part of our cost functions, because its normalizing constant $C_3(\kappa)$ is explicitly available for $u \in \mathbb{S}^2$ while $C_2(\kappa)$ is not for 2D data ($u \in \mathbb{S}$), we propose to artificially define u on $\mathbb{S}^2$ instead of $\mathbb{S}$ in this case by adding a third dimensional coordinate to the normal vector (which is set to zero) to ease and speed up computation.

5.1. 2D rotation registration

Curves S_1 and S_2 differ by a rotation ϕ which is defined as $\phi(x, \theta) = \mathrm{R}x$ (and $\phi(u, \theta) = \mathrm{R}u$ for the normal vector $u \in \mathbb{S}$, ignoring the zero value added to artificially extend $u \in \mathbb{S}^2$), where R is a 2D rotation matrix controlled by the angle θ. We assess the estimation of the rotation angle θ using the cost functions $\mathcal{C}^x$ [5], $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$.

When testing our results we found that $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ as well as $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ are practically equivalent, so for ease of comparison we only present results for $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$. Results for $\mathcal{C}^u_\delta$ and $\mathcal{C}^{x,u}_\delta$ can be found in the appendix.

1. Fig. 2(a) presents the average MSE errors computed for $\mathcal{C}^x$ [5], $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ when considering rotation transformation between the two shapes to be registered. It shows that $\mathcal{C}^{x,u}$ performs the best, followed by $\mathcal{C}^u$ and then $\mathcal{C}^x$. Several values of θ were tested (reported in abscissa Fig. 2(a)), and for each value we created and registered 10 pairs of curves S_1 and S_2.
2. Fig. 2(b) presents the average MSE errors computed for $\mathcal{C}^x$ [5], $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ when estimating a rotation but with missing data between the two shapes to be registered. Again $\mathcal{C}^{x,u}$ performs the best, followed by $\mathcal{C}^u$ and then $\mathcal{C}^x$. In this experiment a parametric curve S_1 is represented by 150 vertices $\{x_1^{(i)}\}_{i=1,\dots 150}$ with their corresponding normal vectors $\{u_1^{(i)}\}_{i=1,\dots 150}$ to create $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\dots 150}$. S_2 is created by rotating S_1 (with $\theta = 60^\circ$) and a percentage of points (reported in abscissa Fig. 2(b)) are removed from S_2 before registration. For each percentage of points removed (7%, 20%, 33%, 47%, 60%) we generating 10 pairs of curves S_1 and S_2 on which we tested the estimation of θ. As the number of removed points increases to 60% or more, $\mathcal{C}^{x,u}$ had a higher tendency to fall into local minima and the error increases as a result. Similar results were found for other values of θ tested.

5.2. 2D non-rigid registration

Shapes S_1 and S_2 differ now by a non-rigid deformation (defined as a TPS transformation with varying degrees of deformation). The estimated TPS transformation is controlled by $N = 12$ control points and our latent space of parameters to estimate is of dimension $(12 \times 2) + 6 = 30$. Cost functions $\mathcal{C}^x$ [5] and $\mathcal{C}^{x,u}$ are assessed, and $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ are omitted as normal information alone is not sufficient when estimating a non-rigid transformation. $\mathcal{C}^{x,u}_\delta$ generates similar results to $\mathcal{C}^{x,u}$ and is not reported here, but can be found in the appendix. As well as comparing $\mathcal{C}^x$ [5] and $\mathcal{C}^{x,u}$, we also compare to other state of the art non-rigid registration techniques namely CPD [6] and GLMD [23]. For comparison we implement a similar experimental framework as that presented by Yang et al.[23] and for this reason we also normalize all curves so that they lie within $[0, 1] \times [0, 1]$.

[1] https://github.com/bing-jian/gmmreg.

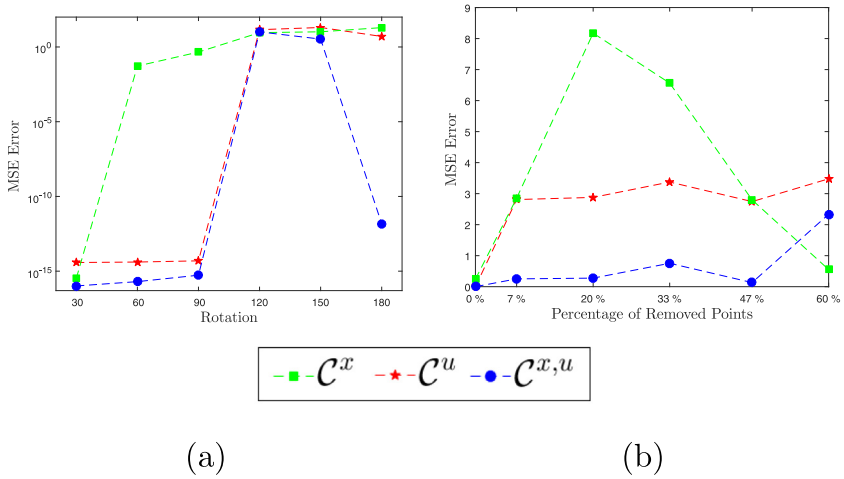


Fig. 2. MSE results comparing our cost functions on 2D data with rigid transformation (rotation). In (a) the MSE value given at each rotation is the average over 10 curve registration results, as is the MSE value given at each percentage of removed points in (b).

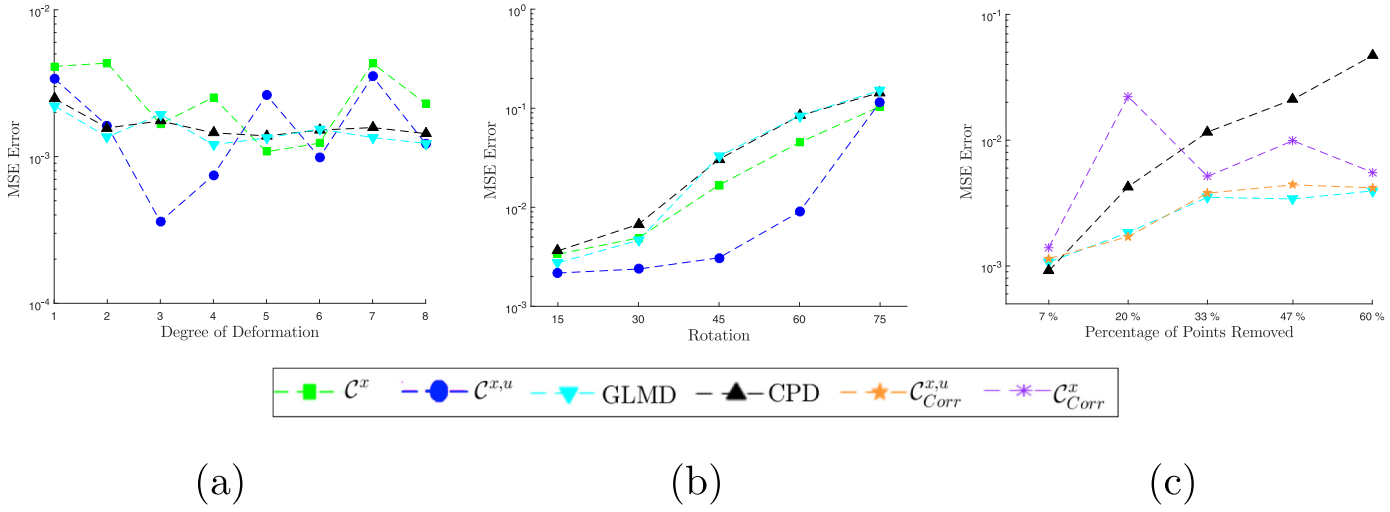


Fig. 3. MSE results for non-rigid registration with 2D data. (a) Deformation estimation results with degree of deformation varying from 1 to 8; (b) Deformation and rotation estimation, with degree of deformation 4 and rotation varying from 15° to 75°; (c) Deformation estimation with missing data.

