Colour Transfer and Shape Registration using Functional Data Representations

by

Mairéad Grogan, M.Sc, B.A.

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfillment

of the requirements

for the Degree of

Doctor of Philosophy

University of Dublin, Trinity College

April 2017

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement

Mairéad Grogan

April 13, 2017

Abstract

In this thesis we propose new colour transfer and shape registration methods based on the robust $\mathcal{L}_2$ distance. For colour transfer, we present an approach inspired by techniques recently proposed in shape registration. We model the colour distribution of a palette and target image using Gaussian Mixture Models and register them using the $\mathcal{L}_2$ distance. We estimate a parametric transfer function ϕ which can be easily stored in memory for later use and allows for the interpolation of several colour transfer functions which can create interesting special effects. We also show that pixel correspondences can be easily incorporated into our method to enhance the colour transfer result. We show that our method compares well both qualitatively and quantitatively to other colour transfer approaches and that our recolouring step is computationally the fastest.

We also propose a new shape registration technique which extends previous registration methods that model shapes as probability density functions and estimate the registration parameters by minimising a divergence between them. Our proposed technique models the point positions and directional information of a shape, and we investigate mixture models with Dirac, Gaussian and von Mises-Fisher kernels. We validate our framework experimentally on shapes differing by both a rotation and non-rigid deformation and show that in general using both point and normal vector information allows for better registration of shapes. Finally, we present a short exploration of the results generated when two optimal transport techniques are applied to the 3D shape registration problem.

Acknowledgments

Firstly, I would like to thank my supervisor Rozenn Dahyot, whose guidance and encouragement over the years has been invaluable. I would also like to thank Trinity College Dublin, who provided me with the Ussher scholarship funding that made this PhD possible. I would like to thank my parents, whose support over the years has allowed me to put my study first, for which I will be forever grateful. Thanks to Úna and Hannah, and my extended family. Your support and encouragement is never ending and I'm always grateful to have such a close family unit, I don't know where I'd be without you all. Thanks to Siobhán, who was always free to drink tea, eat chocolate muffins, and offer words of wisdom from someone who had done it all before. To Louise, for meeting me for many lunch dates and arriving with doughnuts and coffee when thesis writing extended late into the evening, doughnuts and coffee have never tasted so good! Thanks to Dave, who was always free for a coffee break or to help me fix a few linking errors, your help is greatly appreciated. To all the members of GV2 whose friendship and help over the years has been invaluable, and without whom I wouldn't have been introduced to the magic that is coffee! And finally I'd like to thank Brendan. Having you by my side over the last few years has made this journey so much more enjoyable, and I will be forever grateful for all the help and encouragement you have given me.

Mairéad Grogan

Contents

List of Tables

List of Figures

Acronyms

ANN Artificial Neural Network

CIELab CIELab colour space

CNN Convolution Neural Network

DNN Deep Neural Network

EM Expectation Maximisation

GLMD Global and Local Mixture Distance

GMM Gaussian Mixture Model

GR^{2}T Generalised Relaxed Radon Transform

HSL Hue Saturation Luminance colour space

ICP Iterative Closest Point

KC Kernel Correlation

KDE Kernel Density Estimate

KL Kullback-Leibler

MLS Moving Least Squares

MS Mean Shift

OT Optimal Transport

PCA Principal Component Analysis

PDF Probability Density Function

PSNR Point Signal to Noise Ratio

RANSAC Random Sample Consensus

RBF Radial Basis Function

RGB Red Green Blue colour space

SIFT Scale Invariant Feature Transform

SSIM Structural Similarity Index

SURF Speeded Up Robust Features

TPS Thin Plate Spline

YUV YUV colour space

Chapter 1

Introduction

The widespread use of scene capturing devices such as cameras and 3D scanners has generated many different forms of data for processing and analysis, such as images, videos and point clouds. Computer vision techniques have been developed to process and analyse this data in order to develop a better understanding of the scene being captured. In order to combine several images or video frames for analysis, preprocessing steps typically need to be carried out to make the data coherent. Two such steps include colour transfer (or mapping) and shape registration, and in this thesis we will explore both of these areas. In both colour transfer and shape registration, the success of a technique can be assessed by how well it meets certain criteria, such as its speed, robustness to noise, flexibility, consistency of results, amount of user input required and the accuracy or quality of its performance. Methods which tick all of these boxes are highly sought after and in this thesis we will investigate ways to improve these aspects of current state of the art techniques.

1.1 Overview and motivation

1.1.1 Colour Transfer

When many images or videos are captured of a scene, several factors can cause the colours of the same object to become inconsistent over several images. These factors include different lighting conditions, different imaging devices or different device settings. In many computer vision systems, such as image or video stitching systems

Target Image Palette Image Result Image

Figure 1.1: Colour transfer techniques applied to images with similar content and pixel correspondences (Row 1), and applied to images with different content(Row 2).

and 3D stereo systems, images are required to have consistent colours. In these cases, colour transfer methods can be used to fix colour differences between images, as in Figure 1.1.

Colour transfer techniques can also be used to transfer the colour feel of one image to another when there is no shared content between the images, as in Figure 1.1. This type of colour transfer is commonly used as a post processing step in image editing or film production. These techniques can also take user input into account when determining the colour mapping that should be applied, or use pixel correspondences to enhance the colour transfer result.

While many colour mapping systems exist, they are typically inflexible and specified for a single type of data, such as images with similar content, images with different content, or video data. Few techniques can be applied successfully to all data types. With the high demand for high quality HD images and video, a very large number of pixels typically require processing and the speed of recolouring techniques has become an important factor. Many colour transfer techniques also require the storage of large look up tables which can be very memory consuming. Many methods also use discrete approximations of colour distributions which can create blocky artifacts in the result

(a) Point Cloud Registration

(b) Image Registration

(c) Curve Registration

Figure 1.2: Applications of shape registration. (a) Point cloud registration: the meshes shown in gold and green (left) are registered so that they are aligned (right); (b) Image registration: several images of the same scene are registered to create a larger picture of the scene; (c) Curve registration: several curves representing the letters 'fda' have been registered so that they overlap.

images or videos, and require a further smoothing step as a post process. In this thesis, we try to combat some of these issues arising in the current state of the art techniques, and propose a method which creates good colour transfer results with few artifacts, is robust to pixel correspondence outliers, can be applied to images with both similar and different content, and has a fast recolouring step.

1.1.2 Shape Registration

In many computer vision systems, shape registration is also an essential preprocessing step which ensures that images or point clouds representing the same object are properly aligned. Image registration is important in many applications including stereo vision, image panorama creation and in the analysis of medical images. 3D shape registration can be used to register 3D scans or point clouds which are captured using 3D scanners such as the Kinect. This step is essential for techniques such as 3D surface reconstruction, which typically register several scans of a scene and construct a surface from the aligned point clouds. Some applications of shape registration are given in Figure 1.2.

While point set representations of 2D or 3D shapes are easily generated from images or scans, other information capturing the structure of the shape can be easily computed

from images and point clouds but is typically not taken advantage of when registering shapes. For example, image gradients and feature descriptors can be computed from an image, while normal vectors and shape curvature can be computed from 3D point clouds and meshes. While many shape registration techniques use only the point cloud information, taking into account these additional features would create a more comprehensive model of the shape and could prove more powerful for shape registration than models taking into account only point cloud information. In this thesis, we propose a shape registration technique based on the $\mathcal{L}_2$ distance, which not only takes into account the point cloud information of 2D and 3D shapes, but also incorporates normal vectors into the shape model. Our goal is to find a new registration technique which improves on the registration accuracy of current state of the art techniques while still remaining robust to outliers. We propose using the von Mises-Fisher distribution to model the unit normal vectors and investigate mixture models using a combination of Gaussian, Dirac and von Mises-Fisher kernels.

1.2 Summary of Contributions

The key contributions of this thesis are:

1. We propose a colour transfer method based on the $\mathcal{L}_2$ distance which minimises the divergence between two probability density functions (PDFs) in order to match the colour distribution of one image to another. Our framework uses continuous PDFs to model colour distributions which eliminates artifacts that are typically generated by other techniques that used discrete distribution estimation methods such as histograms.

2. We propose a framework for colour transfer that can be applied to different data types including images with no shared content, images with similar content, and video data. While our algorithm can be successfully applied to target and palette images which have no pixel correspondences, correspondences can also be easily incorporated when they are available. We also propose to use a parametric colour transformation which, once estimated, can be easily applied to several video frames without creating any temporal artifacts.

3. We explore the benefits of using a parametric transformation for recolouring and show that it allows the recolouring step to be applied in parallel, ensuring that the recolouring is fast. It also allows for the creation of interesting colouring effects which can be created by interpolating or fading between more than one colour transfer function. These interpolated colouring effects can also be easily applied to video clips.

4. We propose a shape registration technique that takes into account both point positions and directional information in the form of unit normal vectors when modelling 2D and 3D shapes. We also propose using the von Mises-Fisher distribution to model the normal vectors and show that the $\mathcal{L}_2$ distance between two mixture models with von Mises-Fisher kernels has an explicit expression when the dimension $d = 3$. We show that our proposed approach gives improved registration results in comparison to other shape registration techniques, especially when the transformation being estimated includes a rotation.

5. We explore the application of two optimal transport techniques, previously proposed for colour transfer, to the shape registration problem to see if they prove to be advantageous in this area.

1.3 Thesis outline

The work carried out in this thesis is structured into six chapters. An overview of the current state of the art in colour transfer and shape registration is presented in Chapter 2. In Chapter 3 we outline both colour data and shape data, and detail the different representations that these data types can have. We also summarise several transformation functions that are typically used in colour transfer and shape registration.

Chapters 4 and 5 contain the main contributions of this thesis. In Chapter 4 we propose a new colour transfer technique which is based on minimising the $\mathcal{L}_2$ distance between two PDFs, modelled as Gaussian Mixture Models (GMMs). When choosing our parametric colour transfer function ϕ, we investigate several different transformations including affine transformations, thin plate spline transformation and radial basis functions. We explore both RGB and CIELab representations of the colour data and investigate the use of the K-means and Mean Shift clustering algorithms for estimating

the centres of the kernels in each GMM. We show that our technique can be enhanced by pixel correspondences, can be easily applied to video data, and has a fast recolouring step. We also present an extensive experimental section which presents both qualitative and quantitative results showing that our method competes well with other state of the art techniques.

In Chapter 5 we present our shape registration technique that proposes to model both the point positions and directional information of 2D and 3D shapes. We present four cost functions which model directional data using the von Mises-Fisher distribution and minimise the $\mathcal{L}_2$ distance between two such PDFs. In the experimental section we test how our proposed cost functions perform when registering 2D and 3D shapes differing by a rotation or non-rigid deformation, and compare our method to other state of the art techniques.

In Chapter 6 we present a short investigation into the application of optimal transport methods to 3D shape registration to see if they provide any benefits to the area. Again, we present some registration results on 3D shapes differing by both a rotation and non-rigid deformation, and investigate the transformation ϕ estimated by this technique.

Finally, Chapter 7 summarises the work carried out in this thesis and presents possible future directions of investigation.

1.4 List of publications

Part of the work carried out in this thesis has been published in the following articles:

1. M.Grogan, J. Carvalhot and R. Dahyot, Recent techniques for (re)colouring, Irish Machine Vision and Image Processing Conference (IMVIP), Galway, August 2016 [9].

2. M.Grogan and R. Dahyot, short paper, $\mathcal{L}_2$ Registration for Colour Transfer in Videos, Conference on Visual Media Production (CVMP), London, November 2015 [10]. This work is summarised in Chapter 4.

3. M.Grogan, M. Prasad and R. Dahyot, $\mathcal{L}_2$ Registration for Colour Transfer, European Signal Processing Conference (EUSIPCO), Nice France, September 2015

[11]. This work is summarised in Chapter 4.

4. A. Bulbul, M. Grogan and R. Dahyot, 3D reconstruction of Reflective Spherical Surfaces from Multiple Images, Irish Machine Vision and Image Processing Conference (IMVIP), Dublin, August 2015 [12].

5. M.Grogan and R. Dahyot, Mesh from Depth images using GR^2T, Irish Machine Vision and Image Processing Conference (IMVIP), Derry-Londonderry, Northern Ireland, August 2014 [13].

Chapter 2

Related Work

In this chapter we give an overview of shape registration and colour transfer, and their applications in computer vision. One of the main contributions of this thesis is to introduce a colour transfer technique which has been inspired by techniques previously proposed in shape registration. Therefore, in this chapter we will first summarise techniques in the field of registration (Section 2.1), and then present a brief review of colour transfer (Section 2.2).

2.1 Registration

Registration is a fundamental task in computer vision and graphics, and serves as an initial step in many applications including surface reconstruction, medical image analysis, and object recognition. These applications rely on tasks such as stereo matching, shape matching and image retrieval, all of which can be accomplished using point set registration [14, 15, 16]. The point representations used include feature points extracted from images, points on a shape contour or vertices in a mesh [17].

The overall aim of registration is to find the best alignment between two or more shapes, as in Figure 2.1. These shapes can represent the same object or different objects with similar shape. In this section we will outline the registration problem (Section 2.1.1) and registration algorithms that can be categorised as ICP-based algorithms (Section 2.1.2), those that use local shape descriptors (Section 2.1.3), EM-like algorithms (Section 2.1.4) and those that match probability density functions (Section

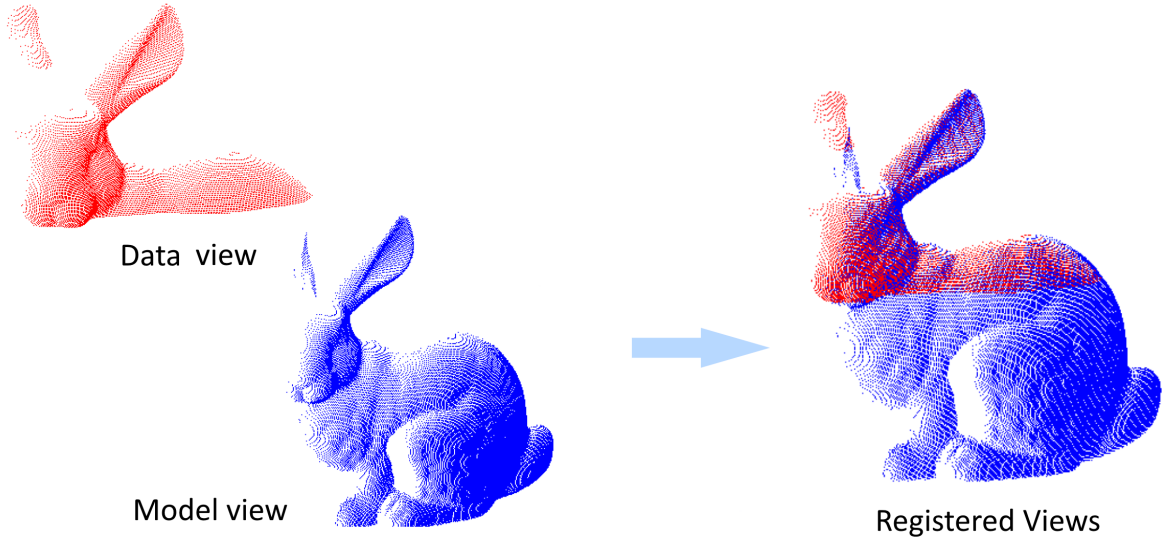


Figure 2.1: Given two views of an object, the aim of registration is to find the transformation that aligns the two views [1].

2.1.5). For an exhaustive review see [18, 1].

2.1.1 The registration problem

Suppose we are given two point sets $X = \{x_i\}_{i=1}^{n_1}$ and $Y = \{y_j\}_{j=1}^{n_2}$, where $x_i, y_j \in \mathbb{R}^d$. The registration problem involves determining the transformation $\phi : \mathbb{R}^d \mapsto \mathbb{R}^d$ to be applied to the model point set X, such that X and Y are aligned and the difference between them is minimised. As there are an infinite number of solutions for ϕ, some a priori knowledge about ϕ is assumed, for example, whether ϕ is a rigid or non-rigid transformation. Often, a set of point correspondences $\{(x_i, y_i)\}_{i=1,..n}$ is used to improve the estimation of ϕ, and indicates corresponding points in the point sets X and Y that should be aligned after registration.

Estimating a rigid transformation is a challenging problem and has been widely studied [19, 20, 21]. Some challenges which face this type of problem include analysing data which is effected by noise, outliers, or self occlusion. Two data sets which have a limited amount of overlap or have different data resolutions can also be difficult to register [18]. As well as facing the above challenges, non-rigid registration must also account for deformation. In many cases this deformation is unknown and can

be hard to model. The solution space therefore has a higher dimensionality than the rigid registration space, making the optimisation more complex [22]. In contrast to rigid transformation estimation, which typically requires only a small number of correspondences to be successful, non-rigid registration often requires a large set of reliable correspondences for an accurate solution to be estimated [18].

2.1.2 Iterative Closest Point

One of the most popular methods for estimating ϕ when it is a rigid transformation is the Iterative Closest Point algorithm (ICP). The ICP algorithm was first proposed by Besl et al. [23] and Zhang et al. [24], and uses the nearest neighbour relationship to assign point-to-point correspondences between the two datasets. These correspondences are then used to estimate the transformation using least squares. This two step process is iterated until some convergence criterion is reached. This method is simple and has a low computational complexity, however it assumes that closest point pairs should correspond, which may not be true if the point sets are not coarsely aligned. Without a good initialisation, the ICP algorithm has a tendency to fall into local minima [25].

Improvements to all phases of the original ICP algorithm have since been proposed, from the selection of point correspondences to the minimisation strategy [25, 21, 26, 27, 28]. To alleviate the local minima issue, Fitzgibbon et al. [21] proposed LM-ICP which uses a smoother objective function, resulting in an enlarged basin of convergence. Instead of minimising point to point distances, Chen et al. [29] minimize point to plane distances. Yang et al. [28] propose to combine ICP with a branch-and-bound scheme which efficiently searches the 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 registration problem.

2.1.3 Local Shape Descriptors

When estimating non-rigid transformations, taking into account the neighbourhood structure of the point sets has been proposed [30, 31, 32, 33]. Belongie et al. [30] propose using the shape context descriptor when registering point sets, which helps establish better point correspondences. Zheng et al. [34] introduce the concept of local neighbourhood structure for the general point matching problem, which was later

improved in [35]. Ma et al. [32] also propose a matching method which exploits local structures and uses the shape context descriptor to establish rough point correspondences before estimating the non-rigid transformation.

Yang et al. [33] 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. They define a distance which combines both global and local feature differences and use it to estimate point correspondences. An annealing scheme is implemented which ensures that the defined distance gradually changes from a local distance to a global distance, allowing the initially estimated transformation to account for any large deformation, with a more non-rigid transformation being estimated in the final iterations.

2.1.4 EM-Like Algorithms

To overcome the limitations of ICP based algorithms, probabilistic methods have been proposed [36, 22]. Instead of using one-to-one correspondences as in ICP based techniques, Chui and Rangarajan [36] propose a one-to-many correspondence scheme, which was later extended in [37]. Using soft assignment and deterministic annealling, they alternately estimate the transformation and update the correspondences. They have also shown that this alternate update scheme is equivalent to the Expectation Maximisation (EM) algorithm for GMMs [36]. The non-rigid transformation is modelled using thin plate splines.

Several methods have also treated the registration problem as a maximum likelihood problem [22, 38, 39, 40]. Myronenko et al. [22] present a similar method to that proposed by Chui et al. [36]. They also consider the registration problem as a probability density estimation problem, letting one point set represent the GMM centroids and the other the data points. They also incorporate a coherent motion constraint and model the non-rigid transformation as a Gaussian radial basis function.