1. For each level of non rigid deformation we register 120 pairs of shapes S_1 and S_2 and present the average MSE results in Fig. 3(a). We found that in general $\mathcal{C}^{x,u}$ performs well, but fails on occasion skewing the average MSE (i.e. at deformations of degree 5 and 7). Similar spikes appear in the results for $\mathcal{C}^x$[5]. Both CPD[6] and GLMD[23] generate consistent results over all deformations.
2. Setting the degree of deformation to 4, rotations of $\pm 15°$, $\pm 30°$, $\pm 45°$, $\pm 60°$ and $\pm 75°$ are added so that both rotation and non rigid parameters now need to be estimated to register S_1 and S_2. At each rotation value we registered 240 pairs of deformed curves for each method and the mean square errors computed can be seen in Fig. 3 (b). $\mathcal{C}^{x,u}$ performs best, followed by $\mathcal{C}^x$, GLMD[23] and CPD[6]. The addition of the normal information in the cost function ensured that in general $\mathcal{C}^{x,u}$ estimated the correct rotation and deformation, while in the case of the other cost functions, the non-rigid deformation parameters were often used to attempt to account for the rotation difference.
3. Setting the degree of deformation to 4 without rotation, a percentage of points is randomly removed from S_1 before registration on S_2. Fig. 3(c) shows that $\mathcal{C}^{x,u}_{corr}$ performed as well as GLMD[23], followed by $\mathcal{C}^x_{corr}$ and CPD[6]. Without correspondences we found that both $\mathcal{C}^{x,u}$ and $\mathcal{C}^x$ (not reported) tried to maximize the amount of overlap between the curves and rarely estimated the correct parameters. For this experiment correspondences were estimated using Yang et al.s technique, as described in Section 4.5, and were used when optimizing $\mathcal{C}^{x,u}_{corr}$ and $\mathcal{C}^x_{corr}$. 120 pairs of curves were registered at each level of missing data (reported in abscissa) and the average MSE is reported in Fig. 3(c).

6. Experimental results in 3D

6.1. 3D rotation registration

We now consider two 3D shapes $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\cdots,n}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,\cdots,n}$ which are represented by their point locations $\{x^{(i)}\} \in \mathbb{R}^3$ and normal vectors $\{u^{(i)}\} \in \mathbb{S}^2$. Shapes S_1 and S_2 differ by a rotation ϕ which is defined as $\phi(x, \theta) = \mathrm{R}x$ with $\theta = \mathrm{R}$. In this case our latent space is of dimension 9. We compared our results to those obtained using Jian and Vemuri's method $\mathcal{C}^x$ [5], CPD [6] and Go-ICP [7]. The shapes used in this experiment are the Stanford Bunny, Dragon and Buddha meshes[2], and the Horse mesh provided by Sumner and Popovic [27]. Each shape has both vertex and edge information available, from which normal vectors are

[2] http://graphics.stanford.edu/data/3Dscanrep/.

easily calculated (Section 4.3). A sub-sample of vertices and their corresponding normal vectors are used in all of our experiments.

When testing our results we found that $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ as well as $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ are practically equivalent, so for ease of comparison we only present results for $\mathcal{C}^u_\delta$ and $\mathcal{C}^{x,u}_\delta$ in the following section. Further comparisons with $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ can be found in the appendix.

The MSE errors for the following experiments are presented in Fig. 4.

1. The first column of Fig. 4 presents the results of rigid registration when target and model meshes have the same sampling w.r.t. different levels of rotation magnitude (reported in abscissa). At each level of rotation magnitude, 15 different pairs of shapes S_1 and S_2 were registered from which MSE is calculated. Correspondences are not used to enhance the registration process in this case and overall CPD performs best, followed by Go-ICP and $\mathcal{C}^u_\delta$, while $\mathcal{C}^{x,u}_\delta$ and Jian and Vemuri's method $\mathcal{C}^x$ seem to generate similar results. Since there is a one to one correspondence between the samples from each shape, all methods perform very well with an average MSE of around 10^{-34} for CPD and 10^{-12} in all other cases. Both CPD and Go-ICP have a tendency to fall into local minima as the rotation increases while $\mathcal{C}^u_\delta$, $\mathcal{C}^x_\delta$ and $\mathcal{C}^{x,u}_\delta$ continue to estimate good solutions.
2. Similarly the second experiment (reported in the second column of Fig. 4) considers the case where target and model meshes do not have the same set of vertices but instead different samples of 1000 points were chosen from S_1 and S_2, along with their corresponding normal vectors, so that no one to one correspondence exist between the subsampled target and model shapes. $\mathcal{C}^{x,u}_\delta$ performs the best in this case, followed by Jian and Vemuri's method $\mathcal{C}^x$. Here $\mathcal{C}^u_\delta$ performed the worst as unlike vertices, normal vectors represent the first derivative of the surface and are more sensitive to noise, thus varying more when they are not sampled at exactly the same locations on S_1 and S_2. Again both CPD and Go-ICP fall into local solutions as the rotation magnitude increases. Fig. 5 shows a sample of the results achieved by the algorithms when registering the bunny mesh.
3. Our cost functions are assessed when noise is present in the data (third column of Fig. 4)) with three levels of Gaussian noise (mean zero and standard deviation varying from 0.001 to 0.003 reported in abscissa) applied to vertices of S_2, which differs from S_1 by a rotation of magnitude 30°. When computing the normal vectors of the noisy shape S_2 we used the nearest neighbours approach implemented by Meshlab, as described in Section 4.3. We found that when noise is present in the point positions, this gives a better estimate of the normal vector than using the vertex connectivity. As the noise on the points $\{x_2^{(i)}\}$ increases we also increase the number of nearest neighbours (N_k) used to compute the normal vectors, for example we set $N_k = 40, 60$ and 120 for noise levels 0.001, 0.002 and 0.003 respectively when registering two Bunny shapes. The normal vectors associated with the noise free shape S_1 were computed using the vertex and edge information provided in the .ply. Different samples of 1000 points, along with their associated normal vectors, were then chosen from S_1 and the noisy S_2, so that no one to one point correspondences exist between the target and model point clouds. The registration process was repeated 15 times for each noise level, and in all cases, $\mathcal{C}^{x,u}_\delta$ performs the best, followed by $\mathcal{C}^x$, CPD and Go-ICP. The additional smoothed normal vector information used by $\mathcal{C}^{x,u}_\delta$ allows it to converge to a more accurate solution, even when a large degree of noise is added to the points in the shape S_1. Again $\mathcal{C}^u_\delta$ does not perform as well as the other approaches. Fig. 5 shows a sample of the results achieved by the algorithms when registering the bunny mesh.

6.2. 3D non-rigid registration

Finally we consider two 3D shapes S_1 and S_2 differ by a non-rigid deformation. We register these shapes by estimating a non-rigid TPS transformation. We choose the number of control points as $N = 125$ so our latent space has $(125 \times 3) + 12 = 387$ dimensions. Point correspondences are used here in the cost functions $\mathcal{C}^x$ and $\mathcal{C}^{x,u}$, notated as $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$. Again we omit $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ as they did not perform well when estimating a non-rigid transformation. We also omit $\mathcal{C}^{x,u}_\delta$ as it has previously been shown to perform similarly to $\mathcal{C}^{x,u}$. Additional graphical results can also be found in the supplementary material.

We present two sets of experiments in this section. In the first set we compare how $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ perform when registering shapes with known correspondences and in the second we compare $\mathcal{C}^x_{corr}$, $\mathcal{C}^{x,u}_{corr}$, CPD[6] and GLMD[23] when registering shapes with unknown correspondences that must be estimated.

1. In this first experiment we use the dataset of shapes provided by Sumner and Popovic [27] containing meshes of several different types of animal in different poses, including a cat, lion and horse. Each mesh of the same animal has an equal number of vertices and exact point correspondences. We use the ground truth point correspondences when computing $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ to reduce computational complexity. Choosing two meshes of the same type of animal, we let the vertices of each mesh be the points $\{x_1^{(i)}\}$ and $\{x_2^{(i)}\}$ and compute the corresponding normal vectors $\{u_1^{(i)}\}$ and $\{u_2^{(i)}\}$ using the edge information provided in the mesh. We then apply a rotation to S_1 so that the shapes differ by both a rotation and non-rigid deformation. For each level of rotation tested we register 10 pairs of shapes S_1 and S_2. Fig. 6(a) reports MSE comparing $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$: due to the large dimension of the latent space (387 dimensions), the gradient ascent technique required a large number of iterations to register the shape S_1 to S_2. For each cost function, to reduce computation time the limit of on the number of function evaluations computed during optimization is set to 50,000 (at each simulated annealing step). Very little difference is observed between the cost functions and even with 50,000 functions evaluations, both $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ failed to converge to a good solution.
2. In this experiment a scan taken of the Stanford Bunny with 1000 points is used to generate S_1 and S_2. Taking the points of the scan to be $\{x_2^{(i)}\}$, we computed the normals vectors $\{u_2^{(i)}\}$ using the nearest neighbour approach (Section 4.3). Then using the same deformation technique proposed by Yang et al. [23], we used 9 control points on the boundary of the points $\{x_2^{(i)}\}$ to deform them, generating the points $\{x_1^{(i)}\}$. Again the normal vectors $\{u_1^{(i)}\}$ were computed using the nearest neighbour approach.
   For cost functions $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$, we estimate the point correspondences using the method proposed by Yang et al. [23] and detailed in Section 4.5. We test 4 levels of deformation and register 15 pairs of shapes at each level. We also test the case in which S_1 and S_2 differ by a rotation and non-rigid deformation by applying a rotation to the shape S_1. We set the level of deformation to 3 and test 5 levels of rotation, with 15 pairs of shapes registered for each rotation.
   The MSE results can be seen in Fig. 6(b) and (c). In Fig. 6(b), for all degrees of deformation, GLMD performs the best, while $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ perform similarly. Although we found that the correspondences estimated by Yang et al.s technique and used by $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ were accurate, using only 125 control points for the estimated TPS transformation limited the accuracy of

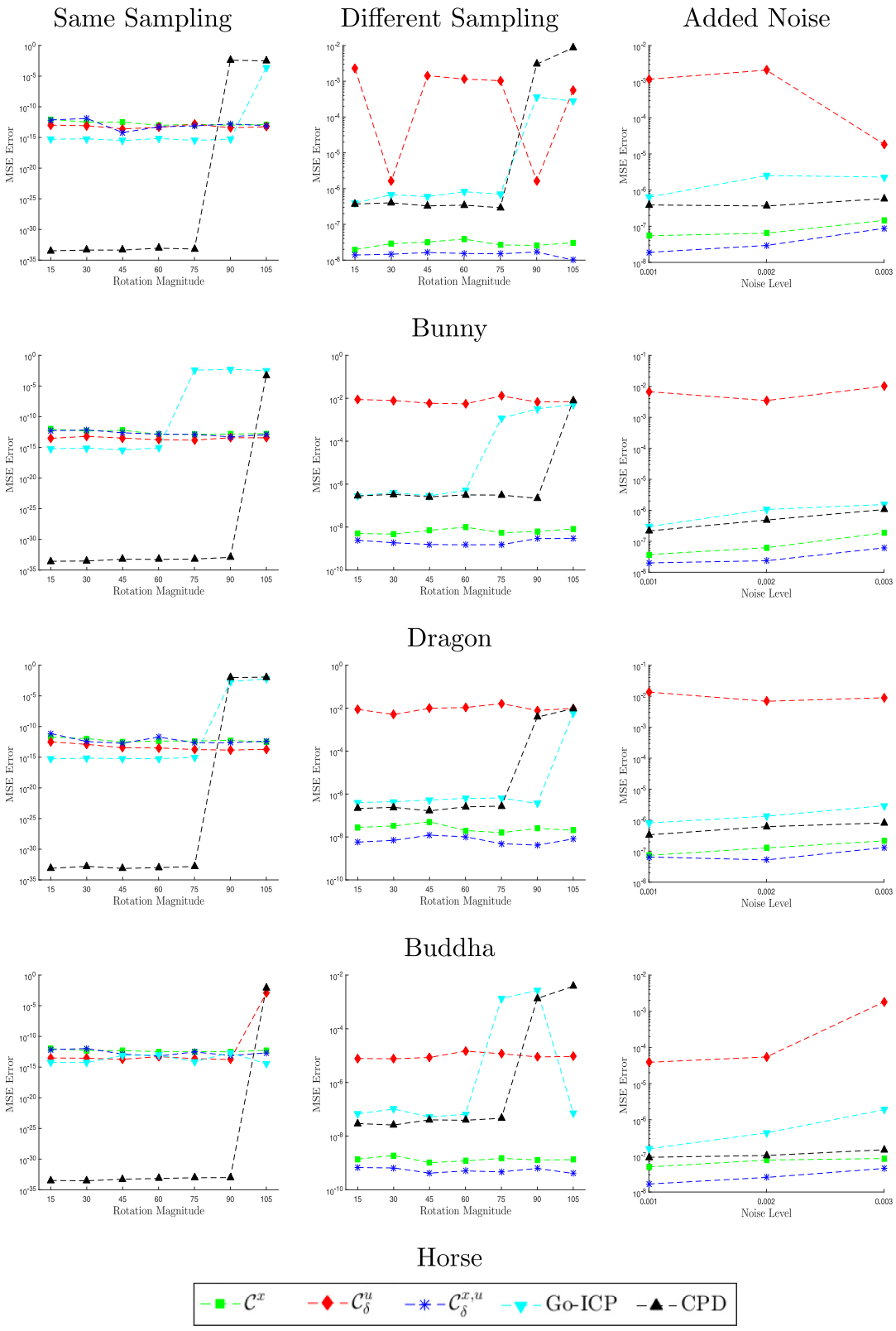


Fig. 4. MSE results for rigid registration with 3D data: same sampling (column 1), different sampling (column 2) and with added noise (column 3). Rows 1–4 give the error results for the Bunny, Dragon, Buddha and Horse meshes respectively.