2.1.5 Matching Probability Density Functions

Other methods propose to model both datasets as probability density functions. The transformation is then estimated by minimising the distance between these density

functions. In [41], Jian and Vermuri showed that both ICP methods and EM-like algorithms can also be interpreted as methods which minimise a divergence between pdfs. Both the modelling of the density functions and the metric used can differ among methods. Kernel density estimates (KDEs) are a non-parametric method used to estimate a density function and are regularly used in density matching methods [42, 41, 43].

Kernel Density Estimates

Given a dataset $\{x_i\}_{i=1,..,n} \in \mathbb{R}^d$ drawn from a distribution with an unknown density $p(x)$, the density $p(x)$ can be estimated using a kernel density estimate as follows:

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h^d}\, \mathcal{K}\Big(\frac{\|x - x_i\|}{h}\Big) \tag{2.1}$$

Here $\mathcal{K}$ is a kernel function. Each kernel function is centred about a data point x_i and is controlled by the bandwidth $h > 0$. A kernel can be any function f which satisfies the following conditions:

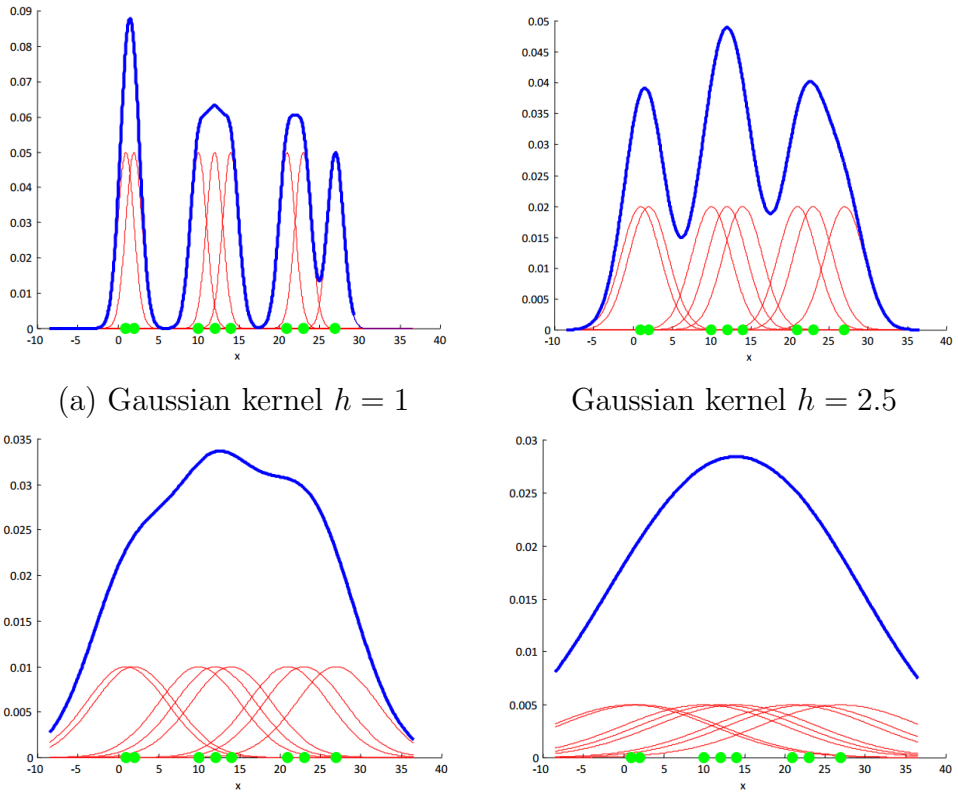
$$(a) \int f(x)dx = 1 \qquad (b) \int xf(x)dx = 0 \qquad (c) \int x^2 f(x)dx < \infty \tag{2.2}$$

The Gaussian kernel is commonly used and is given by:

$$\mathcal{K}\Big(\frac{\|x - x_i\|}{h}\Big) = \frac{1}{(2\pi)^{\frac{d}{2}}} \exp\Big(\frac{-\|x - x_i\|^2}{2h^2}\Big). \tag{2.3}$$

A KDE with a Gaussian kernel is equivalent to a GMM with isotropic Gaussians centred at each data point.

Choosing an appropriate value for h is an important task as it directly affects the shape of the density estimate. Some examples of the impact of different values of h on the density estimate can be seen in Figure 2.2. Several methods for estimating h have been proposed and are based on minimising a similarity metric between the estimated density $\hat{p}(x)$ and the true, but unknown density, $p(x)$ [44, 45].



(a) Gaussian kernel $h = 1$ Gaussian kernel $h = 2.5$

(a) Gaussian kernel $h = 5$ Gaussian kernel $h = 10$

Figure 2.2: This figure shows the effect that changing h has on the density estimated using a kernel density estimate. In each figure, Gaussian kernels (red) with bandwidth h are centred about each data point (green) to compute the density estimation (blue). Figure sourced from [2].

Distance Metrics

Several distance measures have been proposed to compute the similarity between two probability density functions $f(x)$ and $g(x)$ [41, 42, 46]. The Kullback-Leibler (KL) divergence is given by [41]:

$$KL(g, f) = \int g(x) \log \left\{ \frac{g(x)}{f(x)} \right\} dx, \tag{2.4}$$

and in [41], Jian and Vermuri observe that minimising an approximated KL divergence [47] between two GMMs is equivalent to the ICP method. EM-like algorithms such as

those proposed in [36, 22], which maximise the likelihood function, are also equivalent to minimising the KL divergence between two probability distributions. In this case, one pdf is modelled as a GMM while the other is a mixture of delta functions.

Another metric, known as the $\mathcal{L}_2$ distance, is given by [41]:

$$\mathcal{L}_2(g, f) = d_1(g, f) = \int g^2(x)dx - 2\int g(x)f(x)dx + \int f^2(x)dx. \tag{2.5}$$

The $\mathcal{L}_2$ distance is the Euclidean distance between the density functions f and g and can be conveniently rewritten as $\mathcal{L}_2(g, f) = \|g\|^2 - 2\langle g|f\rangle + \|f\|^2$. The term $\|f\|^2$ is proportional to the quadratic Renyi entropy of f, while the scalar product term $\langle g|f\rangle$ is proportional to the Renyi cross entropy between g and f [48].

Both the $\mathcal{L}_2$ distance and the Kullback-Leibler divergence are part of a family of divergences known as the power density divergence, more details of which can be found in [41]. The advantage of computing the $\mathcal{L}_2$ distance over the KL divergence between pdfs is that it can be computed explicitly with GMMs and it has also been shown to be more robust to outliers [41, 49]. Jian and Vermuri [41] propose to register two point sets by fitting a KDE with Gaussian kernels to each data set and minimising the $\mathcal{L}_2$ distance between them. In [50], Arellano et al. extend this framework by adding gradient direction information to the model and show that it gives improved results for ellipse detection. They add an extra dimension to the KDE to capture the direction of the normal vector at each point in the point set. However, their framework proposes modelling the normal vectors using the Gaussian distribution, and while this is suitable when the normal vectors are two dimensional, it is not suitable when using normal vectors of higher dimensions.

When the pdf g is parametrised by a latent variable θ, the robust $\mathcal{L}_2$ distance can be used to estimate θ as follows:

$$\hat{\theta} = \arg\min_{\theta} \left\{ \int g^2(x|\theta)dx - 2\int g(x|\theta)f(x)dx \right\}. \tag{2.6}$$

Since the term $\int f^2(x)dx$ is independent of θ it can be discarded from Equation 2.5. When f is the empirical probability density function this estimator is known as the $\mathcal{L}_2E$ estimator [49], which has been previously proposed for shape registration by Ma et al. [51]. Ma et al. also propose using point correspondences to improve robustness

and reduce computational complexity.

When f and g are KDEs the kernel correlation KC can be computed as [42]:

$$KC(f,g) = \sum_{x_i \in f} \sum_{x_j \in g} \int \mathcal{K}(x, x_i)\mathcal{K}(x, x_j)dx. \tag{2.7}$$

If $\mathcal{K}$ is chosen to be the Gaussian kernel, the kernel correlation between the GMMs f and g is proportional to the term $\int f(x)g(x)dx$ found in the $\mathcal{L}_2$ distance (Equation 2.5) and has been previously proposed for rigid registration by Tsin et al. [42]. When estimating non-rigid transformations a normalisation term can be added to KC leading to the similarity measure Cor:

$$Cor(f,g) = \frac{\int f(x)g(x)dx}{\sqrt{\int f(x)^2dx \int g(x)^2dx}}, \tag{2.8}$$

which is commonly considered as the correlation between f and g [41]. This also has a closed form solution for GMMs and was used to register point sets differing by a rigid transformation in [52]. The Cauchy-Schwarz divergence has a similar form and has also been used for shape registration [53]. The Generalised Relaxed Radon Transform, or GR^{2}T [54], has also been proposed for robust inference and is related to the cross correlation between two pdfs $\langle g|f \rangle$. It has been previously proposed for both shape registration and surface reconstruction [55, 56].

2.2 Colour Transfer

Colour transfer, or colour mapping, is a set of techniques that aim to modify the colour feel of a target image or video using an exemplar colour palette provided by another image or video, as in Figure 2.3. This colour transformation involves estimating a transfer function ϕ which, once estimated, is used to recolour a colour pixel value x to $\phi(x)$. A range of applications in computer vision and graphics has motivated research in this area including image stitching, greyscale image colourisation, and colour correction in stereo images [57, 58, 59]. Color mapping techniques can be classified into three categories: geometry based methods, statistical based methods, and user assisted techniques, as in Figure 2.4. Within these groups the methods can be classified further

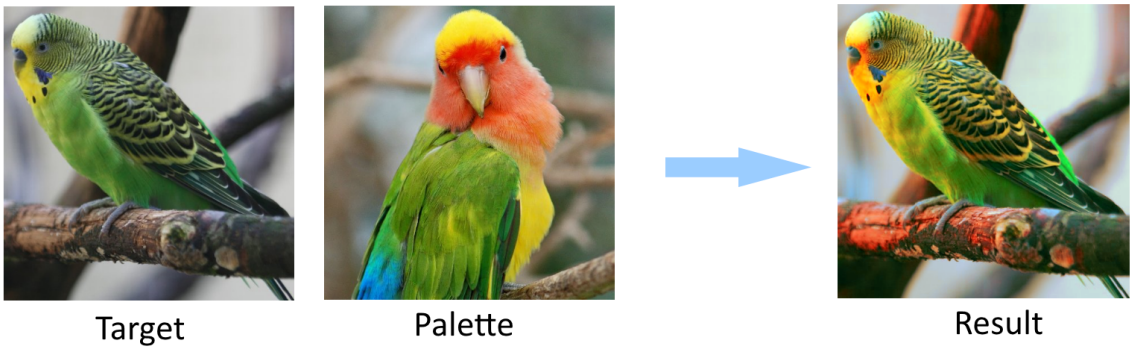


Figure 2.3: Colour transfer methods change the colour feel of a target image or video using a colour palette provided by an exemplar image or video.

based on the images they can be applied to, the colour space that they are performed in, whether they are local or global methods, and the statistical properties that they consider. In this section, we give an overview of the state of the art methods in each area. A more detailed review can be found in [3].

2.2.1 Geometry based methods

In applications such as image stitching and stereo capture, the appearance of two images taken of the same scene may differ due to differences in viewpoint, lighting conditions, imaging devices and acquisition parameters. Colour transfer techniques are then used to enhance the colour consistency between the two images. The transfer of colour from one image to another can be facilitated by searching for correspondences between the two images. This ensures that features which appear in both images have the same colour. However, as many geometry based methods rely on colour correspondences when computing the colour transformation, they cannot be easily extended to target and palette images that have no pixel correspondences available.

Sparse Feature Correspondences

Feature detection methods such as SIFT [60] and SURF [61] have been used in many colour transfer algorithms to find sparse feature correspondences between the target and palette images [62, 63, 64]. Methods such as RANSAC [65] can also be applied to effectively reject outliers from the correspondence set. Yamamoto et al. [62, 63] com-

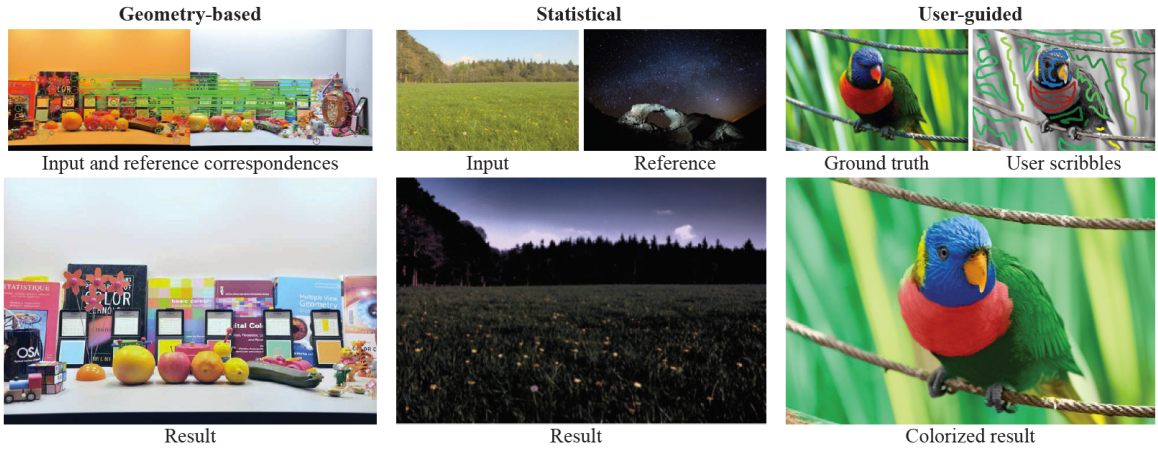


Figure 2.4: Colour transfer methods can be broadly classified as Geometry based methods, statistics based methods and user-guided approaches [3]. Geometry based methods (left) use pixel correspondences in the target and palette images, such as those marked in green (top left), to compute the colour transfer function. Statistical methods (middle) can be applied to images with different content and model the colour distribution of the images using statistical properties. User guided methods (right) rely on user input to recolour images. To recolour the grey scale image of the bird (top right) to look like the ground truth image (top left), the user scribbles colours on the grey scale image which are then used to determine the colours in the result image.

pute a set of spatial correspondences using SIFT, and then apply Gaussian convolution kernels to the target and palette images to compute a set of corresponding colours. The use of Gaussian kernels reduces noise and improves the robustness of the technique. They use lookup tables to indicate corresponding colours and apply these to the target image. A similar technique was implemented by Tehrani et al. [64] and both methods are applied to each colour channel independently. Hwang et al. improve the robustness of the correspondence step by calculating a probability value for each correspondence pair [66]. These values are then used as a weight in the moving least squares step, which computes a non-linear and non-parametric color mapping. The moving least squares transformation is different for each pixel value in the target image and can be very costly for large images. They therefore propose a parallel processing scheme to increase computational efficiency. As the moving least squares transformation is different for each pixel, it allows nearby pixels in the result image to be recoloured differently and unwanted artifacts can appear as a result. This typically happens when

incorrect correspondences are used, indicating that their algorithm is not robust when a large number of pixel correspondences are incorrect.

Region-based Correspondences

Instead of using sparse feature matching, matching regions from both images has also been proposed and can improve the robustness of the results. Wang et al. [67] segment the images using Mean Shift and then carry out feature detection using an optical flow based algorithm. Kagarlitsky et al. [68] use a colour consistency measure to decompose the images into regions and implement histogram matching on the corresponding regions. This quad-tree segmentation can create blocky artifacts which require additional refinement.

Dense Feature Correspondences

In the case where the differences between the images are due to deformation or non-rigid motion, dense correspondence based methods have been proposed. Chen et al. propose to estimate the global disparity and use this to find the dense correspondences [69]. Doutre et al. [70] use block-based methods to determine the disparity between two views. They spatially average colours to extract colour correspondences and use a linear color mapping model.

In [71], Hacohen et al. propose a technique to make photos in a collection more consistent. They use an extension of the Generalised Patch Match algorithm [72] to find dense correspondences in parts of the target and palette images. A global parametric transfer curve for each colour channel can then be defined using these correspondences. This transfer function ensures that colours in the target image, which do not match any colour in the palette image, can still be transformed in a coherent manner. This technique was extended in [73] to deal with a number of images by imposing a graph structure on them. Again, parametric curves were used to apply the colour transformation to the target image.

2.2.2 Statistical based methods

When the target and palette images have different content and direct correspondences are not available, statistical based methods have been proposed to define a mapping

between the images. These methods use statistical properties to describe the colour distribution of both images and propose ways to reshape the colour distribution of the target image so that it matches that of the palette image.

Registration of colour statistical moments

The pioneering work of Reinhard et al. [74] proposed to use a warping function ϕ with parametric form [74]:

$$\phi(x, \theta = \{\mathrm{G}, o\}) = \mathrm{G}\ x + o, \tag{2.9}$$

with the vector o representing an offset and the 3×3 diagonal matrix G representing the gains for each colour channel. The estimation of the parameter θ is performed by registering the probability density functions of the colours in the palette and target images, denoted p_p and p_t respectively, represented as simple multivariate Gaussians ($p_p \equiv \mathcal{N}(x; \mu_p, \Sigma_p)$ and $p_t \equiv \mathcal{N}(x; \mu_t, \Sigma_t)$) with diagonal covariance matrices Σ_p and Σ_t. Since Normal distributions are fully described by their first two statistical moments, means and covariance matrices, the optimal mapping ϕ is specified by the solution $\hat{\theta}$ that maps the empirical estimates of (μ_t, Σ_t) computed using the pixels values $\{x_t^{(i)}\}_{i=1,\cdots,n_t}$ in the target image, to the empirical estimates of (μ_p, Σ_p) computed using the pixels values $\{x_p^{(i)}\}_{i=1,\cdots,n_p}$ in the palette image. This technique performs well when the colour distribution of the target and palette images can be well described by a single Gaussian, but performs poorly when the colour distribution are more complex.

Optimal transport

In the area of optimal transport, for random variables $x \sim p_t(x)$ and $y \sim p_p(y)$, the objective is to find the optimal transport map $\phi(x) = y$ that satisfies

$$p_p(x) = p_t(\phi(x)) \times |\mathrm{det}\nabla\phi(x)| \tag{2.10}$$

and minimises the transportation cost [75, 76]

$$\int_x c(\phi(x) - x)^2\ p_p(x)\ dx. \tag{2.11}$$

where the cost function c meaasures the distance between x and $\phi(x)$. This framework can be applied to the colour transfer problem by letting $p_t(x)$ and $p_p(x)$ represent the colour distributions of the target and palette images respectively. Constraining the map ϕ to minimise the transportation cost ensures that it has some geometric properties that other mappings do not possess, and these prove advantageous when applied to the colour transfer problem. Firstly, this constraint ensures that the minimum number of colour changes are made when matching $p_t(x)$ to $p_p(x)$. It also ensures that $\phi(x)$ has some monotonicity properties which preserve the relative position of colours in the transformed target image. This means that areas of the target image that were bright will remain bright after transformation, and those that were dark will remain dark [76]. Another advantage of using the optimal transport map ϕ is that no strong hypotheses are made about the distributions (as opposed to the Gaussian assumption made by Reinhard et al. [74] described in the previous section). Moreover no parametric form is imposed on the map ϕ.

Finding a solution to equations 2.10 and 2.11 when $x \in \mathbb{R}$ and $y \in \mathbb{R}$ (e.g. grey scale images) can be defined using the cumulative distribution of colours in the target and palette images P_t and P_p:

$$\phi(x) = P_t^{-1} \circ P_p(x), \tag{2.12}$$

However, the problem becomes non trivial in multidimensional colour spaces.