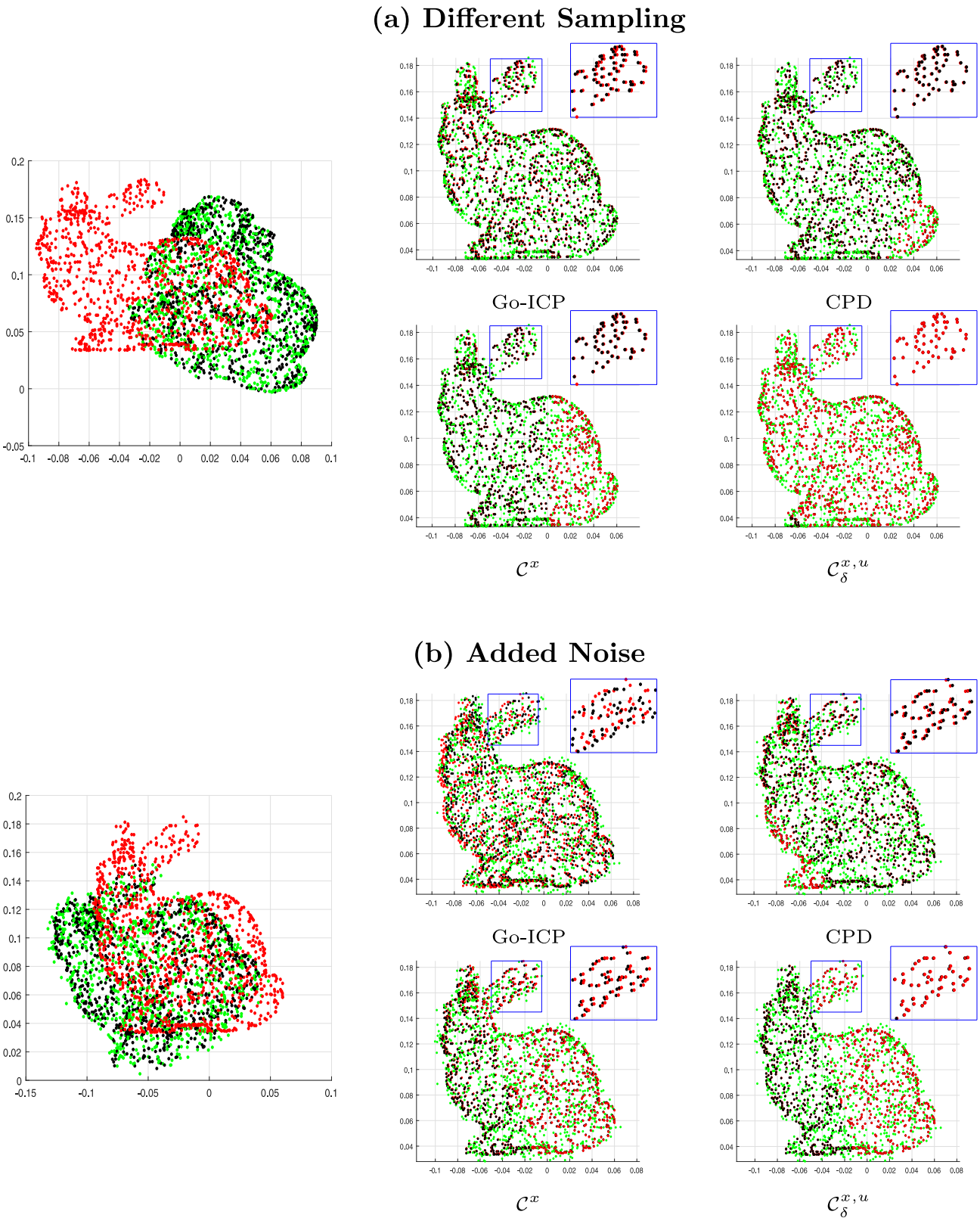


Fig. 5. Results obtained when registering two Bunny shapes S_1 (red) and S_2 (green) differing by a rotation. In (a) the shapes have a different sampling and in (b) noise has been added to the points in S_2, as described in Section 6.1. The graphs on the left show S_1 and S_2 before registration. As S_1 and S_2 will not overlap exactly after registration due to different sampling and added noise, to clarify the difference in accuracy we have also plotted in black the shape S_1 under the same rotation as S_2. After registration, the black and red shapes should overlap exactly. On the right, the registration results of the four algorithms, Go-ICP, CPD, $\mathcal{C}^x$ and $\mathcal{C}_\delta^{x,u}$ are shown, with a zoom-in shown in the top right corner of each graph. In both cases $\mathcal{C}_\delta^{x,u}$ performs best. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

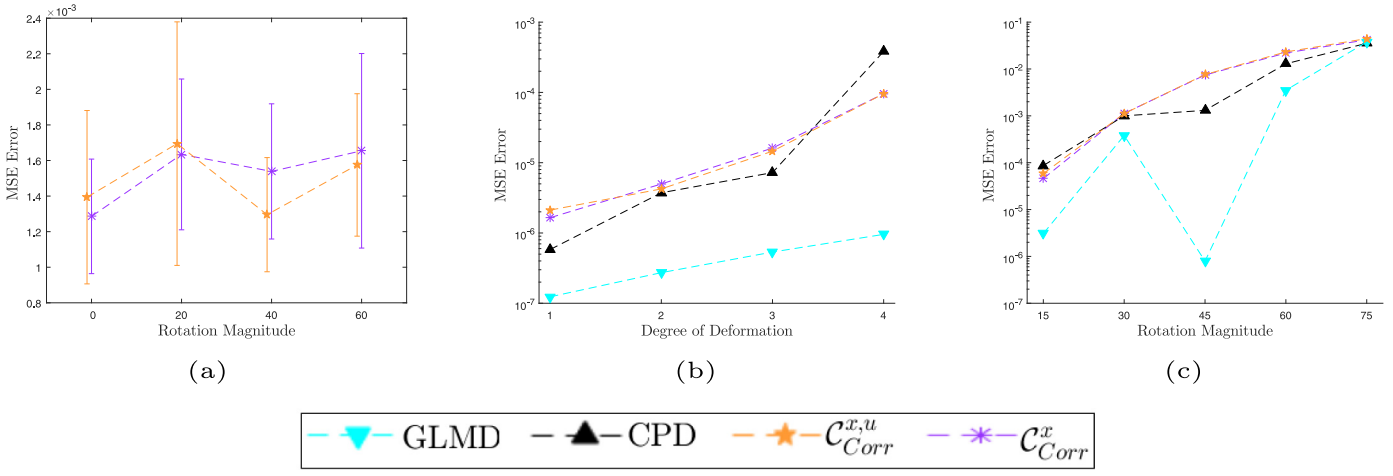


Fig. 6. (a) Comparison between $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ when registering meshes with correspondences that differ by a deformation and rotation. Standard error bars emphasise the similarity between the cost functions; (b) Transformation estimation between bunny shapes differing by a deformation varying from degree 1 to 4; (c) Non-rigid transformation estimation when two bunny shapes differ by a deformation of degree 3 and rotation varying from 15° to 75°.

Table 1
Time taken by each cost function to compute 100 iterations of the gradient ascent algorithm, with the number of TPS control points (col. 2) and latent space dimension (col. 3).

	ctrl pts	dim	n_1	n_2	$\mathcal{C}^{x}$	$\mathcal{C}^{u}$	$\mathcal{C}^{u}_{\delta}$	$\mathcal{C}^{x,u}$	$\mathcal{C}^{x,u}_{\delta}$
2D Rotation	✗	1	100	100	0.20s	0.20s	0.16s	0.21s	0.20s
3D Rotation	✗	9	100	100	0.22s	0.27s	0.29s	0.30s	0.4695
2D TPS	12	30	100	100	2.2s	✗	✗	2.9s	✗
3D TPS	125	387	100	100	16s	✗	✗	30s	✗

Table 2
Number of iterations typically taken by each algorithm to register two pointclouds with 100 points each, computed using our full simulated annealing strategy(number of simulated annealling steps is given in col. 5). *Due to the high dimension of the latent space, we limited the number of function evaluations and iterations in these cases.

	dim	n_1	n_2	Ann Steps	$\mathcal{C}^{x}$	$\mathcal{C}^{u}$	$\mathcal{C}^{u}_{\delta}$	$\mathcal{C}^{x,u}$	$\mathcal{C}^{x,u}_{\delta}$
2D Rotation	1	100	100	6	50	43	50	40	48
3D Rotation	9	100	100	8	220	275	240	390	400
2D TPS	30	100	100	5	1370	✗	✗	1500	✗
3D TPS	387	100	100	8	880*	✗	✗	880*	✗

Table 3
The time taken, on average, by the Go ICP, CPD and GLMD methods to converge to the correct solution.

	n_1	n_2	Go ICP	CPD	GLMD
3D Rotation	100	100	0.78s	32s	✗
2D TPS	100	100	✗	0.09s	0.13s
3D TPS	100	100	✗	0.05s	0.12s

both $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ in comparison to GLMD, which uses all 1000 points in S_1 as control points. However, increasing the number of control points used by $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ also increases the dimension of the latent space, requiring a larger number of iterations to converge to a good solution.

Fig 6(c) shows the results of registering Bunny shapes differed by both a non rigid deformation and rotation. In this case we found that the correspondences estimated by Yang et al.s technique had some errors due to the rotation difference between the shapes. This decreased the accuracy of both GLMD and the cost functions $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$, although GLMD still performed the best. Although $\mathcal{C}^{x,u}_{corr}$ typically performs well when the shapes differ by a rotation, when the wrong point correspondences are used the accuracy of $\mathcal{C}^{x,u}_{corr}$ is reduced. Again we found that using only 125 control points also reduced the accuracy achievable by $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$.

6.3. Computation time

In Table 1 we present the computation times needed by the proposed cost functions to carry out 10 iterations of the gradient ascent algorithm used to register two shapes $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\cdots,100}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,\cdots,100}$, each with 100 points and unit normal vectors. In Table 2 we give the average number of iterations needed by each cost function to converge to the correct solution. These figures were computed when using our full annealing strategy, with the number of annealing steps used given in column 5 of Table 2. In Table 3 we also present the computation times needed by the CPD, Go ICP and GLMD algorithms to register shapes S_1 and S_2.

7. Shape registration and interpolation (qualitative experiments)

We applied the cost functions $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x}$ to the registration of curves extracted from images of patterned letters, taken from a dataset provided by Lu et al. [28][3]. For a given pair of model and

[3] Available at http://gfx.cs.princeton.edu/pubs/Lu_2014_DDS/.

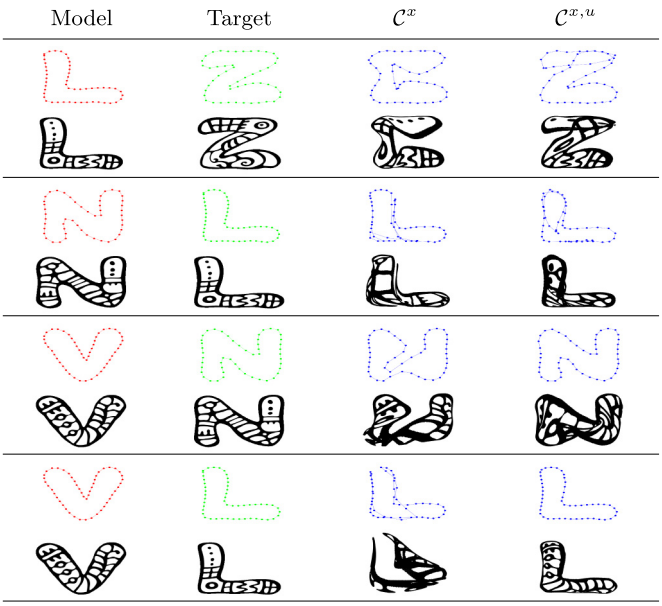


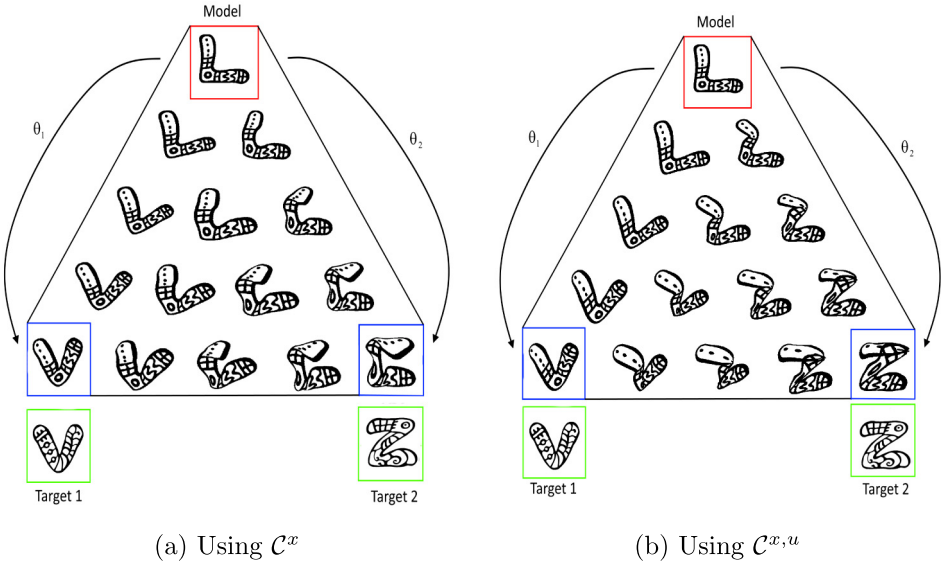
Fig. 7. Rows 1, 3, 5 and 7 show the 50 points extracted along the external boundary of the model (col. 1) and target (col. 2) letters, and the transformation results estimated using $\mathcal{C}^x$ (col. 3) and $\mathcal{C}^{x,u}$ (col. 4). The connectivity information between points is shown, and is used when computing the normal vectors for $\mathcal{C}^{x,u}$. In rows 2, 4, 6 and 8 we show the patterned model letter (col. 1) and target letter (col. 2), and the transformed model letters estimated using $\mathcal{C}^x$ (col. 3) and $\mathcal{C}^{x,u}$ (col. 4).