Of particular interest is the pioneering work of Pitié et al. [7] who propose an iterative algorithm that first projects the colour pixels $\{x^{(k)}\}$ onto a 1D Euclidean space, estimates $\hat{\phi}$ using Equation 2.12 and then applies it to move all values $\{x^{(k)}\}$ along the direction of the 1D space. This operation is repeated until convergence. This method increases the graininess of the recoloured target picture, especially when the colour dynamic of the two pictures is very different. A solution to this artifact is proposed later by the authors [6] as a post processing step to ensure the gradient field of the recolored target image is as close as possible to the original target image.

Bonneel et al. [77] recently proposed to use a similar strategy for colour transfer, and their approach is a generalisation of the method proposed by Pitié et al. which uses 1D Wasserstein distances to compute the barycentre of a number of input measures. As well as being used for colour transfer between target and palette images, their method

can also be used to find the barycentre of three or more weighted image palettes.

Optimal transport is now a widely used framework for solving colour transfer as it allows modelling more various, realistic and complex forms for the distributions p_t and p_p than simple multivariate Gaussians and allows for a parameter free form of the warping function ϕ to be estimated. Histograms are often employed to approximate the colour distributions of images [78, 79, 80] and used in optimal transport methods [81, 82, 80]. Similar to the original algorithm proposed by Pitié et al. [7], these discrete methods have a tendency to introduce grainy artifacts in the gradient of the result image. Pitié et al's subsequent extension [6] proposed a post processing step to correct this artifact and ensure the gradient field of the recoloured target image is as close as possible to the original target image. Similarly, recent methods have proposed adding a step to impose that the resulting spatial gradient of the recoloured image remains similar to the target image [83, 79, 82]. Alternatively, other methods have proposed to relax the constraint that enforces the distributions of the recolored target image and palette image to match exactly [81, 80, 82]. Bonneel et al. [77] also propose using a gradient smoothing technique to reduce any quantisation errors that appear in their results [84]. Frigo et al. [85] propose to remove artifacts by first estimating an optimal transport solution and using it to compute a smooth Thin Plate Spline (TPS) transformation to ensure that a smooth parametric warping function is used for recolouring allowing them to apply their method to video content easily.

Using Gaussian Mixture models

Jeong and Jaynes [86] use colour transfer techniques to harmonise the colour distributions of non overlapping images of the same object, for tracking purposes in a multiple camera setting. The colour chrominance (2D) distribution is modelled using GMMs, and the transfer function is parametric with an affine form and is estimated by minimising the Kullbach-Leibler divergence between Gaussian components, using a robust procedure to tackle outlier pairs. Xiang et al. [87] model the colour distribution of the target image using a Gaussian mixture that is estimated by an EM algorithm. Each Gaussian component in the mixture defines a local region in the target image, and each segmented local region is recoloured independently using multiple palette-source images, which are also segmented into regions using GMMs fitted to their colour distri-

butions. Reinhard's [74] transfer technique is then used to perform the colour transfer, by associating the best Gaussian component from the sources to the Gaussian target region. Segmentation using alternative approaches (i.e. Mean Shift [88], K-means [89]) is also tested to define the Gaussian mixtures. This approach relies on homogeneous colour regions, each captured with one multivariate normal in the 3D colour space. Localised colour transfer using Gaussian Mixture Models between overlapping colour images have also been proposed for colour correction, motion deblurring, denoising and gray scale coloring [90]. Using one to one correspondences between Gaussian components capturing the colour content of the target and palette images (or their regions) have also been proposed. Oliveira et al. [91] proposed finding the mapping of 1-dimensional truncated GMM representations, computed for each colour channel of the target and palette images. However, this method is only applicable when correspondences between the target and palette image are available.

2.2.3 User-assisted Methods

In most user assisted methods, the challenge of finding colour correspondences is solved by the user. For example, Oskam et al. [92] propose a method which allows the user to select the colour correspondences manually. The colour mapping is modelled using radial basis functions in the CIELab colour space. Only colours in flat areas are processed while colours at corners or edges are ignored. This creates artifacts in the colour mapping, especially in areas around edges.

Other user assisted methods are sketch based. In this case, the user draws sketches in the image to be colourised, and from these the colours are transfered. Dalmau-Ceden et al. [93] propose a sketch based system which uses a probabilistic segmentation to divide the image into different regions to be coloured. The user sketches are used to define class labels and each pixel is assigned a probability indicating its membership to each of the classes. The image is then coloured using the class probabilities and user sketch colours. Other methods include that of Lischinski et al. [94] who allow users to manually select regions of an image for local tonal adjustment, and An et al. [95] who propagate user tonal adjustment to nearby regions in the image with similar appearance.

2.3 Conclusion

In this chapter, we have given a review of recent techniques in shape registration and colour transfer. In the remainder of this thesis, we propose to further investigate some applications of the $\mathcal{L}_2$ distance metric. This metric has been shown to be robust and can be used when explicit correspondences are unavailable. It also has a closed form solution when computed between Gaussian Mixture Models, a family of density functions that can be used to approximate any probability density function. Inspired by the compelling results that have been achieved using the $\mathcal{L}_2$ distance in the area of shape registration [41], we decided to investigate how this metric would perform when applied to the colour transfer problem (Chapter 4). Many techniques in non-rigid shape registration also use non-linear transformations such as radial basis functions or thin plate splines. We felt that using these transformations would work well with an $\mathcal{L}_2$ distance technique for colour transfer, and their smoothness constraints would ensure that a good recolouring result was generated, and reduce the gradient artifacts typically generated by other colour transfer techniques such as optimal transport.

We also found that many colour transfer techniques are either applied to images with pixel correspondences, or those without pixel correspondences, and many techniques cannot be applied to both image types. Having seen shape registration techniques based on the $\mathcal{L}_2$ distance being applied to point clouds both with and without point correspondences, we decided to investigate whether a colour transfer technique based on the $\mathcal{L}_2$ distance could also be applied to images both with and without pixel correspondences.

For shape registration, we wanted to investigate how normal vectors could be incorporated into the $\mathcal{L}_2$ distance framework to improve registration results. While a technique that incorporates normal vectors into the $\mathcal{L}_2$ framework has been previously proposed [50], the normal vectors were modelled using the Gaussian distribution. While this modelling is sufficient in 2D, it is not suitable when modelling normal vectors in higher dimensions. Having seen the von Mises-Fisher distribution effectively used to model normal vectors in applications such as the clustering of directional data [8], we decided to investigate whether adding an additional kernel to our KDE to model normal vectors using the von Mises-Fisher distribution would give improved results (Chapter 5).

Finally, having investigated the application of the $\mathcal{L}_2$ distance metric to colour transfer, we began to consider whether the optimal transport techniques proposed for colour transfer would prove advantageous to the shape registration problem, and what problems would arise by applying these techniques directly to shape data (Chapter 6).

In the next chapter we will give a brief outline of both colour data and shape data and some of the representations that both data types can have, as well as giving a more in depth outline of the transformation functions regularly used in shape registration and colour transfer.

Chapter 3

Transformation Functions and Data

In this chapter we give a more in depth outline of the different data types and transformation functions that are regularly used in shape registration and colour transfer. In Section 3.1 we give a brief outline of both colour data and shape data, and detail the different representations that these data types can have. In Sections 3.2 and 3.3, a summary of several transformation functions is given, including rigid and affine transformations, radial basis functions, moving least squares, and optimal transport.

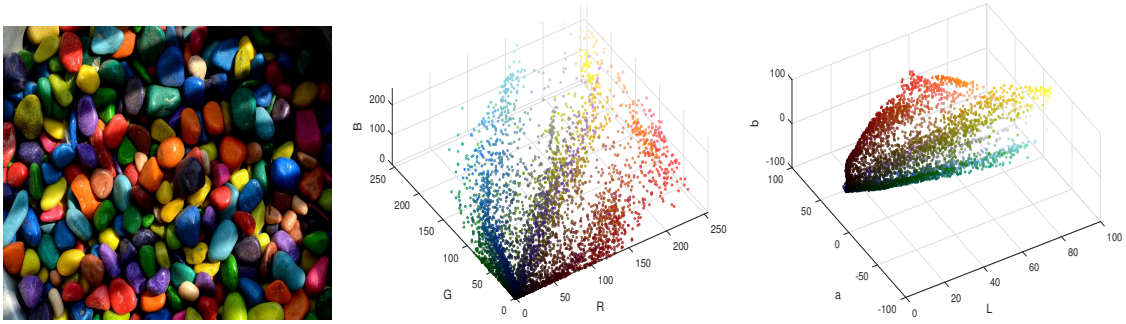
3.1 Data

Data is a set of values which are collected and stored for further analysis or processing. In this section we present the two main types of data that we will analyse in later chapters, colour data and shape data.

3.1.1 Colour Data

As colours are captured by cameras, the most common representations of colour data are images and video. As many applications in computer vision process images and video, colour data is one of the most frequently used data types and is particularly useful for object and face detection, classification, and tracking [96, 97, 98].

A colour is represented as a tuplet of values, typically made up of 3 scalars or colour components. These components can describe aspects of the colour such as hue, brightness and chrominance. These values determine where the colour lies within the colour



(a) (b) RGB (c) CIELab

Figure 3.1: The transformation of each colour in image (a) to RGB space in (b) and CIELab space in (c).

space. A colour space defines a mapping between each triplet of colour components and real world colours. Some common colour spaces include the RGB, CIELab, HSL and YUV colour spaces. Here we will focus on the RGB and CIELab colour spaces, as we will investigate how our proposed colour transfer technique performs in both colour spaces in Chapter 4.

RGB (Red-Green-Blue)

An RGB colour space is an additive colour space, with each RGB colour c defined by three values, its red, green, and blue component:

$$c = (R, G, B), \tag{3.1}$$

typically defined with $0 \leq R \leq 255$, $0 \leq G \leq 255$ and $0 \leq B \leq 255$. In Figure 3.1(b) we present the transformation of each colour in an image to the RGB colour space. This colour model is commonly used to encode and store colour images and video, and to represent the colour of pixels on most monitors and televisions. One of the main drawbacks of using an RGB colour space is the high correlation between its colour components. This is due to the fact that each colour component includes some representation of the colour's brightness. This can be seen in Figure 3.2. The linear shadows appearing in the colour image can be seen in each colour channel. Another problem is that two colours which are perceived to be quite similar can differ by a large

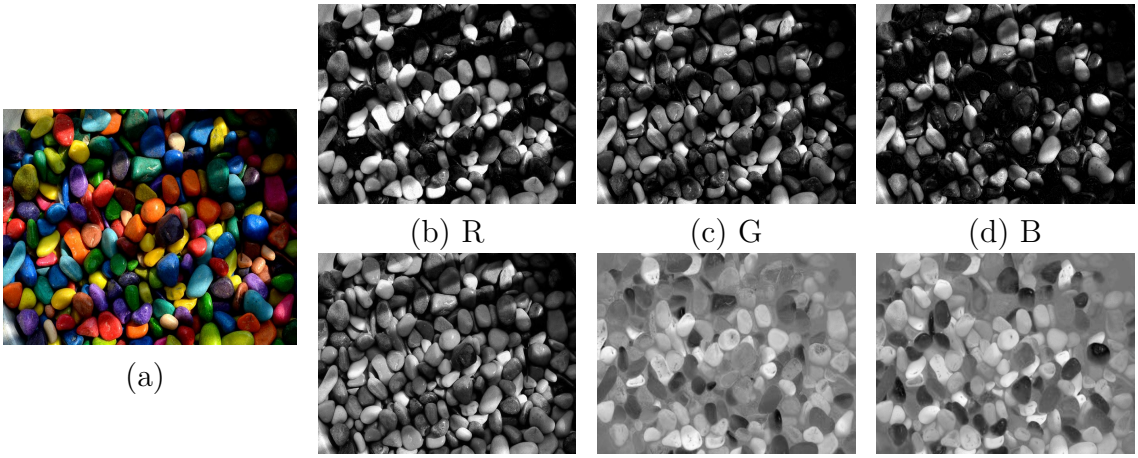


Figure 3.2: Colour space transformations of coloured image in (a): (b - d) RGB colour channels; (e - g) CIELab colour channels. Note that the L,a and b colour values have been rescaled for visualisation. While the linear shadows can be seen in each colour channel in the RGB space, these brightness changes can only be seen in the L channel in CIELab space.

Euclidean distance within the RGB space [99, 100].

CIELab

The CIELab colour space, on the other hand, tries to accurately represent how humans perceive differences in luminance and chrominance, which means that the Euclidean distance between two colours in CIELab space is strongly correlated to their perceived similarity. It is based on the CIE system, which bases colour specification on the human visual system. A colour c in CIELab space is defined by three values:

$$c = (L, a, b), \tag{3.2}$$

where L represents the luminance of the colour, and a and b represent its chrominance. The value L is typically defined for $0 \leq L \leq 100$, while the range of values for which a and b are defined is dependent on the conversion used to compute the CIELab coordinates. In Figure 3.1(c) we present the transformation of each colour in an image

to the CIELab colour space.

The conversion from the RGB representation (R, G, B) to CIELab (L, a, b) representation is non-linear in nature. The first step involves applying a linear transformation to the RGB coordinates [101]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49 & 0.31 & 0.20 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00 & 0.01 & 0.99 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{3.3}$$

Then, choosing a reference white value as $(Xn, Y_n, Z_n) = (0.9642, 1, 0.8249)$ and defining the function $f(t)$ as:

$$f(t) = \begin{cases} t^{1/3} & \text{if } t > \delta^3 \\ t/(3\delta^2) + 2\delta/3 & \text{otherwise} \end{cases} \tag{3.4}$$

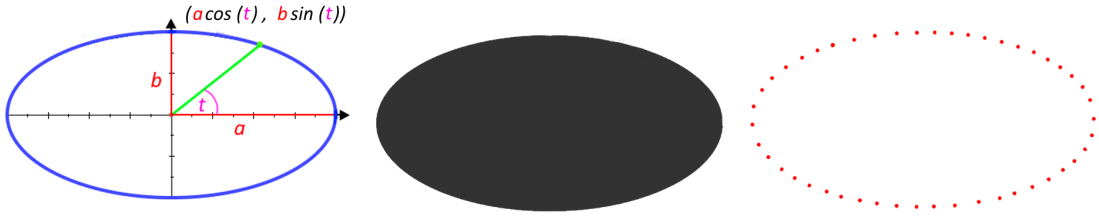
where $\delta = 6/29$, we can compute (L, a, b) as follows [101]:

$$\begin{aligned} L &= 166\left[f\left(\frac{Y}{Y_n}\right)\right] \\ a &= 500\left[f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right] \\ b &= 200\left[f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right] \end{aligned} \tag{3.5}$$

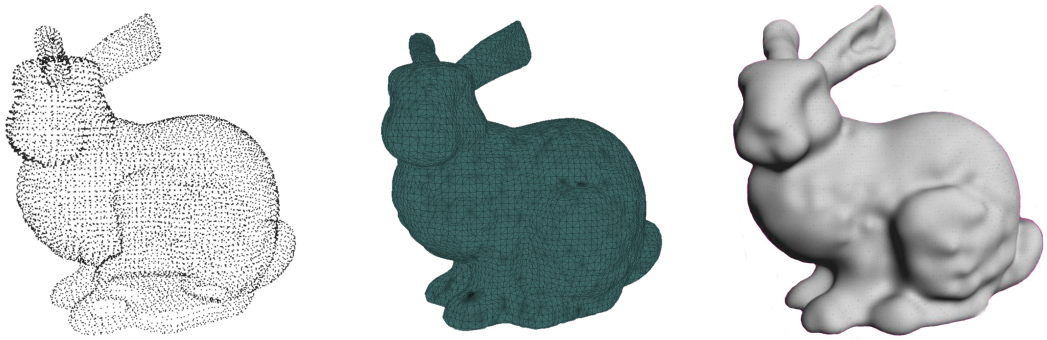
As the L colour channel represents the luminance in an image, while the a and b colour channels represent the chrominance, in Figure 3.2 we can see that the linear shadows in the colour image only appear in the luminance channel L and not in a or b.

3.1.2 Shape data

The shape of an object refers to its form, outline, external boundary, or external surface. Common shapes include lines, curves, circles, ellipses, planes and cubes and can be described mathematically using a parametric representation. The parametric representation of a shape is a set of equations which define the coordinates of all points



(a) Parametric Equation (b) Binary image (c) Point cloud



(a) Point Cloud (b) Polygonal mesh (c) Implicit surface

Figure 3.3: Several different shape representations for 2D and 3D shape data are shown. (a) Parametric equation representing an ellipse; (b) Binary image of ellipse; (c) Point cloud sampling the edge contour; (d) Point cloud representation of 3D shape; (e) Mesh representation with vertices, edges and faces; (f) Implicit surface representation.

on the shape, and each representation depends on a set of parameters. For example, a parametric representation of an ellipse in $\mathbb{R}^2$ is dependent on the parameter t and is given by:

$$\begin{aligned} x(t) &= a\ \cos(t) \\ y(t) &= b\ \sin(t) \end{aligned} \tag{3.6}$$

with $t \in (0, 2\pi)$, $a \in \mathbb{R}$ and $b \in \mathbb{R}$, as shown in Figure 3.3(a).

In computer vision, we observe shape using sensors such as cameras and depth scanners, and capture discrete versions of shape such as images and point clouds. 2D shapes can be captured by images and generally represent a 2-dimensional view of a 3-dimensional object in the real world. 3D shape data can be captured directly using 3D scanners or computed from several images. In order to analyse the shape data, a suitable representation must be chosen, and shape descriptors can then be calculated for further analysis in applications such as shape classification or comparison [102].

Shape Representation

There are several different ways shapes can be represented once they have been captured. For 2D shapes, contour based techniques can be used to represent the shape boundary. One method is to store the shape contour as a string of the form:

$$S = s_1, s_2, .., s_n \tag{3.7}$$

where s_i may correspond to an element of chain code, a spline or an arc [102]. The contour could also be represented mathematically as a parametric curve, such as the example given in Equation 3.6, or as a point cloud. A 2D point cloud corresponds to a set of points, specified by a tuple (x, y), defined on an orthogonal coordinate system. Unlike contour based techniques, region based methods take into account all pixels within the shape, and include point clouds sampled from the entire shape, or binary images [102]. Some examples can be seen in Figure 3.3.

Samples of 3D shape representations are polygonal meshes, parametrised surfaces or patches of parametrised surfaces, implicit surfaces, NURB surfaces, voxel structures or point clouds [103], as in Figure 3.3. The most common representation of a 3D shape is the polygonal mesh. A mesh is a set of vertices, edges and faces and corresponds to a structured representation of a point cloud and is frequently used to store objects in computer graphics and computer games. Several mesh formats exist, including the .ply, .obj, and .stl formats.