target letters we extracted 50 points along their external boundary contour (cf. Fig. 7, row 1,3,5,7). We then registered these point clouds, and applied the estimated transformation to all black pixels in the model letter, transforming it into the target letter (cf. Fig. 7, row 2,4,6,8). From Fig. 7 we can see that $\mathcal{C}^{x,u}$ outperforms $\mathcal{C}^x$ when registering the model and target point clouds (row 1,3,5,7). While the connectivity between the points is not taken into account when registering the point clouds using $\mathcal{C}^x$, with $\mathcal{C}^{x,u}$ the normal vectors are computed by fitting a spline to the ordered points, thus capturing some of the connectivity information between them, and giving a better registration result. When the estimated transformation is applied to all black pixels in the model letter, $\mathcal{C}^{x,u}$ again gives a better result. In row 6 and 8 we can see that artifacts can emerge during this step, even when the original point clouds have been registered almost exactly. This occurs when some points inside the boundary curve of the model letter, which are not taken into account during registration, get mapped outside the boundary contour by the TPS transformation, eg. in row 6, column 4, points inside the model letter 'V' have been mapped outside the boundary contour when it is transformed to 'N'. This could be resolved by considering points inside the boundary contour during registration. We also found that in some cases, neither cost function performed well as the model and target letters were too different, and an appropriate TPS transformation could not be estimated that would transform one shape into another.

As the transformations being estimated are parametric, we can create new transformations by interpolating between solutions. For example, given two solutions θ_1 and θ_2, estimated when registering a model letter to two different target letters, we can create interpolations between the three letters using the new transformation θ_{new}:

$$\theta_{new} = \alpha_1\theta_{Id} + \alpha_2\theta_1 + \alpha_3\theta_2, \tag{27}$$

where θ_{Id} is the identity transformation and α_i are scalars. The pyramids in Fig. 8 display samples of shapes generated by interpolating between the estimated transformations, computed using different values of α_i. Similar interpolation results have been presented when using optimal transport for colour transfer and shape registration. These methods use a discrete grid representation of shapes in 2D and 3D and do not explicitly take into account shape connectivity when estimating a registration solution [29,30].



(a) Using $\mathcal{C}^x$ (b) Using $\mathcal{C}^{x,u}$

Fig. 8. Curve registration and interpolation results generated using (a) $\mathcal{C}^x$ and (b) $\mathcal{C}^{x,u}$. In both cases, $\mathcal{C}^x$ and $\mathcal{C}^{x,u}$ are used to register the model letter 'L' (red) to target letters 'V' and 'Z' (green). The registration results after transformation using the estimated parameters θ_1 and θ_2 are outlined in blue, showing that $\mathcal{C}^{x,u}$ performs better than $\mathcal{C}^x$ when registering 'L' to 'Z'. In both cases, new shapes can be created by interpolating between the model shape 'L' and its transformations into 'V' and 'Z'. These are shown in the pyramids. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

8. Conclusion

We have proposed several cost functions to perform registration of shapes encoded with vertex and normal information. These were assessed experimentally for rigid (rotation) and non-rigid transformation for 2D contours and 3D surfaces. We found that our new cost function $\mathcal{C}^{x,u}$ combining normal and vertex information overall outperform others:

- For rotation estimation (2D & 3D), $\mathcal{C}^{x,u}$ performs best overall in terms of accuracy, outperforming Jian and Vemuri's cost function $\mathcal{C}^{x}$ [5] as well as CPD [6] and Go ICP [7].
- For 2D shapes differing by ONLY a non-rigid transformation we found that all techniques perform similarly.
- For 2D shapes differing by a non-rigid transformation AND a rotation, $\mathcal{C}^{x,u}$ outperforms $\mathcal{C}^{x}$ [5] as well as CPD [6] and GLMD [23].
- When partial curves are registered and correspondences are used, $\mathcal{C}^{x,u}_{corr}$ also outperforms CPD [6] and $\mathcal{C}^{x}_{corr}$, giving similar results to GLMD [23].

However, in the case of 3D shapes differing by a non-rigid deformation we found that the high dimensional latent space and small number of control points used reduced the accuracy of $\mathcal{C}^{x,u}_{corr}$ and $\mathcal{C}^{x}_{corr}$. The accuracy of the results also depended on the quality of the correspondences estimated, and the need to compute derivatives and normals vectors at each iteration when using $\mathcal{C}^{x,u}_{corr}$ also increased it's computational cost. Implementing a less time consuming optimisation technique which could explore the latent space quickly would ensure that this type of cost function could be used in higher dimensions. Optimising a combination of these cost functions could also prove beneficial for robust registration, such as removing the rotational difference between shapes using normal information with $\mathcal{C}^{x}$ before estimating the non-rigid transformation with $\mathcal{C}^{x,u}$.

Acknowledgements

This work has been supported by an Ussher scholarship from Trinity College Dublin and partially by EU FP7-PEOPLE-2013- IAPP GRAISearch grant (612334).

Appendix A. Scalar products

The product of two von Mises-Fisher distributions, $vMF_1 = V_d(u;\mu_1,\kappa_1)$ and $vMF_2 = V_d(u;\mu_2,\kappa_2)$ can be written as:

Table A1
Scalar products for Gaussian ($\mathcal{N}$), von Mises-Fisher (vMF) and Dirac (δ) kernels.

$u \in \mathbb{S}^{d-1}$		
	$\delta(u-\mu_1)$	$vMF(\mu_1,\kappa_1)$
$\delta(u-\mu_2)$	✗	$C_d(\kappa_1)\exp(\kappa_1\ \mu_1^T\mu_2)$
$vMF(\mu_2,\kappa_2)$	$C_d(\kappa_2)\exp(\kappa_2\ \mu_2^T\mu_1)$	$\frac{C_d(\kappa_1)\ C_d(\kappa_2)}{C_d(\|\kappa_1\mu_1+\kappa_2\mu_2\|)}$
$x \in \mathbb{R}^d$		
	$\delta(x-\mu_1)$	$\mathcal{N}(x;\mu_1,h_1^2)$
$\delta(x-\mu_2)$	✗	$\mathcal{N}(\mu_1;\mu_2,h_1^2)$, [9]
$\mathcal{N}(x;\mu_2,h_2^2)$	$\mathcal{N}(\mu_1;\mu_2,h_2^2)$, [9]	$\mathcal{N}(\mu_1;\mu_2,h_1^2+h_2^2)$ [5]