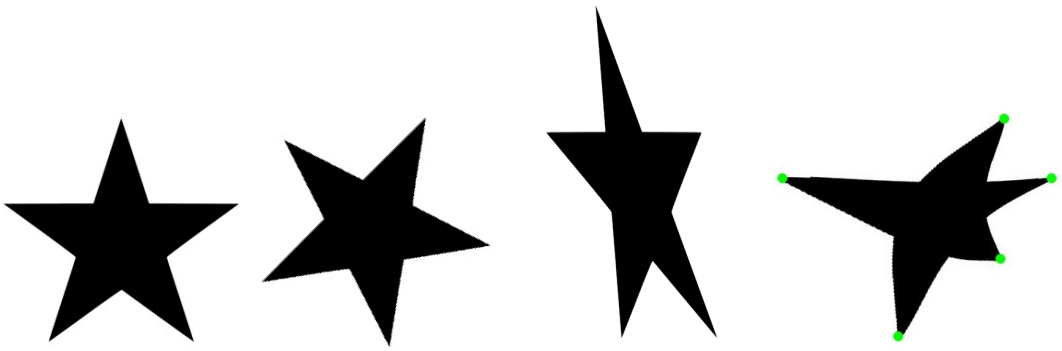
The .ply mesh format describes an object using vertices and faces. Each vertex is described as an [x,y,z] tuple and each face is made up of n vertices. A face is defined as n indices in the vertex list. The .ply format can also store characteristics such as vertex colour, normal vectors, texture coordinates, transparency and vertex confidence. It also allows users to define and store their own elements such as edges or ambient colour. A .ply mesh can be stored in ASCII or binary format, with the binary version being more compact in terms of storage and allows for rapid reading and writing to and from memory. A sample .ply defining the shape of a cube in ASCII format is shown in Figure 3.4.

```
ply
format ascii 1.0           { ascii/binary, format version number }
comment made by Greg Turk  { comments keyword specified, like all lines }
comment this file is a cube
element vertex 8           { define "vertex" element, 8 of them in file }
property float x           { vertex contains float "x" coordinate }
property float y           { y coordinate is also a vertex property }
property float z           { z coordinate, too }
element face 6             { there are 6 "face" elements in the file }
property list uchar int vertex_index { "vertex_indices" is a list of ints }
end_header                 { delimits the end of the header }
0 0 0                      { start of vertex list }
0 0 1
0 1 1
0 1 0
1 0 0
1 0 1
1 1 1
1 1 0
4 0 1 2 3                  { start of face list }
4 7 6 5 4
4 0 4 5 1
4 1 5 6 2
4 2 6 7 3
4 3 7 4 0
```

Figure 3.4: Sample .ply file for a 3D cube written in ASCII format [4].

Shape Descriptors

Shape descriptors are simplified representations of 2D or 3D shapes and include shape features such as curvature, area and geometric ratios. They can be defined as either local or global descriptors, depending on whether they are computed on the whole shape region or just the shape boundary, and are often used to classify or compare shapes [102]. The neighbourhood structure of a shape can be summarised by many shape descriptors including shape context, which has been used to compute point correspondences between shapes [30, 51]. The normal vectors of a 3D mesh have also been shown to provide most of the shape and structural information of an object and descriptors derived from them have been proposed for object detection and classification in [104, 105].



(a) Original (b) Rotation (c) Affine (d) TPS

Figure 3.5: Several transformations applied to the 2D star shape in (a), (b) a rotation (c) an affine transformation including scaling and sheering and (d) a TPS transformation controlled by the 5 control points marked in green.

3.2 Transformation $\phi(x)$ with $x \in \mathbb{R}^d$

A transformation defined for $x \in \mathbb{R}^d$ is a function ϕ which transforms the point x to a new value $y \in \mathbb{R}^d$, such that $y = \phi(x)$. The function ϕ can be defined as either parametric or non-parametric. Parametric transformations can be defined using parametric equations while non-parametric transformations typically apply an independent transformation to each point x. Functions of each type will be outlined in Sections 3.2.1 and 3.2.2.

3.2.1 Parametric Transformations

Parametric transformations $\phi(x, \theta)$ are dependent on some parameter θ and include rigid, affine and spline transformations such as radial basis functions or b-splines. Some example parametric transformations applied to a 2D shape can be seen in Figure 3.5.

Rigid Transformations

A rigid transformation in $\mathbb{R}^d$ with $\phi(x, \theta) = y$ preserves the distance between pairs of points $x, y \in \mathbb{R}^d$ and includes rotations and translations. When $d = 2$, a rotation

transformation can be expressed as $\phi(x, \theta) = \mathrm{R}(x)$ with:

$$\mathrm{R} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{3.8}$$

controlled by the angle θ.

When $d = 3$, the rotation matrix R is controlled by the angles $\theta = (\alpha, \beta, \gamma)$, with each angle representing the degree of rotation about a given axis. These are known as Euler angles, and there are several conventions which dictate which axis the angles α, β and γ are associated with. One of the most common conventions lets α represent the rotation about the x-axis, β the rotation about the y-axis and γ the rotation about the z-axis. The rotation matrix R is then given by [106]:

$$\mathrm{R} = \begin{bmatrix} \cos\beta\cos\gamma & \cos\beta\sin\gamma & -\sin\beta \\ \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \cos\beta\sin\alpha \\ \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\beta\cos\alpha \end{bmatrix}. \tag{3.9}$$

Singularities arising from gimbal lock are found in various Euler angle representations. Gimbal lock occurs when the Euler angle β is at some critical value and the two other Euler angles α and β become indistinguishable. For example, when $\beta = \frac{\pi}{2}$, R in Equation 3.9 becomes:

$$\mathrm{R} = \begin{bmatrix} 0 & 0 & 1 \\ \sin(\alpha-\gamma) & \cos(\alpha-\gamma) & 0 \\ \cos(\alpha-\gamma) & -\sin(\alpha-\gamma) & 0 \end{bmatrix}. \tag{3.10}$$

Since row 1 and column 3 of R are invariant to changes in α and γ, R represents a rotation of $(\alpha - \gamma)$ in a single direction, meaning our rotation has lost a degree of freedom. For R given in Equation 3.9, singularities occur at $\beta = \frac{\pi}{2} + n\pi$, for $n \in \mathbb{Z}$.

A rotation transformation $\phi(x, \theta)$ in $\mathbb{R}^d$ can also be written as a unit quaternion $q \in \mathbb{S}^3$. Unit quaternions are a very popular representation as they lack any singularities and are therefore easier to estimate when using iterative optimisation algorithms such as gradient ascent. During optimisation the estimated quaternions can be renormalised at every iteration or an added term such as $\cos(1 - \|q\|)^2$ can be added to the objective function to prevent large violations of the unit normal constraint.

To convert the Euler angles $\theta = (\alpha, \beta, \gamma)$ to their unit quaternion representation q we use the following equation [106]:

$$q = \begin{bmatrix} \cos\frac{\alpha}{2}\cos\frac{\beta}{2}\cos\frac{\gamma}{2} + \sin\frac{\alpha}{2}\sin\frac{\beta}{2}\sin\frac{\gamma}{2} \\ -\cos\frac{\alpha}{2}\sin\frac{\beta}{2}\sin\frac{\gamma}{2} + \cos\frac{\beta}{2}\cos\frac{\gamma}{2}\sin\frac{\alpha}{2} \\ \cos\frac{\alpha}{2}\cos\frac{\gamma}{2}\sin\frac{\beta}{2} + \sin\frac{\alpha}{2}\cos\frac{\beta}{2}\sin\frac{\gamma}{2} \\ \cos\frac{\alpha}{2}\cos\frac{\beta}{2}\sin\frac{\gamma}{2} - \sin\frac{\alpha}{2}\cos\frac{\gamma}{2}\sin\frac{\beta}{2} \end{bmatrix}. \tag{3.11}$$

Using these equations for R, a rigid transformation in $\mathbb{R}^d$, controlled by the parameter θ, can be written as

$$\phi(x, \theta) = \mathrm{R}x + t \tag{3.12}$$

where R is the rotation matrix, t is a d-dimensional translation vector and $\theta = (R, t)$.

Affine Transformations

An affine transformation in $\mathbb{R}^d$ preserves parallel lines and ratios of distances between points lying on a line. It includes rotation, translation, dilation, shearing and reflection. This type of transformation has the following form:

$$\phi(x, \theta) = \mathrm{A}x + t, \tag{3.13}$$

where A is a $d \times d$ affine matrix, t is a d-dimensional translation vector and $\theta = (\mathrm{A}, t)$.

Radial Basis Functions

A Radial Basis Function (RBF) $\psi : \mathbb{R}^+ \to \mathbb{R}$ is a real valued function which is radially symmetric about some control point $c \in \mathbb{R}^d$. In other words, it effects all points of the same distance from c equally. For a point $x \in \mathbb{R}^d$ we compute $\psi(r)$ as:

$$\psi(r) = \psi(\|x - c\|), \tag{3.14}$$

with $r = \|x - c\|$ and $\|\cdot\|$ denoting the Euclidean norm on $\mathbb{R}^d$. The most commonly used radial basis functions include the Gaussian, Multiquadric and Thin Plate Spline (TPS) functions, all of which are summarised in Table 3.1. The parameter ϵ appears in all equations but TPS, and is known as the shape parameter.

RBF name	**RBF equation**
TPS	$\psi(r) = r^2 log(r)$ (2D)
	$\psi(r) = -(r)$ (3D)
Gaussian	$\psi(r) = e^{-(\epsilon r)^2}$
Inverse Multiquadric	$\psi(r) = \frac{1}{\sqrt{1+(\epsilon r)^2}}$
Inverse Quadric	$\psi(r) = \frac{1}{1+(\epsilon r)^2}$
Multiquadric	$\psi(r) = \sqrt{1+(\epsilon r)^2}$

Table 3.1: The equations for some commonly used radial basis functions.

Finite linear combinations of RBFs are frequently used and take the form:

$$\phi(x, \theta) = \sum_{i=1}^{N} w_i \ \psi(\|x - c_i\|) \tag{3.15}$$

with $x \in \mathbb{R}^d$, weights $w_i \in \mathbb{R}^n$, and control points $c_i \in \mathbb{R}^d$. For a given set of control points, the parameter $\theta = W = (w_1, w_2, \cdots, w_N)^T$ determines the transformation ϕ. This form of transformation is known as a pure radial sum. The drawback of using pure radial sums is that this type of transformation has little effect on points x which are far away from the control points c_i. It is also impossible to generate an affine transformation with this type of function. For this reason an affine transformation is typically added to ϕ as follows:

$$\phi(x, \theta) = \mathrm{A}\ x + t + \sum_{i=1}^{m} w_i \ \psi(\|x - c_i\|) \tag{3.16}$$

where A is an affine transformation matrix, t is a translation vector and $\theta = (\mathrm{A}, t, W)$.

Radial basis functions are often used for function approximation or data interpolation. In these cases, the parameter θ is estimated by fitting a linear combination of RBFs, as in Equations 3.15 or 3.16, to scattered data. For example, given the scattered data

$$(x_i, f_i)_{i=1,..N}, \qquad x_i \in \mathbb{R}^d, \qquad f_i \in \mathbb{R}, \tag{3.17}$$

the pure radial sum approximation $\phi(x, \theta)$, with control points $c_i = x_i$, is given by:

$$\phi(x, \theta) = \sum_{i=1}^{N} w_i \ \psi(\|x - x_i\|), \ x \in \mathbb{R}^d, \ w_i \in \mathbb{R}, \tag{3.18}$$

can be estimated by minimising the cost function [107]

$$\mathcal{C}(\theta) = \sum_{i=1}^{N} (\phi(x_i, \theta) - f_i)^2 \tag{3.19}$$

Data interpolation is possible when $\Psi W = F$ has a unique solution, where

$$\Psi = \{\psi_{ij}\} = \psi(\|x_i - c_j\|), \ W = (w_1, w_2, \cdots, w_N)^T, \ F = (f_1, f_2, \cdots, f_N)^T. \tag{3.20}$$

In this case the weights W are typically estimated by solving $\Psi W = F$ using linear least squares.

The Thin Plate Spline basis function is unique in that when used to compute the transformation ϕ, it minimises the bending energy

$$\int_{\mathbb{R}^d} \|D^2 \phi(x, \theta)\|^2 dx. \tag{3.21}$$

Therefore, similar to the rigidity found in a thin metal sheet, the transformation ϕ consisting of TPS basis functions resists bending [108]. In this case ϕ typically contains an affine part, as described in Equation 3.16, and additional constraints are added to the estimated weights w_i as follows:

$$\sum_{i=1}^{N} w_i = 0 \text{ and}$$

$$\sum_{i=1}^{N} w_i x_{i1} = \sum_{i=1}^{N} w_i x_{i2} = \ \cdots \ \sum_{i=1}^{N} w_i x_{id} = 0$$

These constraints ensure that the matrix Ψ is not singular and that a solution for θ can be estimated [108, 109].

The bending energy proposed in Equation 3.21 can be computed for any RBF transformation ϕ to determine how much bending is applied, and is proportional to

$W^T \Psi W$ [41]. This term is often added as a regularisation term to the cost function $\mathcal{C}(\theta)$ as follows:

$$\mathcal{C}(\theta) = \sum_{i=1}^{N} (\phi(x_i) - f_i)^2 + \lambda W^T \Psi W, \tag{3.22}$$

and controls the non-rigidity of the estimated transformation controlled by θ. Here the parameter $\lambda \geq 0$ controls the strength of the regularisation. Setting $\lambda = 0$ reduces the transformation to exact interpolation.

B-Splines

A B-spline, or basis spline, $P_k(x)$ of order k, is defined as a linear combination of $n+1$ control points c_i, and B-spline basis functions $N_{i,k}(x)$ as follows:

$$P_k(x) = \sum_{i=0}^{n} c_i N_{i,k}(x), \;\; x \in [t_k, t_{n+2}], \tag{3.23}$$

with $2 \leq k \leq (n+1)$, and $x = t_1, ...t_{n+1+k}$ the $n+k+1$ knots associated with this curve. At the knots the basis functions are C^{k-2} continuous. B-splines can be thought of as a local transformation as they are non-zero on a finite number of adjacent intervals. They have been proposed for both non-rigid image registration [110, 111] and 3D shape registration [112] as they are smooth, have compact support and the locally supported basis functions are computationally efficient [110, 113]. Measures are sometimes required however to ensure that folding of the deformation field is avoided [114, 115, 110].

Neural Networks

Artificial Neural Networks (ANN) are inspired by the neural networks in the brain and consist of layers of processing units which work together to solve problems. Each processing unit outputs a non-linear transformation of its input data and as a result, an ANN can give rise to very powerful non-linear transformations. Deep Neural Networks (DNN) consist of ANNs with multiple layers of processing units and can model complex non-linear relationships. They have the ability to derive meaning from complex data and have been shown to be very successful in the areas of object and face recognition, performing at near human accuracy [116, 117]. DNNs are trained for specific tasks

and in order to generate state of the art results they typically estimate hundreds of parameters. They also require large data sets for training and can take a long time to train. In image processing the most powerful class of Deep Neural Network are called Convolution Neural Networks (CNNs). In CNNs each processing unit can be thought of as an image filter which extracts a particular feature from the input image. CNNs have been proposed for painting style transfer, image colourisation, image recognition and video classification [118, 119, 120, 121, 122].

3.2.2 Non-parametric Transformations

Non-parametric transformations $\phi(x)$ cannot be expressed as a parametric equation, with the transformation typically depending on $x \in \mathbb{R}^d$, the point being transformed. Non-parametric transformations include piece-wise linear transformations, moving least squares, and optimal transport maps.

Piecewise Linear

A piecewise linear transformation is composed of several rigid transformations defined over a number of intervals. They are used frequently in image registration, whereby the target image is divided into subimages or blocks which are individually registered to the target image [123, 124]. However, the transformation is not necessarily continuous and many approaches implement a regularisation step to solve this problem, for example using low pass filtering. Issues also arise when the size of the subimages or blocks are too small and the information available in each block is insufficient to drive the registration [113, 125].

Moving Least Squares

Another non-parametric transformation method is Moving Least Squares (MLS). Given two set of points $\{x_i\}$ and $\{y_i\}$ where each x_i and y_i are corresponding pairs, MLS is used to estimate the transformation that maps $\{x_i\}$ to $\{y_i\}$. Given a point $x \in \mathbb{R}^d$, we solve for the optimal transformation $\phi(x)$ that minimises:

$$\sum_i w_i(x) \|\phi(x_i) - y_i\|^2 \tag{3.24}$$

where $\phi(x)$ is either an affine or rigid transformation of the form $\phi(x) = \mathrm{A}x + t$, and

$$w_i(x) = \|x_i - x\|^{2\alpha}. \tag{3.25}$$

The weights in this least squares problems are dependent on the evaluation point x, and therefore so too is the transformation $\phi(x)$. This is where the name Moving Least Squares comes from. In Equation 3.25, α is a fall-off parameter which controls the amount of influence far away points have on the deformation. Therefore MLS is a local deformation method, meaning only the points which are close to x are taken into account.

In computer graphics, MLS has been used to improve the quality of noisy point sets by replacing them with a smoother interpolated version [126, 127]. An MLS formulation has also been used to derive implicit surface functions from polygon data [128]. MLS is frequently used in 2D image registration [129], and a probabilistic version of the MLS technique has been recently proposed for registering the colour distributions of images which have many correspondences [5].

Optimal Transport

Optimal transport methods estimate the optimal transport map ϕ which transforms a point x to a point y with minimal transportation cost, as described in Section 2.2.2. There is no explicit parametric expression for ϕ, instead a set of correspondences $(x, \phi(x))$ are estimated. In colour transfer applications optimal transport maps are used to recolour a colour pixel value x to $\phi(x)$ [6, 77]. The constraint on ϕ to minimise the transportation cost ensures that the minimal amount of colour changes are introduced when matching the target to the palette image. The map ϕ also has some nice monotonicity proporties that ensure that areas that were bright in the target image remain bright after transformation, and areas that were dark remain dark [76]. Typically the colour distribution of the target and palette images are estimated using histograms and the optimal transport map can be computed as the inverse cumulative distribution of the palette image [81, 82, 80]. Optimal transport methods have also been used for non-rigid registration of 2D and 3D images. In this case the transport map typically takes the form of a deformation vector field [130, 131, 132].

3.3 Transformation $\phi(u)$ with $u \in \mathbb{S}^d$

The hypersphere $\mathbb{S}^d$ of radius r, and embedded in Euclidean space of dimension $(d+1)$, is defined as

$$\mathbb{S}^d = \{x \in \mathbb{R}^{d+1} \; ; \; \|x\| = r\}. \tag{3.26}$$

A transformation ϕ which is defined on the non-Euclidean space $\mathbb{S}^d$ maps a point $u \in \mathbb{S}^d$ to a new value $v \in \mathbb{S}^d$, such that $v = \phi(u)$. The transformations outlined in Section 3.2 are defined in Euclidean space and when applied to points $u \in \mathbb{S}^d$, often generate points $\phi(u)$ such that $\phi(u) \notin \mathbb{S}^d$. One transformation which is defined for both $u \in \mathbb{S}^d$ and $x \in \mathbb{R}^d$, where $d \in \{1, 2\}$, is the rotation transformation described in Section 3.2.1. In Chapter 5 we will apply this transformation to the normal vectors of both 2D and 3D shapes, which are defined on $\mathbb{S}^1$ and $\mathbb{S}^2$ respectively.

3.4 Conclusion

In this chapter we summarised both colour data and shape data, and outlined some of the most common representations for each data type. We also detailed some transformations defined for points $x \in \mathbb{R}^d$ and $u \in \mathbb{S}^d$. In the remainder of this thesis we will model both shape data and colour data using mixture models and propose methods to estimate a transformation between two probability density functions. In the next chapter we will investigate how this method can be used to solve the colour transfer problem.

Chapter 4

$\mathcal{L}_2$ Registration for Colour Transfer

Colour transfer refers to a set of techniques that aim to modify the colour feel of a target image or video using an exemplar colour palette provided by another image or video. Most techniques are based on the idea of warping some colour statistics from the target image colour distribution to the palette image colour distribution. The transfer (or warping) function ϕ, once estimated, is then used to recolour a colour pixel value x to $\phi(x)$. In this chapter we present a colour transfer technique (Section 4.1) which can be applied to both images of similar and different content, can be enhanced by pixel correspondences, and provides a computationally efficient and convenient recolouring tool for users. An exhaustive set of experiments has been carried out to assess performance against leading techniques in the field (Section 4.2) including computational time needed for recolouring. We also show the usability of our approach for creating visual effects (Section 4.3) and conclude (Section 4.4). The work presented in this chapter has been published in [11, 10] and is currently under submission in [9].