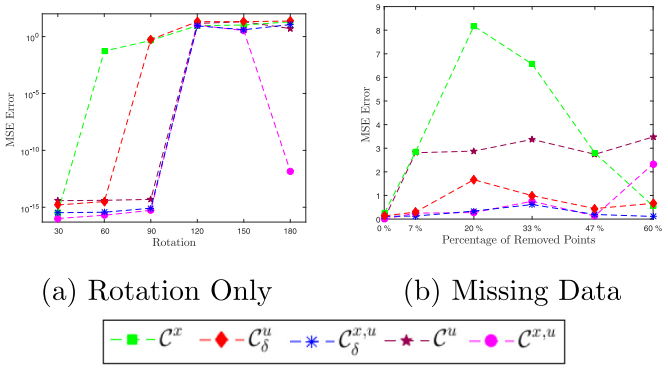


Fig. B1. MSE results for each of our experiments on 2D data differing by a rotation.

$$vMF_1 \times vMF_2 = C_d(\kappa_1)\ C_d(\kappa_2) \times \exp\left(\|\kappa_1\mu_1+\kappa_2\mu_2\|\ \frac{u^T(\kappa_1\mu_1+\kappa_2\mu_2)}{\|\kappa_1\mu_1+\kappa_2\mu_2\|}\right) \quad (A.1)$$

and is proportional to $vMF = V_d(u;\mu,\kappa)$ such that:

$$vMF_1 \times vMF_2 = \frac{C_d(\kappa_1)\ C_d(\kappa_2)}{C_d(\kappa)}\ vMF \quad (A.2)$$

with $\kappa = \|\kappa_1\mu_1+\kappa_2\mu_2\|$ and $\mu = \frac{\kappa_1\mu_1+\kappa_2\mu_2}{\|\kappa_1\mu_1+\kappa_2\mu_2\|}$. Since vMF integrates to 1, the scalar product between vMF_1 and vMF_2 can be defined as:

$$\langle vMF_1|vMF_2\rangle = \int_{u\in\mathbb{S}^{d-1}} vMF_1 \times vMF_2\ du\ = \frac{C_d(\kappa_1)\ C_d(\kappa_2)}{C_d(\kappa)} \quad (A.3)$$

The scalar product between two von Mises-Fisher distributions can therefore be easily computed when an explicit expression for the function $C_d(\kappa)$ is available (e.g. Eq. (6) for $d=3$). Alternatively numerical integration can be used as an approximation to Eq. (5) for any value $d>1$.

Computing the $\mathcal{L}_2$ distance between the KDEs proposed in Section 3 relies on the scalar products between their associated kernels, all of which are summarised in Table A.1.

Appendix B. Additional results

Here we present additional results comparing all cost functions. Fig. B.1 presents results on 2D rotation estimation, similar to those in Fig. 3. We can see that $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ perform similarly in (a), as do $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. In (b) $\mathcal{C}^u_\delta$ seems to perform better than $\mathcal{C}^u$, however both still perform better than $\mathcal{C}^x$ and worse than $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. In Fig. B.2 we present results similar to those in Fig. 4 of the paper for all cost functions. Again both $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ perform similarly, as do $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$, although $\mathcal{C}^u$ appears to get caught in alternate solutions at times, creating spikes in the average MSE results (Fig. B.2, column 1).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.patcog.2018.02.021.

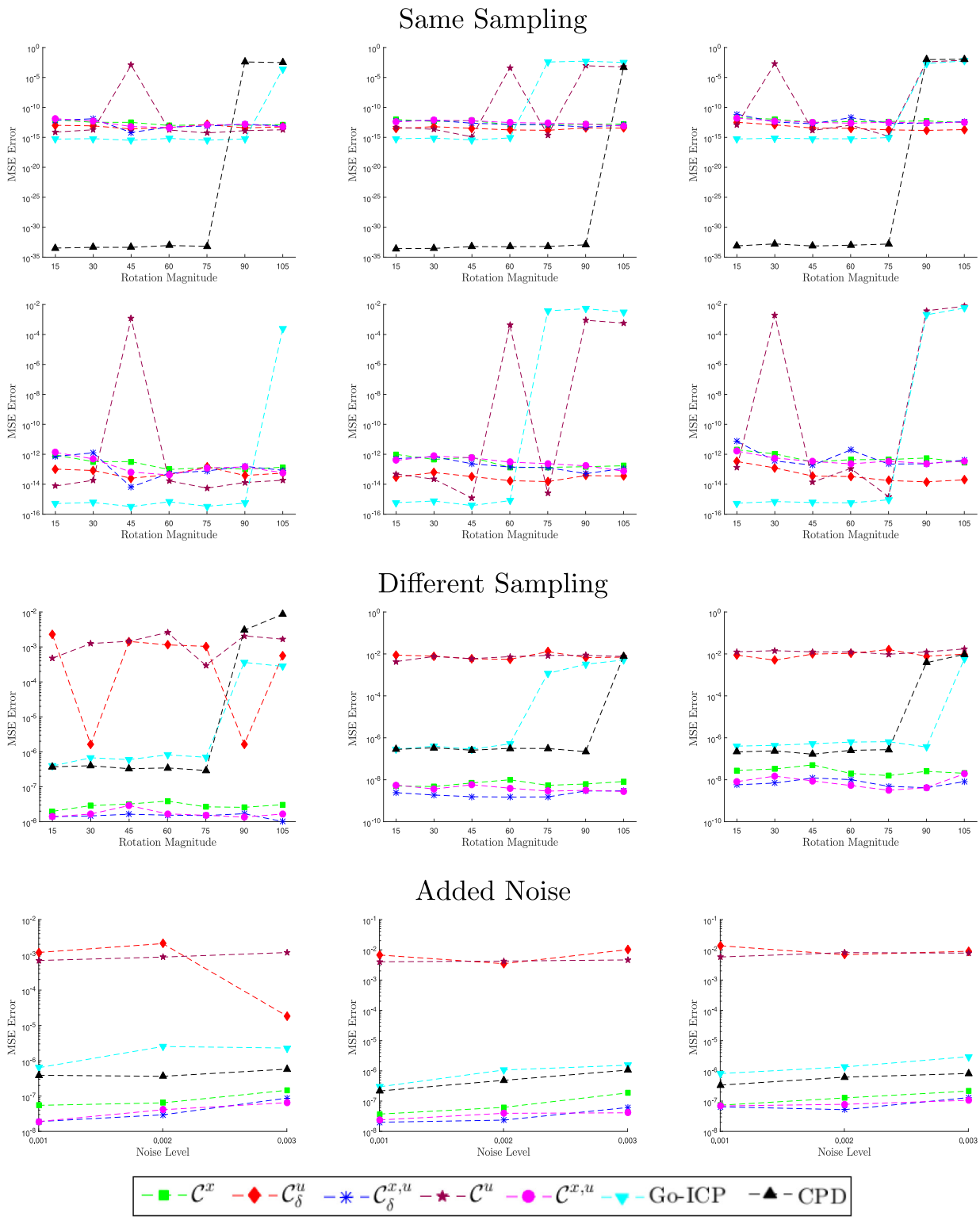


Fig. B2. Error results obtained when registering shapes S_1 and S_2 with the same sampling (row 1), registering shapes with different sampling (row 3) and with added noise (row 4).