4.1 Robust Colour Transfer

As described in Section 2.2.2, Pitié et al. [7] propose an optimal transport method for colour transfer, which estimates a colour transfer function ϕ by estimating several 1-dimensional optimal transport mapping. They have shown that their algorithm iteratively decreases the Kullbach Leibler divergence between the probability density functions p_t and p_p [6], proving that their estimated transfer function ϕ brings the

colour distribution of the target image close to the colour distribution of the palette image. However, no parametric formulation of the solution $\hat{\phi}$ is available (instead, for each pixel $x_t^{(k)}$ a corresponding value $\hat{\phi}(x_t^{(k)})$ is calculated), limiting the possibility of applying this transformation to a previously unseen value x, and also limiting direct manipulation of the estimated warping function $\hat{\phi}$. As an improvement, Frigo et al. [85] propose to fit a smooth Thin Plate Spline (TPS) transformation to the optimal transport solution for colour transfer.

As an alternative to the optimal transport framework, we propose to formulate the colour transfer problem as a shape registration one, whereby a parametric warping function is directly estimated by minimising the $\mathcal{L}_2$ distance between two GMMs which capture the colour content of the palette and target images. We also take advantage of correspondences that can be defined between pixels in the target image to be recoloured, and the pixels in the palette image, which is used as an exemplar. When considering target and palette images of the same scene, correspondences can easily be computed using registration techniques, potentially creating some outlier pairs, and our technique is shown to be robust to these occurrences. We explore several clustering techniques for defining the GMMs for palette and target images before registration. Affine and radial basis functions are also tested for modelling the warping function ϕ (Section 4.1.5). Finally recolouring using the estimated warping function ϕ is implemented using parallel programming and we show that our approach is currently the fastest for recolouring (Section 4.1.7).

4.1.1 GMM representation of colour content

We define two Gaussian mixture models $p_t(x)$ and $p_p(x)$ which capture the colour content of the target and palette images respectively, with $p_p(x)$ defined as:

$$p_p(x) = \sum_{k=1}^{K_p} \mathcal{N}(x; \mu_p^{(k)}, \Sigma_p^{(k)})\ \pi_p^{(k)} \tag{4.1}$$

and $p_t(x)$ defined as:

$$p_t(x) = \sum_{k=1}^{K_t} \mathcal{N}(x; \mu_t^{(k)}, \Sigma_t^{(k)})\ \pi_t^{(k)} \tag{4.2}$$

Here K_p and K_t represent the number of Gaussians in the respective mixtures, $\{\mu_p^{(k)}\}_{k=1,..K_p}$ and $\{\mu_t^{(k)}\}_{k=1,..K_t}$ the Gaussian means, and $\{\Sigma_p^{(k)}\}_{k=1,..K_p}$ and $\{\Sigma_t^{(k)}\}_{k=1,..K_t}$ the covariance matrices. The coefficients $\{\pi_p^{(k)}\}_{k=1,..K_p}$ and $\{\pi_t^{(k)}\}_{k=1,..K_t}$ are positive weights, with each set summing to one and capturing the relative importance of each Gaussian component in the two mixtures.

In order to transform the target distribution $p_t(x)$ to match the palette distribution $p_p(x)$, $p_t(x)$ is changed to a parametric family of distributions $p_t(x|\theta)$, with θ the parameter controlling the transformation ϕ which maps $p_t(x|\theta)$ to $p_p(x)$. One method of defining $p_t(x|\theta)$ involves applying the transformation ϕ to the variable x, as in [76], and defining a Gaussian mixture model of the form:

$$p_t(\phi(x,\theta)|\theta) = \sum_{k=1}^{K_t} \mathcal{N}(\phi(x,\theta); \mu_\phi^{(k)}, \Sigma_\phi^{(k)})\ \pi_t^{(k)} \tag{4.3}$$

Here $\{\mu_\phi^{(k)}\}$ and $\{\Sigma_\phi^{(k)}\}$ represent the means and covariance matrices of the Gaussian components, and have been computed by transforming $\{\mu_t^{(k)}\}$ and $\{\Sigma_t^{(k)}\}$ by some transformation which depends on ϕ. When ϕ is an affine transformation, $\phi(x,\theta)$ can be modelled as a Gaussian distribution with $\{\mu_\phi^{(k)} = \phi(\mu_t^{(k)})\}$ and $\{\Sigma_\phi^{(k)} = \phi\Sigma_t^{(k)}\phi^T\}$. However, when ϕ is a non-linear transformation, $\phi(x,\theta)$ may not follow a Gaussian distribution, and although techniques have been proposed to approximate suitable values for $\{\mu_\phi^{(k)}\}$ and $\{\Sigma_\phi^{(k)}\}$ [133], there is no guarantee that $\mathcal{N}(\phi(x,\theta); \mu_\phi^{(k)}, \Sigma_\phi^{(k)})$ will integrate to 1 and thus may not represent a true distribution. Then, when computing the $\mathcal{L}_2$ distance between $p_t(x|\theta)$ and $p_p(x)$, the cross product term $\langle p_t(x|\theta)|p_p(x)\rangle$, which can be written explicitly as:

$$\langle p_t(x|\theta)|p_p(x)\rangle = \sum_{k=1}^{K_t}\sum_{l=1}^{K_p} \int \mathcal{N}(\phi(x,\theta); \mu_\phi^{(k)}, \Sigma_\phi^{(k)})\ \ \mathcal{N}(x; \mu_p^{(l)}, \Sigma_p^{(l)})\ \pi_t^{(k)}\ \pi_p^{(l)} dx \tag{4.4}$$

may not be easily computed.

As an alternative to transforming the variable x, we instead propose to transform

only the means of the Gaussians $\{\mu_t^{(k)}\}$. The distribution $p_t(x|\theta)$ is then given by:

$$p_t(x|\theta) = \sum_{k=1}^{K_t} \mathcal{N}(x; \phi(\mu_t^{(k)}, \theta), \Sigma_t^{(k)})\ \pi_t^{(k)}. \tag{4.5}$$

In this case $p_t(x|\theta)$ is guaranteed to be a distribution for all transformations ϕ and the computation of $\langle p_t(x|\theta)|p_p(x)\rangle$ is straight forward.

We also choose isotropic identical covariance matrices for both the target and palette GMMs, controlled by a bandwidth h:

$$\Sigma_t = \Sigma_p = h^2\mathrm{I} \tag{4.6}$$

with I the identity matrix. By using the modelling proposed in Equation 4.5 as opposed to that proposed in Equation 4.3, we ensure that the covariance matrix associated with each Gaussian component remains spherical after transformation. Therefore, rather than capturing the true distribution of the transformed target image, we instead approximate it, giving less emphasise to the spread of the colours associated with each Gaussian component, and modelling only how the transformation effects the colours captured by the Gaussian means. As target and palette images do not have the same visual content in general, other colour transfer methods have also proposed approximating the true colour distribution of the images and have shown improved results using this approach [85].

Computing $\mathcal{L}_2$

As described in Section 2.1.5, the $\mathcal{L}_2$ distance between $p_t(x|\theta)$ and $p_p(x)$, denoted p_t and p_p for brevity, can be written as

$$\mathcal{L}_2(p_t, p_p) = \|p_t\|^2 - 2\langle\ p_t\ |p_p\ \rangle + \|p_p\|^2 \tag{4.7}$$

We propose to minimise $\mathcal{L}_2(p_t, p_p)$ in order to estimate θ. Since the term $\|p_p\|^2$ is independent of θ, it can be removed from our cost function and θ can be estimated by minimising

$$\|p_t\|^2 - 2\langle\ p_t\ |p_p\ \rangle. \tag{4.8}$$

The scalar product term between p_t and p_p in the $\mathcal{L}_2$ distance equation is given by:

$$\langle p_t | p_p \rangle = \sum_{k=1}^{K_t} \sum_{l=1}^{K_p} \mathcal{N}(0; \phi(\mu_t^{(k)}, \theta) - \mu_p^{(l)}, \Sigma_t^{(k)} + \Sigma_p^{(l)})\ \pi_t^{(k)}\ \pi_p^{(l)} \tag{4.9}$$

while that between p_t and p_t is given by:

$$\|p_t\|^2 = \sum_{k=1}^{K_t} \sum_{l=1}^{K_t} \mathcal{N}(0; \phi(\mu_t^{(k)}, \theta) - \phi(\mu_t^{(l)}, \theta), \Sigma_t^{(k)} + \Sigma_t^{(l)})\ \pi_t^{(k)}\ \pi_t^{(l)} \tag{4.10}$$

Jian et al. use the $\mathcal{L}_2$ distance to register shapes encoded as Gaussian Kernel Density Estimates [41]. Using this idea for our colour transfer application would mean that the number K of Gaussians in the mixtures (Eq. 4.1 and 4.5) is the number of pixels in the image (target or palette), and the cluster centres $\mu^{(k)} = x^{(k)}$ are the colour pixels $x^{(k)}$ in the image. In this case K is extremely large and our cost function becomes too computationally intensive to be practical. As an alternative, two algorithms for clustering, K-means and Mean Shift, are proposed to compute the means (Sections 4.1.2 and 4.1.3). When correspondences are available between palette and target images, a third method for setting the means is proposed in Section 4.1.4.

Weights

The natural choice for setting the weights $\pi^{(k)}$ is to choose the proportion of pixels in the image associated with the cluster k. However, since target and palette images do not have exactly the same visual content in general, similar to Frigo et al. [85], we found that matching the colour content of the images was preferable to matching the true colour distribution of the images. Therefore we choose equal weights for each cluster and set $\pi^{(k)} = \frac{1}{K}$.

4.1.2 K-means Algorithm

The K-means clustering algorithm can be used to define $\{\mu^{(k)}\}$ for both target and palette images [11, 10]. The computational complexity of the algorithm is then controlled by K, the number of K-means clusters found in the pixel value sets $\{x_t^{(i)}\}_{i=1,\cdots,n_t}$ and $\{x_p^{(i)}\}_{i=1,\cdots,n_p}$. The K-means algorithm is equivalent to using an EM algorithm

which enforces identical isotropic covariance matrices [134].

When choosing K we took a variety of images with different colour distributions and applied the K-means algorithm to them, choosing values of K from 5 to 80. For each value K we computed the sum of squared distances between each pixel in the image and its assigned cluster centre. We then plotted the value K versus the sum of squared distances value, as in Figure 4.1. We found that although some images could be well described by a low value of K (such as the image in Column 1 of Figure 4.1), images with a wide range of colours seemed to be well described by $K = 50$ clusters, with the sum of squared distances value tapering off around $K = 50$. In order to ensure that our framework worked when images had a large variety of colours we choose to set $K_t = K_p = 50$ in our experiments. This value could be decreased by the user depending on the colour distribution of the images being processed, which would decrease the computational complexity of the algorithm.

4.1.3 Mean Shift Algorithm

A possible alternative to K-means would be to estimate the cluster centres $\{\mu^{(k)}\}$ using the Mean Shift algorithm instead [88][1]. The Mean Shift algorithm clusters the pixel data based on both their colour and position values and can also be thought of as an EM algorithm [135]. The number of clusters generated is automatically determined by the Mean Shift algorithm. Figure 4.2 illustrates the selection of means using both K-means and Mean Shift algorithms in the RGB colour space. Note how K-means propose a more evenly spread out set of means.

4.1.4 Correspondences

Pixel pairs between target and palette images, denoted $\{(x_t^{(k)}, x_p^{(k)})\}_{k=1,\cdots,n}$, can be computed when considering target and palette images capturing the same scene. Colour transfer techniques are indeed often used in this context to harmonise colours across a video sequence and/or across multiple view images such as in image stitching. In this case, the means of the Gaussian mixtures are set such that $\{(\mu_t^{(k)}, \mu_p^{(k)}) = (x_t^{(k)}, x_p^{(k)})\}_{k=1,\cdots,n}$ imposing $K_t = K_p = n$ when defining the distributions p_t and p_p. Moreover, the scalar

[1]Code for [88] at `http://coewww.rutgers.edu/riul/research/code.html`

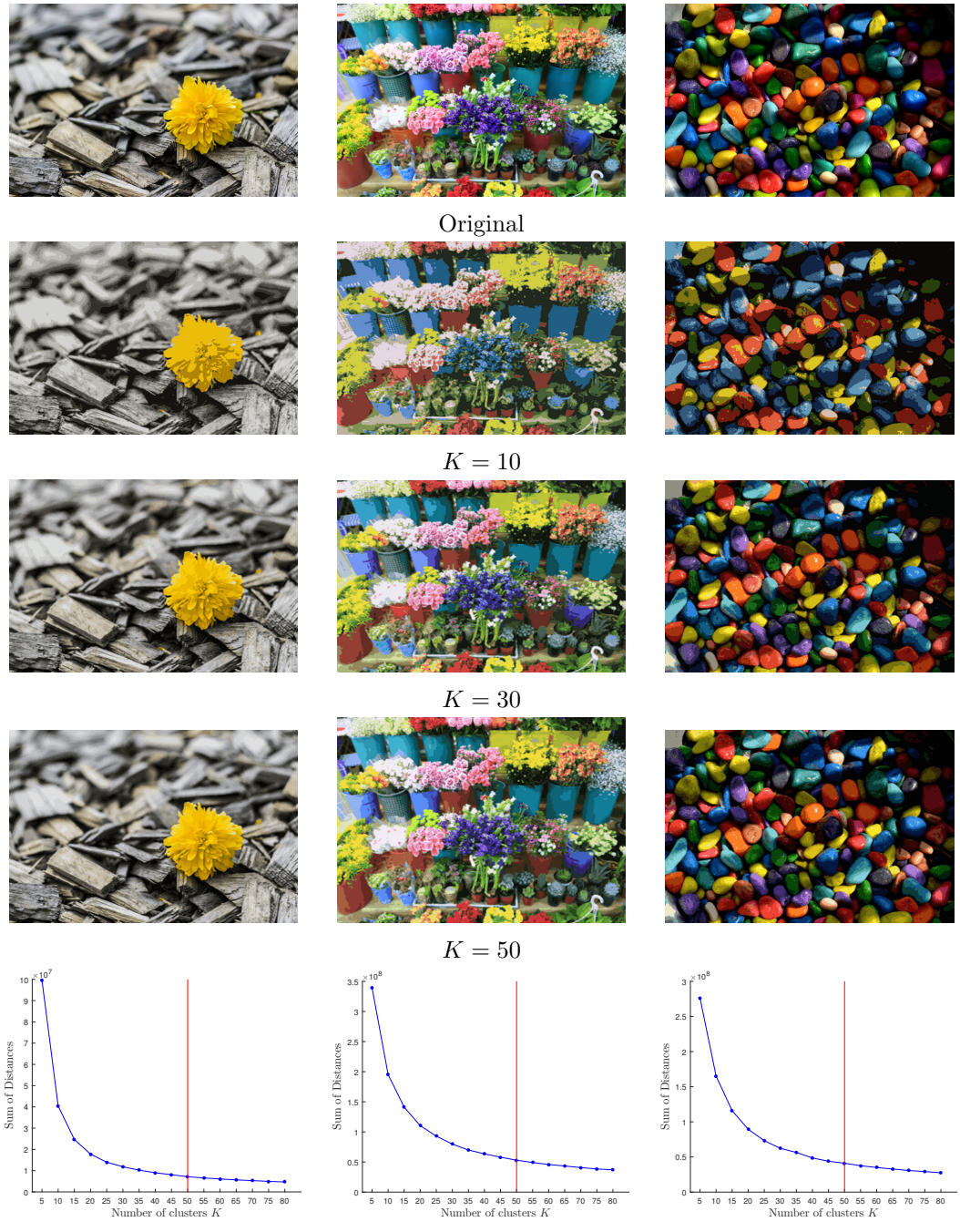


Figure 4.1: This figure shows the effect of choosing different values of K, the number of clusters chosen by the K-means algorithm. Rows 1: Original images; Rows 2-4: Clustered image with $K = 10, 30$ and 50; Row 5: The plots showing K versus the total squared distance between each point in the image and its assigned cluster centre. For images with a varied colour distribution this seems to taper off around $K = 50$ (red line).

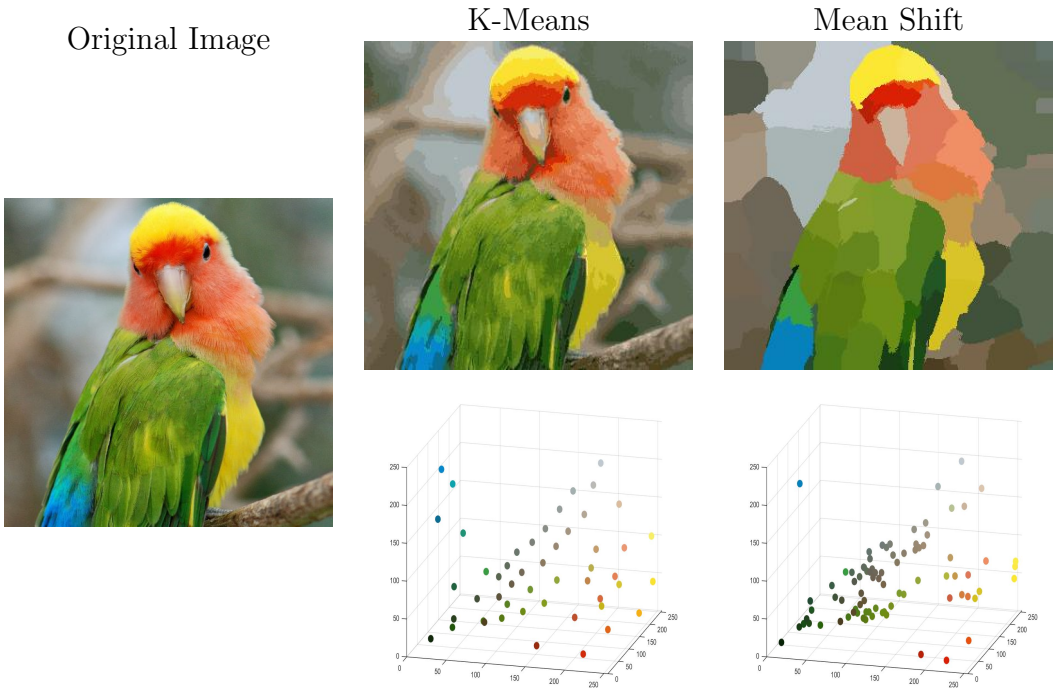


Figure 4.2: This figure compares the colours sampled using the K-Mean (K= 50) and Mean Shift (K= 147) algorithms. When using K-means the value for K is set by the user, while the Mean Shift algorithm determines the value of K automatically. Column 1: Original image; Column 2: K-Mean results; Column 3: Mean Shift results. The top row of column 2 and 3 shows the original image recoloured using only the colours selected using each clustering technique. Row 2 shows the distribution of these sampled colours in RGB space.

product $\langle p_t | p_p \rangle$ in our cost function is then simplified as follows:

$$\langle p_t | p_p \rangle = \sum_{k=1}^{n} \mathcal{N}(0; \phi(\mu_t^{(k)}, \theta) - \mu_p^{(k)}, \Sigma_t^{(k)} + \Sigma_p^{(k)})\ \pi_t^{(k)}\ \pi_p^{(k)} \tag{4.11}$$

The computational complexity of this term is then $n = K_t = K_p$ when using n correspondences, and $K_t \times K_p$ without correspondences. Performance of our approach with correspondences is assessed in Section 4.2 using palette and target images with similar content. Pixel correspondences are not used when considering palette and target images with different content but K-means or Mean Shift clustering are used instead.

4.1.5 Warping function ϕ

RBF name	RBF equation
TPS	$\psi(r) = -(r)$
Gaussian	$\psi(r) = e^{-(\epsilon r)^2}$
Inverse Multiquadric	$\psi(r) = \frac{1}{\sqrt{1+(\epsilon r)^2}}$
Inverse Quadric	$\psi(r) = \frac{1}{1+(\epsilon r)^2}$

Table 4.1: The equations for the different radial basis functions ψ tested in this chapter.

In this chapter, several transformation functions ϕ are tested including an affine transformation, as defined in Equation 3.13, and radial basis functions, as defined in Equation 3.16. When using RBFs, various basis functions ψ are used when defining the transformation ϕ, including TPS, Gaussian, Inverse Multiquadric and Inverse Quadric, and are listed in Table 4.1. The colour spaces considered for testing our framework are the RGB and CIELab spaces. In both cases, when using an RBF transformation, the N control points $\{c_j\}_{j=1,\cdots,N}$ are chosen on a regular grid spanning the 3D colour space such that the 3D grid has $N = 5 \times 5 \times 5 = 125$ control points in the colour space. As a consequence the dimension of the latent space that needs to be explored when estimating θ in this case is:

$$\dim(\theta) = 125 \times 3 + 9 + 3 = 387$$

with $\dim(c_j) = 3 \; \forall j$, $\dim(\mathrm{A}) = 3 \times 3 = 9$ and $\dim(o) = 3$ (cf. Equation 3.16). On the other hand, when estimating an affine transformation, the latent space has dimension 9 (cf. Equation 3.13).

4.1.6 Estimation of θ

To enforce that a smooth solution ϕ is estimated when using a radial basis function transformation, the Euclidean $\mathcal{L}_2$ distance is minimised with a roughness penalty on the transfer function ϕ [48] and the estimation is performed by minimising:

$$\hat{\theta} = \arg\min_{\theta} \left\{ \mathcal{C}(\theta) = \|p_t\|^2 - 2\langle p_t | p_p \rangle + \lambda \int \|D^2 \phi(x, \theta)\|^2 dx \right\} \tag{4.12}$$

with $\|D^2\phi(x,\theta)\|^2 = \sum_{i,j\in\{1,\cdots,d\}} \left(\frac{\partial^2\phi}{\partial x_i \partial x_j}\right)^2$ and $d = 3$.

4.1.7 Algorithm

Our strategies to estimate θ are summarised in Algorithm 1.

Algorithm 1 Our strategies for estimating the parameter θ for the warping function $\phi(x,\theta)$.

Require: $\hat{\theta}$ initialised so that $\phi(x,\hat{\theta}) = x$ (identity function)
Require: $hmin$, $hmax$, λ and choose ψ (with ϵ)
Require: Initialisation of $\{\mu_t^{(i)}\}$, $\{\mu_p^{(i)}\}$ or $\{(\mu_t^{(i)}, \mu_p^{(i)})\}$ (correspondences)
  if Using K-means (no correspondences) **then**
    choose K_t and K_p
    Cost function $\mathcal{C}(\theta)$ (Eq. 4.12) defined with term $\langle p_t | p_p \rangle$ (Eq. 4.9)
  else if Using Meanshift (no correspondences) **then**
    K_t and K_p are estimated by Meanshift
    Cost function $\mathcal{C}(\theta)$ (Eq. 4.12) defined with term $\langle p_t | p_p \rangle$ (Eq. 4.9)
  else if Using n correspondences **then**
    $K_t = K_p = n$
    Cost function $\mathcal{C}(\theta)$ (Eq. 4.12) defined with term $\langle p_t | p_p \rangle$ (Eq. 4.11)
  end if
  Start $h = hmax$ (controls covariance matrix Eq. 4.6)
  repeat
    $\hat{\theta} \leftarrow \arg\min_\theta \mathcal{C}(\theta)$
    $h \leftarrow .5 \times h$ (annealing)
  until Convergence $h < hmin$ **return** $\hat{\theta}$

In order to avoid local minima, we implement a simulated annealing strategy which is controlled by the bandwidth h. We begin our optimisation with a large value of h and slowly decrease it to ensure the algorithm does not get caught in local solutions. When estimating a radial basis function ϕ, two values, λ and ϵ (in the case of the Gaussian (G), Inverse Quadric (InQ) and Inverse Multiquadric (InMQ) basis functions, as defined in Tab. 4.1) need to be chosen in our framework. Table 4.2 gives the values that were

RBF ψ and colour space	Correspondences		No Correspondences	
	λ	ϵ	λ	ϵ
TPS_{rgb}	$3e^{-3}$	✗	$3e^{-6}$	✗
TPS_{lab}	$3e^{-3}$	✗	$3e^{-4}$	✗
G_{rgb}	$3e^{-5}$	$6e^{-3}$	$3e^{-8}$	$6e^{-3}$
G_{lab}	$6e^{-3}$	3	$3e^{-4}$	3
InMQ_{rgb}	$3e^{-5}$	$6e^{-3}$	$3e^{-8}$	$6e^{-3}$
InMQ_{lab}	$6e^{-3}$	10	$3e^{-4}$	3
InQ_{rgb}	$3e^{-6}$	$6e^{-3}$	$3e^{-8}$	$6e^{-3}$
InQ_{lab}	$6e^{-3}$	30	$3e^{-4}$	3

Table 4.2: The values for λ and ϵ used for each of our proposed techniques when generating the results in Section 4.2. The values for λ and ϵ differ depending on whether correspondences are used or not. If correspondences are not available, the K-means or Mean Shift algorithms are used to compute the Gaussian centres.

used in our experiments and found to give the best results overall. For clarity, we extend the notation in Table 4.2 so that TPS_{rgb}^{KM} indicates that the basis function ψ is TPS, the colour space is RGB, and the clustering techniques for finding the means of the GMMs is K-means (similarly, we denote TPS_{rgb}^{MS} for meanshift and TPS_{rgb}^{Corr} when using Correspondences). The subscript *lab* is used to notate methods tested in CIELab colour space.

4.1.8 Optimisation Details

When minimising the cost function $\mathcal{C}(\theta)$ to estimate the transformation ϕ we used the Matlab function *fminunc*, which minimises unconstrained multivariate functions. The minimisation algorithm used by *fminunc* is the Quasi-Newton method [136], which is a gradient ascent algorithm. We computed the analytical derivative of the cost function $\mathcal{C}(\theta)$ and passed it to the gradient ascent algorithm to speed up the optimisation [41].

4.1.9 Parallel Recolouring step

One of the main advantages of this method is the fact that our transformation is controlled by a parameter θ and any pixel x can be recoloured by computing the new colour value $\phi(x, \theta)$. This computation can be done in parallel by distributing the

pixels to be recoloured to the multiple processors that are available. Moreover, with one target image and N palette images of choice, the transformations $\{\phi_1, \cdots, \phi_N\}$ controlled by the parameters $\{\hat{\theta}_1, \cdots, \hat{\theta}_N\}$ can also be estimated in parallel. A new value θ_{new} could also be computed via interpolation and used for recolouring:

$$\theta_{new} = \sum_{j=1}^{N} \gamma_j \, \hat{\theta}_j, \quad \text{with} \sum_{j=1}^{N} \gamma_j = 1, \text{and } \gamma_j \geq 0 \; \forall j \tag{4.13}$$

Interpolating in the θ space to create a new warping function is made easy thanks to the fact that we chose control points on a regular grid (cf. Section 4.1.5) and not as sub samples of pixel values from the palette images. Of particular interest are the creation of smooth temporal transitions between the identity warping function and a colouring warping function for instance. In Section 4.3.1 we give more details on how to create visual effects using interpolation masks. A fast recolouring step is essential for giving the user instant feedback about the new effects being applied to the image or video. Our transformation ϕ can be applied to each pixel independently and it is therefore highly parallelisable. A parallel implementation on the CPU or GPU would ensure that the target image is recoloured almost instantly. For our implementation, we parallelised the recolouring step on the CPU using OpenMP, and the performance of our algorithm is assessed in Section 4.2.4.

4.2 Experimental results

In this section we assess how our proposed colour transfer techniques perform in comparison to other state of the art methods. To quantitatively assess recolouring results two metrics, peak signal to noise ratio (PSNR) and structural similarity index (SSIM), are often used when considering palette and target images of the same content for which correspondences are easily available [57, 91, 5]. The PSNR metric is used to compute the colour similarity between the palette image P and colour transfer result image R, while SSIM is used to measure the structural similarity between the target image T and result image R. Alternatively, user studies have also been used to assess the perceptual visual quality of the recolouring [137].

We evaluate our proposed algorithms and show that they are comparable to current

state of the art colour transfer algorithms in terms of the perceptual quality of the results (Section 4.2.1, 4.2.2 and 4.2.3), and superior in terms of computational speed (Section 4.2.4). Moreover our parametric colour transfer formulation provides artists with an easy and flexible way to create new visual effects (Section 4.3.1).

Table 4.3 summarises the methods (including ours) used for comparison in this chapter. We chose to compare against a number of recent optimal transport based methods whose code was made available online by the authors [6, 82, 77], as well as a recent colour transfer technique which takes into account pixel correspondences and has recently been shown to provide better results than other leading correspondence based methods [5].

Method Name	Ref.	Corr.	No Corr.	Code availability for testing	Image	Video	Test $P \simeq T$ (Sec. 4.2.1)	Test $P \neq T$ (Sec. 4.2.2)
Ours		yes	yes	`https://www.scss.tcd.ie/~mgrogan/colourtransfer.html`. Partial code, needs updating.	yes	yes	yes	yes
Bonneel	[77]	no	yes	`https://github.com/gpeyre/2014-JMIV-SlicedTransport`	yes	no	yes	yes
PMLS	[5]	yes	no	Results for this method in this chapter have been processed by the authors of [5]	yes	yes	yes	no
Ferradans	[82]	no	yes	`https://github.com/gpeyre/2013-SIIMS-regularized-ot`	yes	no	no	yes
Pitié	[6]	no	yes	`https://github.com/frcs/colour-transfer`	yes	no	yes	yes

Table 4.3: Road-map for experiments. Our framework is able to take advantage of correspondences (Corr) between Target (T) and Palette (P) images when available, and also works without correspondences (No Corr). While correspondences are easily available when palette and target images capture the same visual content ($P \simeq T$), they are not available when using images of different content ($P \neq T$).

4.2.1 Images with similar content $P \simeq T$

One important application of colour transfer is in harmonising the colour palette of several images or videos capturing the same scene. To evaluate our algorithm when applied to images with similar content, we use the 15 images in the dataset provided by Hwang et al. [5][2] which includes images with many different types of colour changes

[2] `https://sites.google.com/site/unimono/pmls`

including different illuminations, different camera settings and different colour touch up styles. This dataset provides palette images which have been aligned to match the target image (c.f. Figure 4.4). To define correspondences, pixels at the same location in the target and aligned palette images are randomly selected together to form a pair [5]. For our proposed techniques, the number of corresponding pixel pairs chosen is $n = 50000$.

Choice of transformation ϕ

Table 4.4 provides quantitative results comparing colour transfer results when ϕ is an affine transformation (Aff) or one of the tested radial basis functions. In this case our algorithms use correspondences and the quantitative results are computed using the PSNR and SSIM metrics. We provide image results in Appendix A. In general the

	PSNR		SSIM	
	μ	SE	μ	SE
Aff_{rgb}^{Corr}	24.97	0.8	0.914	0.02
Aff_{lab}^{Corr}	25.35	0.8	0.903	0.02
TPS_{rgb}^{Corr}	30.30	1.5	0.944	0.02
TPS_{lab}^{Corr}	30.56	1.4	0.942	0.02
G_{rgb}^{Corr}	30.03	1.5	0.944	0.02
G_{lab}^{Corr}	30.56	1.4	0.942	0.02
InMQ_{rgb}^{Corr}	30.37	1.5	0.944	0.02
InMQ_{lab}^{Corr}	30.49	1.4	0.942	0.02
InQ_{rgb}^{Corr}	30.37	1.5	0.944	0.02
InQ_{lab}^{Corr}	30.22	1.4	0.940	0.02

Table 4.4: Assessment of our algorithms using correspondences with several basis functions ψ in two colour spaces. The mean PSNR and SSIM values μ for each method are computed on the 15 images in the dataset. Highest PSNR and SSIM values indicate the best results (best in red, second best in green, third best in blue). The standard error SE for each method is also given.

methods applied in the CIELab colour space provide slightly better results in terms of PSNR, and slightly lower results in terms of SSIM. While the affine transformation gives significantly lower results in terms of PSNR and SSIM in both RGB and CIELab colour space, a t-test comparing the mean values of each radial basis function method confirms

that the difference between them is statistically insignificant (with 99% confidence). Visually, we found that the results created with an affine transformation were the least successful, while the results from the radial basis functions matched the palette image well and were all very similar (See Appendix A). TPS has the advantage of being faster to compute (Section 4.2.4) and since it achieves perceptually similar performance to other basis functions, it is mainly used for comparison in the rest of this chapter.

With or without correspondences

Table 4.5 compares TPS_{rgb}^{KM} and TPS_{rgb}^{MS}, computed without correspondences, to TPS_{rgb}^{Corr}, computed with correspondences, when they are applied to images with the same content. The results stress that using correspondences allows TPS_{rgb}^{Corr} to outperform TPS_{rgb}^{KM} and TPS_{rgb}^{MS}. Note however, that in terms of SSIM and PSNR, there is no statistical difference between the results of TPS_{rgb}^{KM} using K-means and TPS_{rgb}^{MS} using Mean Shift when the palette and target images capture the same scene. Section 4.2.2 will show that TPS_{rgb}^{KM} gives more perceptually pleasing results than TPS_{rgb}^{MS} for images of different content.

	TPS_{rgb}^{Corr}	TPS_{rgb}^{KM}	TPS_{rgb}^{MS}
PSNR	30.30 (1.5)	24.20 (0.8)	24.47 (0.8)
SSIM	0.944 (0.02)	0.908 (0.02)	0.896 (0.02)

Table 4.5: The mean PSNR and SSIM values for the set of images with similar content, with the standard error shown in brackets. Using correspondences leads to better results.

Robustness of $\mathcal{L}_2$ to outlier correspondences

To evaluate the robustness of the $\mathcal{L}_2$ metric, we applied TPS_{rgb}^{Corr} to images that had many false correspondences. Taking the registered palette and target images, we applied a horizontal shift of s pixels to the target image. Then taking pixels at the same location in the palette and new target image as correspondences, we computed the colour transfer result. Figure 4.3 shows that even when a large number of false correspondences are present, the colours in the result image are very similar to those in the palette image. Areas which have changed colour are long thin structures which no longer have many correct colour correspondences (the blue bars on the tower become

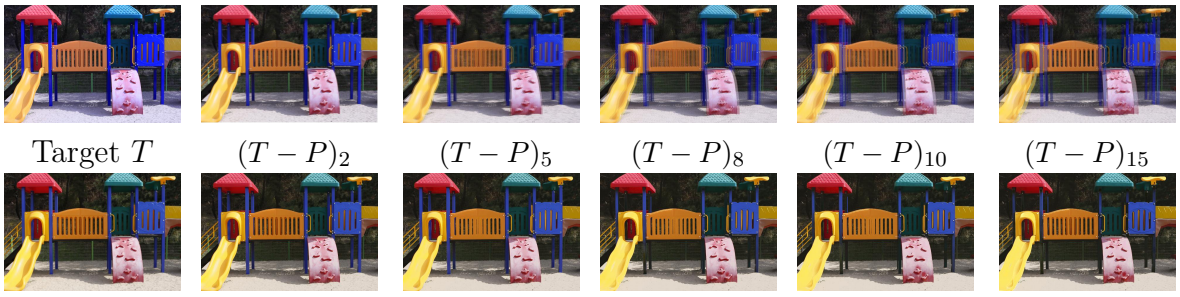


Figure 4.3: Robustness of $\mathcal{L}_2$ with outlier correspondences. The first column gives the target (top) and palette image (bottom). The remaining columns show the colour transfer results of TPS_{rgb}^{Corr} when the target image is shifted horizontally by s pixels, creating incorrect colour correspondences. The top row shows the shifted target superimposed on the palette image ($(T-P)_s$) and the bottom row shows our colour transfer result with corresponding SSIM and PSNR results.

green in Figure 4.3). The structure of the target image has also been well maintained overall.

Comparison to current leading techniques

Table 4.6 provides a quantitative evaluation of our proposed method (TPS in RGB and CIELab spaces) in comparison to leading state of the art colour transfer methods [77, 6, 5][3], with the notations used explained in Table 4.3. In terms of both PSNR and SSIM, PMLS performs slightly better in most cases, closely followed by our TPS_{rgb}^{Corr} and TPS_{lab}^{Corr} methods, but t-tests confirm that there is no significant quantitative difference between PMLS and each of our proposed techniques TPS_{rgb}^{Corr} and TPS_{lab}^{Corr} (with a 99% confidence level).

PMLS however, introduces some visual artifacts when the content in the target and palette images is not registered exactly. These artifacts can be seen around the car in Row 3, Column 5 of Figure 4.4. PMLS is not robust to registration errors, while our algorithm is, thanks to the robust $\mathcal{L}_2$ distance. Our approach allows us to maintain the structure of the original image and to create a smooth colour transfer result (cf.

[3]Results using PMLS were provided by the authors of [5].

	PSNR					SSIM				
	Pitié	Bonneel	PMLS	TPS_{rgb}^{Corr}	TPS_{lab}^{Corr}	Pitié	Bonneel	PMLS	TPS_{rgb}^{Corr}	TPS_{lab}^{Corr}
building	20.50	12.32	22.63	20.50	22.51	0.807	0.675	0.865	0.862	0.864
flower1	24.02	18.42	26.98	26.86	26.85	0.908	0.822	0.967	0.966	0.961
flower2	25.32	21.26	25.76	25.77	25.92	0.900	0.836	0.928	0.927	0.924
gangnam1	24.61	23.86	35.74	35.37	35.70	0.899	0.908	0.992	0.990	0.985
gangnam2	26.59	26.82	36.58	35.55	35.51	0.918	0.928	0.993	0.986	0.984
gangnam3	22.23	19.69	35.02	33.29	33.10	0.877	0.816	0.991	0.980	0.971
illum	19.89	14.34	20.17	19.08	19.84	0.632	0.527	0.649	0.648	0.650
mart	22.71	22.15	24.74	24.45	24.92	0.906	0.901	0.957	0.956	0.955
playground	27.38	25.96	27.84	27.65	27.91	0.916	0.900	0.940	0.939	0.938
sculpture	29.85	22.57	32.06	32.07	32.10	0.942	0.873	0.971	0.972	0.971
tonal1	28.55	17.87	37.22	37.33	37.19	0.940	0.852	0.988	0.987	0.987
tonal2	27.88	23.00	31.51	31.36	31.33	0.968	0.948	0.987	0.986	0.985
tonal3	29.37	16.90	36.25	36.65	36.23	0.961	0.865	0.992	0.992	0.991
tonal4	28.57	14.80	34.52	34.34	34.44	0.943	0.812	0.983	0.983	0.983
tonal5	30.20	21.08	35.26	34.30	34.96	0.965	0.911	0.986	0.985	0.984
μ	25.85	20.07	30.82	30.30	30.56	0.899	0.838	0.946	0.944	0.942
SE	0.9	1.1	1.5	1.5	1.4	0.02	0.03	0.02	0.02	0.02

Table 4.6: Comparison of our algorithms TPS_{rgb}^{Corr} and TPS_{lab}^{Corr} against state of the art techniques. SSIM and PSNR results for colour transfer techniques on images with similar content: highest values (in red) for best performance, in green second best performance and in blue third best performance. The overall mean score μ and its standard error SE for each method are also given.