References

[1] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 1, 2005, pp. 886–893, doi:10.1109/CVPR.2005.177.

[2] I. Markovic, I. Petrovic, Bearing-only tracking with a mixture of von mises distributions, in: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, 2012, pp. 707–712, doi:10.1109/IROS.2012.6385600.

[3] R. Fisher, Dispersion on a sphere, Proc. R. Soci. London A 217 (1130) (1953) 295–305, doi:10.1098/rspa.1953.0064.

[4] M.A. Hasnat, Unsupervised 3D image clustering and extension to joint color and depth segmentation, Université Jean Monnet - Saint-Etienne, 2014 Theses.

[5] B. Jian, B. Vemuri, Robust point set registration using gaussian mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1633–1645, doi:10.1109/TPAMI.2010.223.

[6] A. Myronenko, X. Song, Point set registration: coherent point drift, IEEE Trans. Pattern Anal. Mach. Intell. 32 (12) (2010) 2262–2275, doi:10.1109/TPAMI.2010.46.

[7] J. Yang, H. Li, D. Campbell, Y. Jia, Go-icp: A globally optimal solution to 3d icp point-set registration, IEEE Trans. Pattern Anal. Mach. Intell. 38 (11) (2016) 2241–2254, doi:10.1109/TPAMI.2015.2513405.

[8] B. Maiseli, Y. Gu, H. Gao, Recent developments and trends in point set registration methods, J. Vis. Commun. Image Represent. 46 (Supplement C) (2017) 95–106, doi:10.1016/j.jvcir.2017.03.012.

[9] D.W. Scott, Parametric statistical modeling by minimum integrated square error, Technometrics 43 (3) (2001) pp.274–285.

[10] C. Arellano, R. Dahyot, Robust ellipse detection with gaussian mixture models, Pattern Recognit. 58 (Supplement C) (2016) 12–26, doi:10.1016/j.patcog.2016.01.017.

[11] J. Fan, J. Yang, D. Ai, L. Xia, Y. Zhao, X. Gao, Y. Wang, Convex hull indexed gaussian mixture model (ch-gmm) for 3d point set registration, Pattern Recognit. 59 (Supplement C) (2016) 126–141, doi:10.1016/j.patcog.2016.02.023.

[12] M. Grogan, M. Prasad, R. Dahyot, L2 registration for colour transfer, in: Proceedings of the 23rd European Signal Processing Conference (EUSIPCO), in: EUSIPCO '15, 2015, pp. 2366–2370. Nice, France.

[13] M. Grogan, R. Dahyot, L2 registration for colour transfer in videos, in: Proceedings of the 12th European Conference on Visual Media Production, in: CVMP '15, 2015, pp. 16:1–16:1.

[14] M. Grogan, R. Dahyot, Robust Registration of Gaussian Mixtures for Colour Transfer, Technical Report, arxiv:1705.06091, 2017.

[15] S. Rusinkiewicz, M. Levoy, Efficient Variants of the ICP Algorithm, in: International Conference on 3-D Imaging and Modeling, 2001, pp. 145–152, doi:10.1109/IM.2001.924423.

[16] S. Ying, J. Peng, S. Du, H. Qiao, A scale stretch method based on icp for 3d data registration, IEEE Trans. Autom. Sci. Eng. 6 (3) (2009) 559–565, doi:10.1109/TASE.2009.2021337.

[17] S. Ying, Y. Peng, Z. Wen, Iwasawa decomposition: a new approach to 2d affine registration problem, Pattern Anal. Appl. 14 (2) (2011) 127–137.

[18] S. Ying, Y. Wang, Z. Wen, Y. Lin, Nonlinear 2d shape registration via thin-plate spline and lie group representation, Neurocomput. 195 (C) (2016) 129–136.

[19] S. Du, J. Liu, C. Zhang, J. Zhu, K. Li, Probability iterative closest point algorithm for m-d point set registration with noise, Neurocomputing 157 (Supplement C) (2015) 187–198, doi:10.1016/j.neucom.2015.01.019.

[20] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 509–522, doi:10.1109/34.993558.

[21] Y. Zheng, D. Doermann, Robust point matching for nonrigid shapes by preserving local neighborhood structures, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 643–649, doi:10.1109/TPAMI.2006.81.

[22] J. Yang, Q. Zhang, Y. Xiao, Z. Cao, Toldi: an effective and robust approach for 3d local shape description, Pattern Recognit. 65 (Supplement C) (2017) 175–187, doi:10.1016/j.patcog.2016.11.019.

[23] Y. Yang, S.H. Ong, K.W.C. Foong, A robust global and local mixture distance based non-rigid point set registration, Pattern Recognit. 48 (1) (2015) 156–173, doi:10.1016/j.patcog.2014.06.017.

[24] H. Guan, W.A.P. Smith, Structure-from-motion in spherical video using the von mises-fisher distribution, IEEE Trans. Image Process. 26 (2) (2017) 711–723, doi:10.1109/TIP.2016.2621662.

[25] R. Hoffman, A.K. Jain, Segmentation and classification of range images, IEEE Trans. Pattern Anal. Mach. Intell. 9 (5) (1987) 608–620, doi:10.1109/TPAMI.1987.4767955.

[26] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, SIGGRAPH Comput. Graph. 26 (2) (1992) 71–78.

[27] R.W. Sumner, J. Popović, Deformation transfer for triangle meshes, ACM Trans. Graph. 23 (3) (2004) 399–405, doi:10.1145/1015706.1015736.

[28] J. Lu, C. Barnes, C. Wan, P. Asente, R. Mech, A. Finkelstein, Decobrush: drawing structured decorative patterns by example, ACM Trans. Graph. 33 (4) (2014) 90:1–90:9, doi:10.1145/2601097.2601190.

[29] J. Solomon, F. de Goes, G. Peyré, M. Cuturi, A. Butscher, A. Nguyen, T. Du, L. Guibas, Convolutional wasserstein distances: efficient optimal transportation on geometric domains, ACM Trans. Graph. 34 (4) (2015) 66:1–66:11, doi:10.1145/2766963.

[30] N. Bonneel, J. Rabin, G. Peyr, H. Pfister, Sliced and radon wasserstein barycenters of measures, J. Math. Imaging Vis. 51 (1) (2015) 22–45, doi:10.1007/s10851-014-0506-3.

[31] M. Grogan, Colour Transfer and Shape Registration using Functional Data Representations, Ph.D. thesis, Trinity College Dublin, Ireland, 2017.

Mairéad Grogan received a degree in Mathematics from Maynooth University in Ireland and a Masters in Computer Science from Trinity College Dublin. She gained a Ph.D. from Trinity College Dublin and is currently a research fellow there. Her research interests include robust statistics, colour transfer and image processing.

Rozenn Dahyot (Ph.D. 2001, University of Strasbourg) is a Associate Professor in the School of Computer Science and Statistics in Trinity College Dublin, Ireland. Her research interests include pattern recognition and statistical learning applied image processing amongst others.