Row 3, Column 6 in Fig 4.4). So while PMLS and our algorithms provide equivalent quantitative performances as measured by PSNR and SSIM, our techniques in fact provide better qualitative visual results.

4.2.2 Images with different content $P \neq T$

First we compare two methods for estimating the Gaussian means $\{\mu^{(k)}\}$ - K-means and Mean Shift. In Figure 4.6 we present the target and palette images recoloured using the K cluster centres computed using these techniques. The cluster centres generated using K-means are evenly spaced throughout the colour distribution of the images and the recoloured images look very similar to the original target and palette. On the other hand, the Mean Shift algorithm takes pixel colour and position into account and the cluster centres therefore depend on the structure of the image. Visually, we found that setting the Gaussian mixture means to be the K-means cluster centres gave better results than the Mean Shift clusters, as seen in Figure 4.6. Therefore, we present results

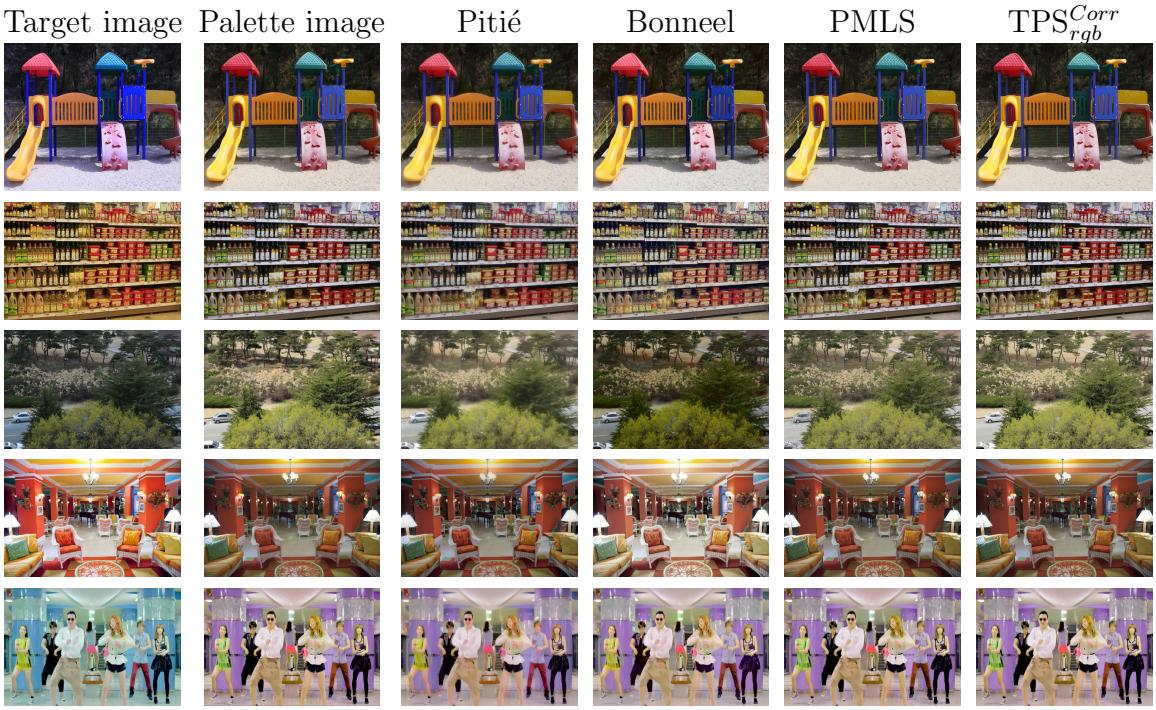


Figure 4.4: Results on images with similar content on the 'playground', 'mart', 'illum', 'tonal4' and 'gangnam2' images. On close inspection grainy artifacts can be seen appearing in some PMLS results. For example, around the car in Row 3 Column 5 or in the top right corner of Row 2 Column 5. In comparison, the results generated by TPS$^{Corr}_{rgb}$ remain smooth (Column 6). For zoom see Fig. 4.5.

obtained using the K-means clustering technique in the rest of this section.

We compare our algorithm with other colour transfer techniques [77, 82, 6] applied to images of different content and without correspondences. In the case of Ferradans et al's results, all images were generated using the parameters $\lambda_X = \lambda_Y = 10^{-3}$ and $\kappa = (0.1, 1, 0.1, 1)$ [82]. Figure 4.7 shows that the Bonneel and Ferradans methods create blocky artifacts in the result image gradient in some cases (Row 2, 6, 8). On the other hand, the added constraint in the Pitié algorithm, which enforces a smooth image gradient, ensures that these errors do not appear in their results, creating images that are more visually pleasing. Similarly, the results of our algorithm produce results that match the colours in the palette image well, while still maintaining a smooth image gradient.

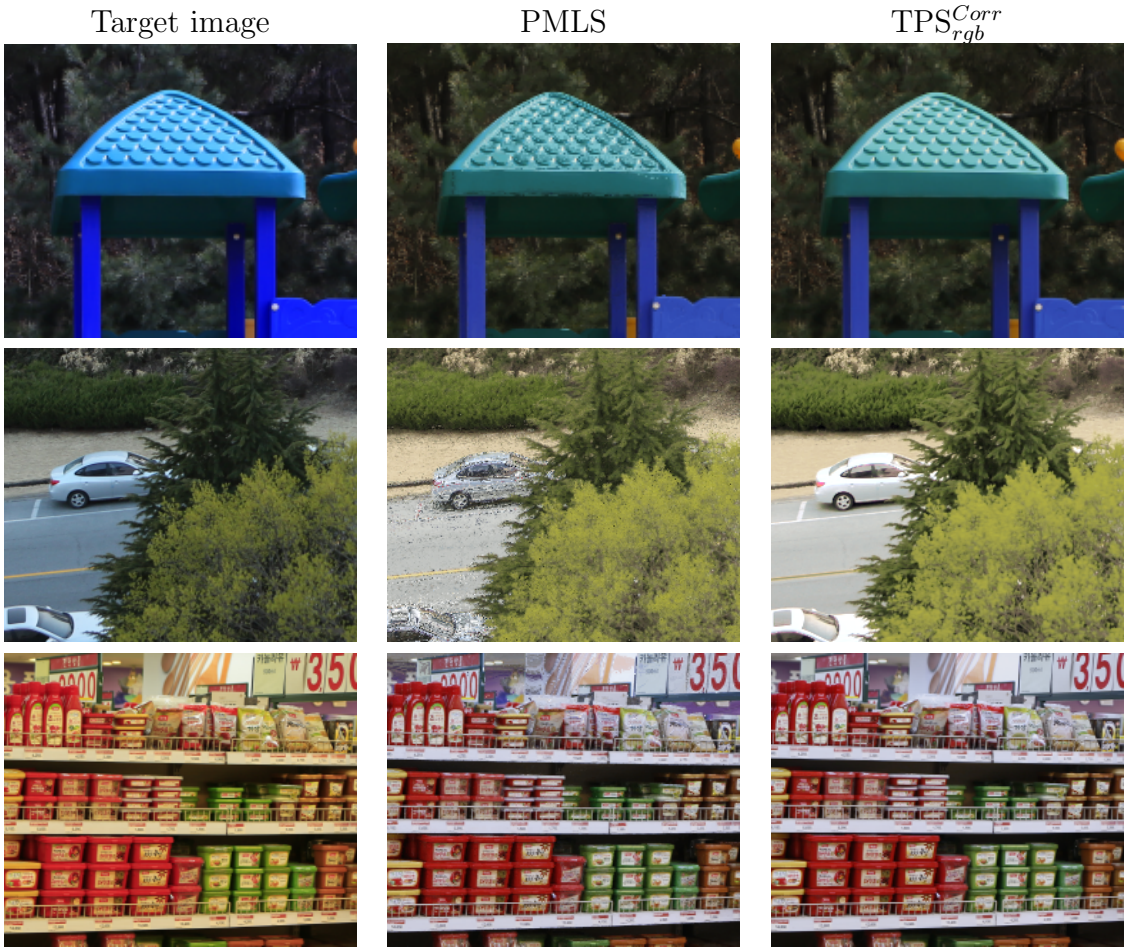


Figure 4.5: A close up look at some of the errors generated using the PMLS [5] algorithm in comparison to our smooth result with TPS$_{rgb}^{Corr}$.

In our framework, we found that although an affine transformation can give good results in some cases, in others it does not perform as well as the RBF functions (see Appendix A). The non-linear nature of the RBF functions allows them to achieve more complex colour transfer results, while the affine transformation is limited in the complexity of colour transfer results it can achieve. We found that TPS is the most consistent RBF function in terms of quality of results and the parameters selected for TPS, given in Table 4.2, generate good results overall. However, when more non-linear basis functions ψ are used to estimate θ, they have a tendency to fall into local minima when no correspondences are available. Therefore choosing the best parameters, as in

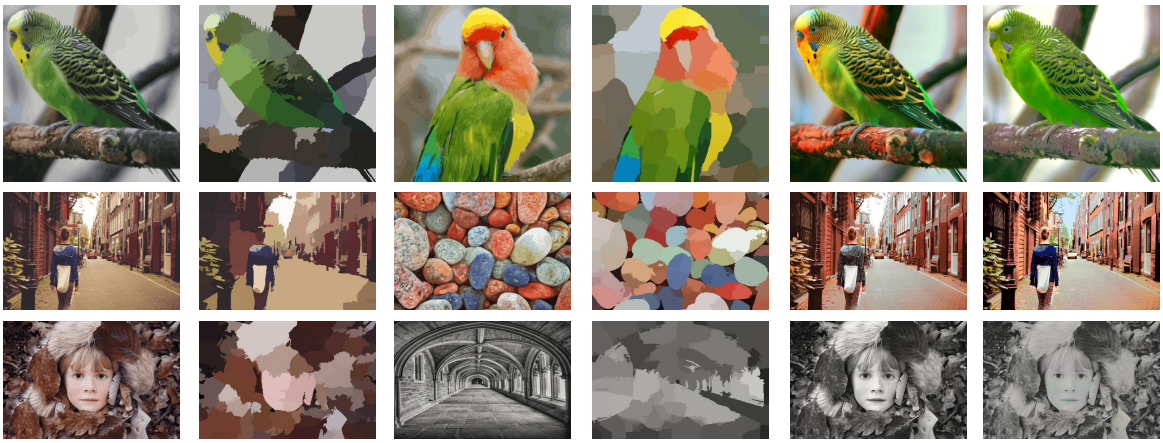


Figure 4.6: Comparison between K-means clustering and Mean Shift clustering when setting the Gaussian centres $\{\mu^{(t)}\}$ of p_t and p_p. Columns 1 to 4 show the target and palette images recoloured with the K cluster centres found using either K-means or Mean Shift. Columns 5 and 6 shows the results obtained by TPS$_{rgb}$ using the clusters determined by each algorithm.

Table 4.2, that create good results for every image is quite difficult. However, when the estimate $\hat{\theta}$ found is the global minimum, the results are very similar to TPS for both the CIELab and RGB colour spaces. Image results comparing the affine transformation and alternate basis functions ψ can be seen in Appendix A.

4.2.3 Qualitative assessment

User Study Methodology

Colour transfer methods have also been evaluated using a subjective user study with 20 participants evaluating 53 sets of images. They were asked to choose the colour transfer result that they thought was the most successful. Out of these 53 sets of images, 38 of them had a palette and target image with different content (no correspondences available), and 15 of them contained a palette and target image with the same content (correspondences available and used). These 15 images were taken from the dataset provided by Hwang et al. as detailed in Section 4.2.1.

Each participant was presented with six images at a time - a target image, a palette

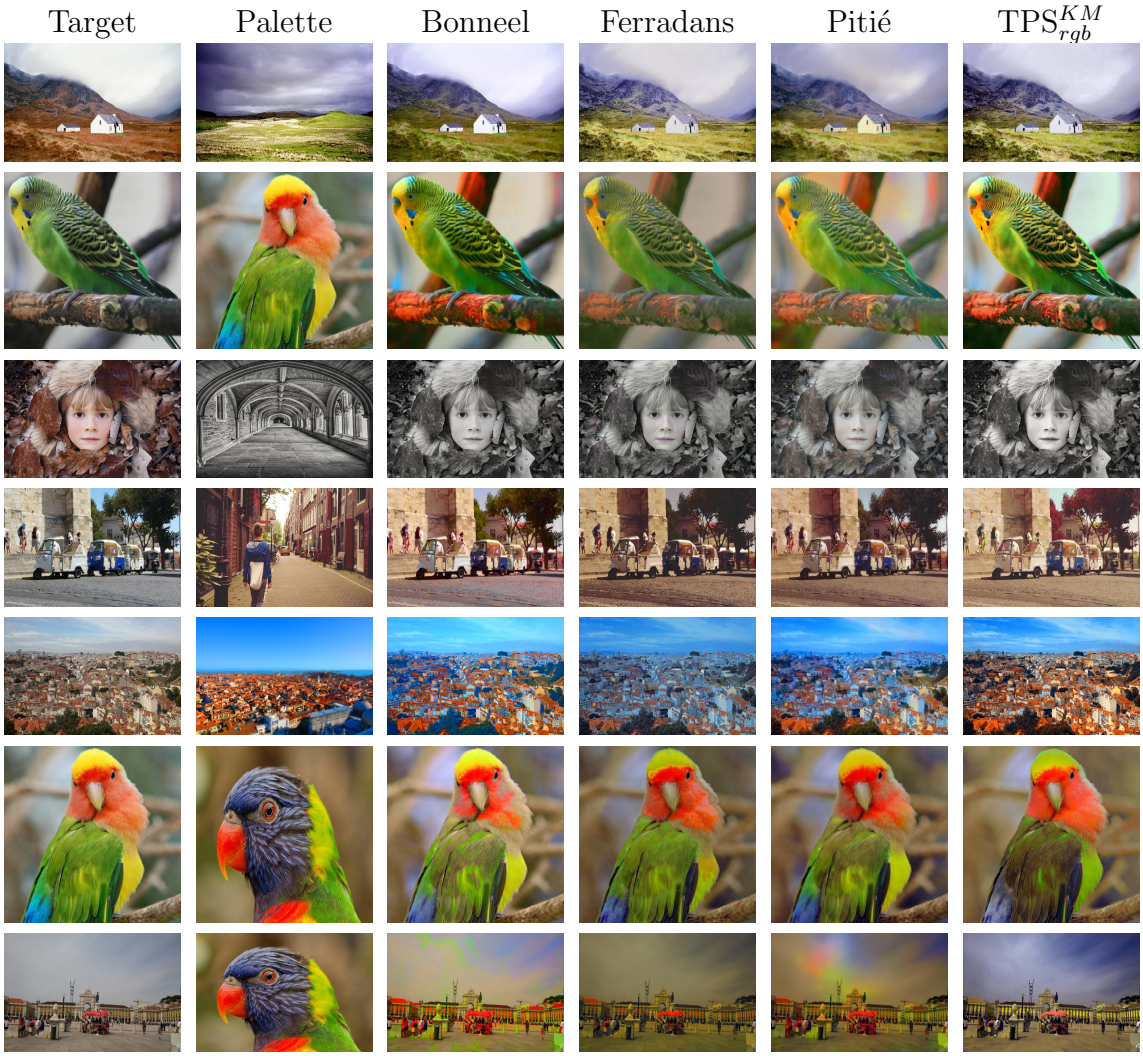


Figure 4.7: Some results on images with different content which were presented to participants in the user study.

image, and four result images generated using different colour transfer techniques. They then had 20 seconds to view the images (presented simultaneously side by side) and decide which result image was the best. Each user evaluated the results individually. The display properties and indoor conditions were kept constant for every user. The order in which the image sets appeared was randomised for each user and a short trial run took place before the evaluation to ensure the users adapted to the task at hand.

Data analysis

The four methods compared for target and palette images with similar content were TPS_{rgb}, PMLS, Pitié and Bonneel. The total number of times each method was chosen can be seen in Table 4.7. We used a z-test to determine whether there was a significant difference between these proportions. We found that TPS_{rgb} performs better than both Pitié and Bonneel. However, the results indicating TPS_{rgb} performed better than PMLS is not statistically significant. Hence these two methods, PMLS and ours, can be thought of as performing equally well and both being superior to Pitié and Bonneel methods.

Similarly, for images with different content we compared TPS_{rgb}, Ferradans, Pitié and Bonneel. The total number of times each method was chosen can be seen in Table 4.7. Again using a z-test we determined that TPS_{rgb} performs better than both Ferradans and Bonneel, while there is no statistically significant difference between TPS_{rgb} and Pitié.

Similar Content $P \simeq T$				
Bonneel	Pitié	PMLS	TPS_{rgb}^{Corr}	Total
38	63	98	101	300
Different Content $P \neq T$				
Bonneel	Pitié	Ferradans	TPS_{rgb}^{KM}	Total
163	211	152	234	760

Table 4.7: Number of votes given to each method by participants in our perceptual study. This indicates how many times each method was chosen as the best method (best in red, second best in green, third best in blue).

4.2.4 Computation Time

It is important to provide timely feedback for artists and amateurs alike when recolouring image and video materials. The recolouring step is highly parallelisable, allowing the recolouring of video content at a high speed. In terms of computation, our algorithm is split into three parts: the clustering step, the estimation step of θ and the recolouring step $x \to \phi(x, \theta)$. As K-means can be quite a time consuming clustering technique, we investigated some fast quantisation methods, including those provided

by Matlab and the GNU Image Manipulation Program (GIMP). We found that using Matlab's minimum variance quantisation method (MVQ) provided almost identical results to K-means as well as being much faster and could be used as an alternative to K-means to speed up the clustering step (Table 4.8). A comparison between these techniques and K-means can be seen in Figure 4.8.

The estimation step takes approximately 6 seconds using non optimised code. Once θ is estimated however, it can then be used to recolour an image of any size, or be applied to recolour a video clip. It can also be stored for later use to create interesting effects such as those described in Sections 4.3.1 and 4.3.2 .

The recolouring step on the other hand is highly parallelisable and can be applied independently to each pixel. The time taken to recolour a HD image (1080×1920) for each type of radial basis function is given in Table 4.8. In our implementation we used OpenMP within a Matlab mex file to parallelise this step on 8 CPU threads. All times are computed on a 2.93GHz Intel CPU with 3GB of RAM with 4 cores and 8 logical processors. In comparison, the GPU implementation of PMLS takes 4.5 seconds to recolour a 1 million pixel image using an nVIDIA Quadro 4000 as reported in [5], which is 9 times slower than our implementation with TPS. Similarly, Bonneel et al. report a time of approximately 3 minutes to recolour 108 frames of a HD segmented video on an 8 core machine with their algorithm [138]. In comparison, our algorithm would take less than 2 minutes using TPS in the same situation.

It can also be applied to each pixel in parallel and a GPU implementation would allow videos to be recoloured extremely quickly.

4.3 Usability for recolouring

The estimated parametric transfer function computed by our algorithm can be stored and combined easily with other transfer functions computed using various colour palettes for the creation of visual effects in images (Section 4.3.1) and video (Section 4.3.2). Existing postprocessing can also be used to further improve the quality of the results (Section 4.3.3).

Figure 4.8: Comparison between the results obtained using the K-means clustering technique and two faster quantisation techniques. TPS$^{GIMP}_{rgb}$ are the results obtained when the target and palette images have been clustered using GIMPs quantisation method (the median cut algorithm). TPS$^{MVQ}_{rgb}$ are the results obtained when Matlab's minimum variance quantisation algorithm is used to cluster the images. TPS$^{MVQ}_{rgb}$ gives results that are very similar to TPS$^{KM}_{rgb}$.

4.3.1 Mixing colour palettes

For images, as described in Section 4.1.9, a new transformation $\phi(x, \theta_{new})$ can be generated by interpolating between N estimated parameters $\{\hat{\theta}_1, \hat{\theta}_2, ..\hat{\theta}_N\}$ as in Equation 4.13. A simpler interpolation between the identity transformation $\theta^{(Id)}$ and an estimated transformation $\hat{\theta}$ with a selected colour palette can also be created:

$$x \rightarrow \phi\left(x, \gamma\ \hat{\theta}^{(Id)} + (1-\gamma)\ \hat{\theta}\right) \tag{4.14}$$

Some sample results generated by interpolating between colour palettes can be seen in Figure 4.9. This type of recolouring gives the user the flexibility to transition smoothly from one colour mood to another.

While the same transformation can be applied to all pixels in an image, the transformation can also vary depending on the spatial location $p = (p_1, p_2)$ of the colour

Clustering

Method	Time
K-means	10.4s
Mean Shift	3.2s
MVQ	0.005s

Estimating $\hat{\theta}$

K	Time
K = 50	6s

Recolouring

RBF name	Time
TPS	1.04s
Gaussian	3.31s
Inverse Multiquadric	3.16s
Inverse Quadric	2.74s

Table 4.8: Computation times in seconds for each step of our algorithm for a HD (1080 $\times$ 1920) image with over 2 million pixels. (For the clustering step, the images were first downsampled to 300 $\times$ 350 pixels to reduce computation time).

pixel x as follows:

$$x(p) \rightarrow \phi\left(x, \gamma(p)\ \hat{\theta}^{(1)} + (1-\gamma(p))\ \hat{\theta}^{(2)}\right) \tag{4.15}$$

where $\gamma(p)$ is a greyscale mask with values between 0 and 1. Figure 4.10 presents some sample effects created using this type of interpolation. In this figure, pixels in the target image whose corresponding pixel in the grey scale mask is white have been recoloured using palette 1, and those whose corresponding pixel is black have been recoloured using palette 2. The remaining pixels have been recoloured using an interpolation between the two transformations. For each pixel this interpolation is determined by the value of its corresponding pixel in the grey scale mask. These effects can be extended to mixing more than 2 transformations (or palettes).

4.3.2 Application to Video Content

We can also easily extend our colour transfer method to videos [10, 85]. Taking a frame from both the target and palette video clips, we can apply the proposed colour transfer technique to estimate a smooth mapping function ϕ that can then be easily applied

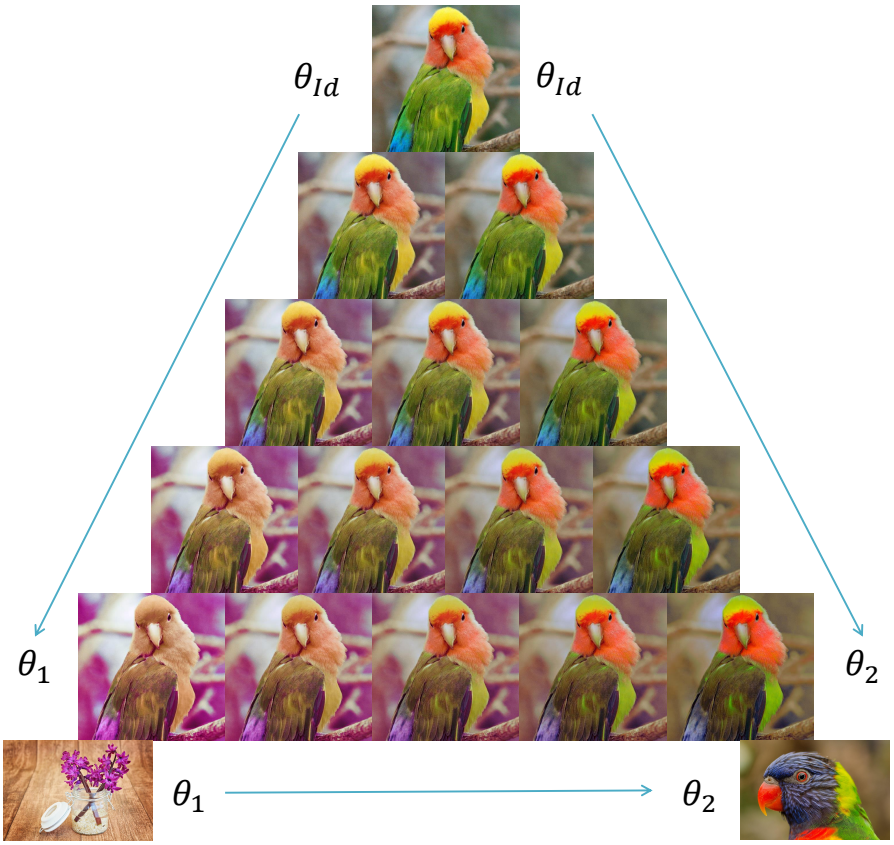
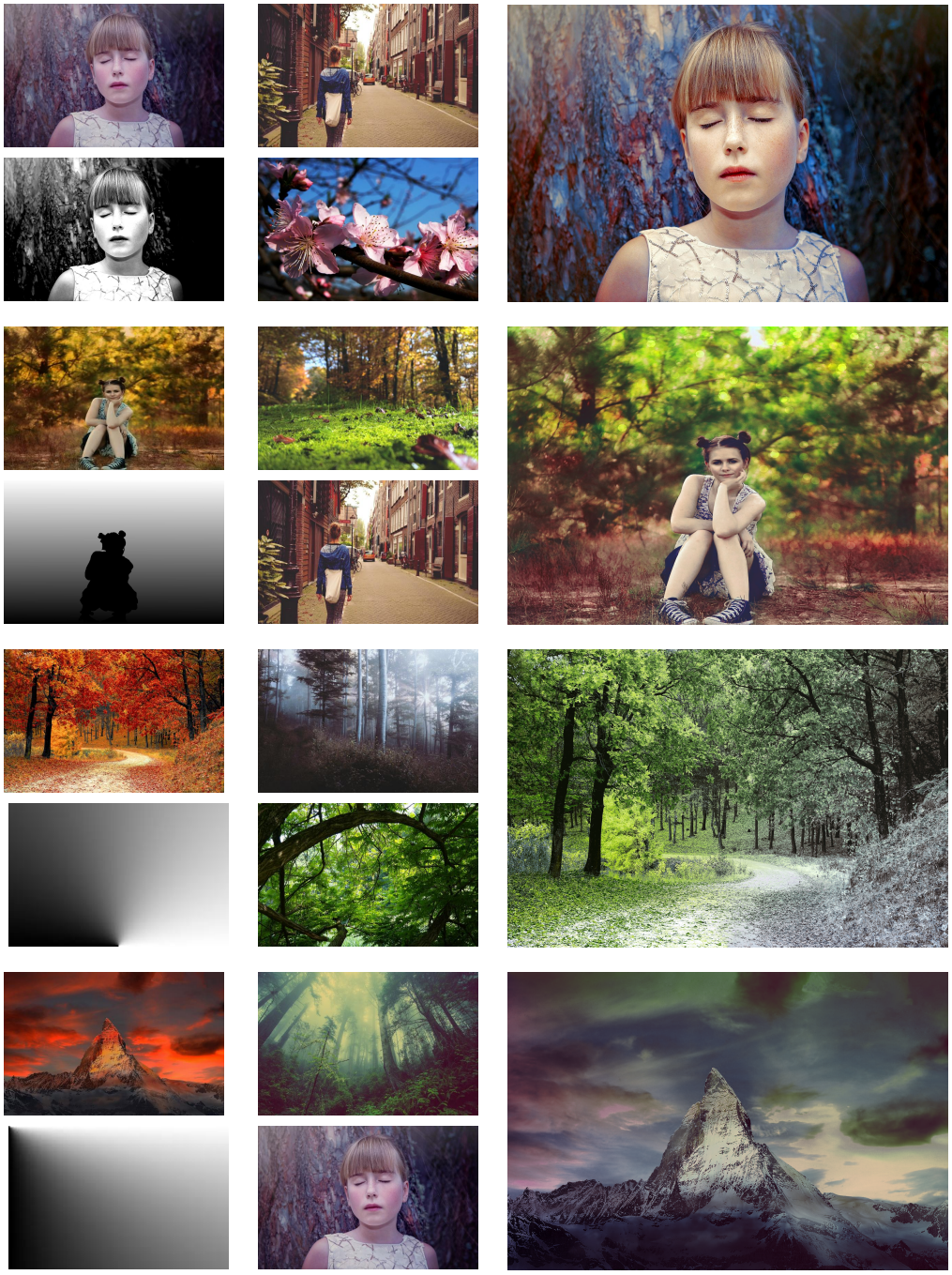


Figure 4.9: Grading effects achievable by recolouring the target image at the top of the table by the transformation $\phi(x, \theta)$, where θ is an interpolation of three parameters θ_{Id}, θ_1 and θ_2: θ_{Id} - the identity; θ_1 - the parameters estimated when transforming the target to the first palette image (bottom left); θ_2 - the parameters estimated when transforming the target to the second palette image (bottom right).

to all frames in the target video clip. As the same transformation is applied to each frame, no temporal smoothing is required and no temporal artifacts are introduced by our method. Figure 4.11 shows the temporal differences $\Delta(t)$ between consecutive frames $I(t)$ and $I(t+1)$ in the original and transformed video clip:

$$\Delta(t) = \sum_p |I^{(p)}(t+1) - I^{(p)}(t)| \tag{4.16}$$



Target & Mask Palettes 1 & 2 Results TPS_{rgb}^{KM}

Figure 4.10: Results images generated using the target and mask (column 1) and two palette images (column 2). Pixels in the target image with a corresponding white pixel in the grey scale mask are recoloured using palette 1 (top), those that are black are recoloured using palette 2(bottom), and the rest are recoloured using an interpolation between the two transformations.

with $I^{(p)}(t)$ the pixel value at pixel location p of the frame $I(t)$. The strong correlation between the curves indicates that the difference between frames is very similar for both video clips. We computed an average correlation of 0.99476 between curves computed for a series of corresponding video clips. Temporal artifacts and discontinuities would cause these curves to differ a lot and these results indicate that our method produces a temporally smooth transformed video.

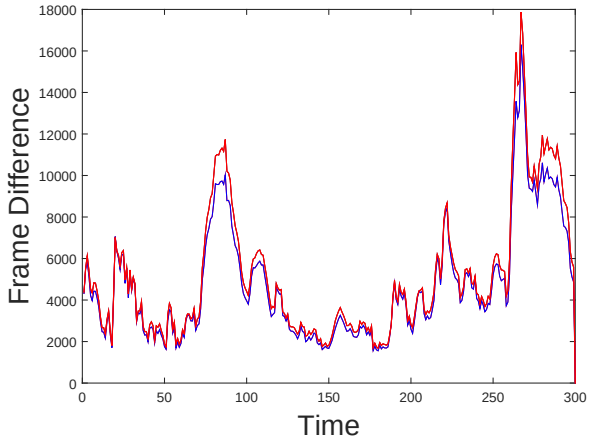


Figure 4.11: Temporal differences $\Delta(t)$ over time. Red: transformed; Blue: original.

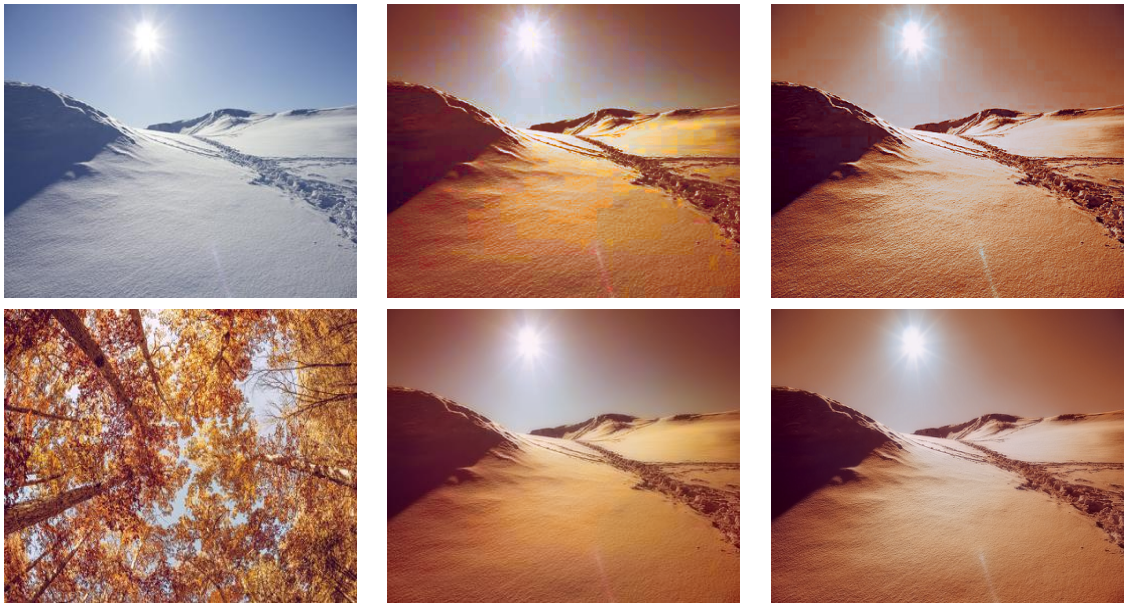
Similar interpolation effects as those applied to images can also be applied to video clips, with the interpolation weight $\gamma(t) \in [0; 1]$ varying over time. To create dissolve effects between colour palettes in videos, a colour pixel x in the sequence at time t can be recoloured as follows [10]:

$$x(t) \rightarrow \phi\left(x, \gamma(t)\ \hat{\theta}^{(1)} + (1 - \gamma(t))\ \hat{\theta}^{(2)}\right) \tag{4.17}$$

As in images, we can extend this idea further by using interpolation weights that vary spatially as well as in the temporal domain, with a colour pixel x at pixel spatial location $p = (p_1, p_2)$ recoloured as follows:

$$x(p, t) \rightarrow \phi\left(x, \gamma(p, t)\ \hat{\theta}^{(1)} + (1 - \gamma(p, t))\ \hat{\theta}^{(2)}\right) \tag{4.18}$$

where $\gamma(p, t)$ is a dynamic greyscale mask with values between 0 and 1.



Target & Palette | Pitié et al.[7, 6] | TPS_{rgb}^{KM}

Figure 4.12: Example of quantisation errors enhanced by our transfer method (e.g. in the sky) as the gradient becomes stretched to match the colours in the palette image (row 1, col 3). The gradient smoothing step proposed by Pitié et al. [6] as an improvement from their ealier method without post processing [7] could be used in our pipeline to remove artifacts although in general this can reduce the sharpness of the image which is undesirable. Column 2: Pitié et al. [7] (before smoothing) and [6] (after smoothing); Column 3: TPS_{rgb}^{KM} (before smoothing) and TPS_{rgb}^{KM} (after smoothing).

4.3.3 Post-processing

While our approach produces excellent results in general, rare saturation artifacts can occur when many colours in the palette image lie close to the boundary of the colour space. As we do not constrain the transformed colours to lie within a specific range, some colours can potentially get transformed outside of the colour space. We currently clamp the colour values so that they are within the desired range. Recently, Oliveira et al. [91] proposed using truncated Gaussians which could be implemented in future to overcome this problem. Additionally, a post-processing step similar to that proposed by Pitié et al.[6] could be added to our pipeline when needed, to constrain the smoothness of the gradient field of the recoloured image so that it remains similar to the gradient field of the target image, as shown in Figure 4.12.

4.3.4 User Interaction

Our approach can also be easily enhanced to allow user interaction when recolouring images and videos. As an alternative to using registration techniques to find pixel correspondences between images, users could manually choose corresponding pixels from the target and palette images. While these pixel correspondences alone could be used to compute the transformation ϕ, there may not be enough to create a good recolouring result. Therefore, these pixel correspondences could be used in addition to cluster centres computed using a clustering technique such as K-means, so that the transformation ϕ is computed using a combination of colours that have no correspondences (K-means clusters) and those that do (user defined correspondences).

Other frameworks have also proposed allowing the user to choose their own colour palette, rather than relying on a palette image, as it may not necessarily provide the user with their desired colours [139]. In this case, rather than fitting a GMM to a select number of colours in the palette image, the GMM would be fitted to each of the colours in the user defined palette. This kind of interaction would give the user the ability to individually select the colours that they would like to see in the result image.

4.4 Conclusion

In this chapter we have presented a new framework for colour transfer which registers two GMMs capturing the colour of the target and palette images using the $\mathcal{L}_2$ distance. We have shown that our method can be applied to images with both similar and different content, and can be enhanced by correspondences when they are available. Our algorithms compete very well with current state of the art approaches since no statistical differences can be measured using metrics such as PSNR and SSIM, and visual inspection of our results shows that our algorithms are more immune to occasional artifacts created in the gradient field of the recoloured image. Our parametric formulation of the transfer function allows for fast recolouring of images and videos. The transfer functions can also be stored and easily combined and interpolated for creating visual effects. Future work could investigate techniques to capture users' preferences by learning from exemplar transfer functions [140]. In the next chapter we will apply a similar registration technique to the shape registration problem, and

investigate whether adding directional information to the shape model can improve the registration results.

Chapter 5

$\mathcal{L}_2$ Registration with Directional Data

In this chapter we further extend the registration ideas presented in Chapter 4 and apply them to the shape registration problem. The proposed method is based on the idea of minimising the distances between two distributions which capture the directional data of the shapes being registered. We present two directional distributions - the von Mises-Fisher distribution and the Watson distribution. We propose four new cost functions which model the directional data using a von Mises-Fisher kernel, two of which combine both the positional and directional information of a shape to enhance the registration. To validate the proposed techniques we use them to register shapes differing by both a rigid and non-rigid transformation and compare them to other state the of the art registration techniques. The 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.

5.1 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 processed regularly and includes surface normals and tangent vectors, orientations of image gradients, the direction of sound sources and GPS coordinate information [141, 142, 143]. 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 visualised 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 [144] used in a range of applications including data clustering, segmentation and texture mapping [8]. For shape registration, as previously discussed in Chapter 2, Arellano et al. propose to extend the registration technique proposed by Jian et al. [41] by augmenting their GMM modelling framework with an extra dimension capturing angles of the normals of the shapes [50]. Their resulting algorithm maps GMMs $p_1(x, \psi)$, generated with the shape $\{(x_1^{(i)}, \psi_1^{(i)})\}$, and $p_2(x, \psi)$ modelled from $\{(x_2^{(j)}, \psi_2^{(j)})\}$ where the angle $\psi^{(i)}$ is the normal direction associated with location $x^{(i)}$ [50].

In this section we propose to use von Mises-Fisher kernels to model the normal vectors of the shape, adding them as an extra dimension to the KDE previously proposed by Jian et al. [41]. The $\mathcal{L}_2$ distance is then shown to have an explicit expression and our proposed cost function is shown in some cases to give improved results to other state of the art techniques. The remainder of this chapter is structured as follows: Section 5.2.1 details some important directional distributions and their scalar products, Section 5.3 proposes four new cost functions for registration using normal information, Section 5.4 outlines some of the implementation details of our method and Section 5.5 presents the experimental results.

5.2 $\mathcal{L}_2$ with 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 $\mathbb{R}^d$). Having a set of observations $\{u^{(i)}\}_{i=1,\cdots,n}$, a kernel density estimate can be used as an approximation of the probability density function of u:

$$p(u) = \sum_{i=1}^{n} \pi_i \, \mathcal{K}\left(u; u^{(i)}, \kappa\right) \tag{5.1}$$

provided that $\mathcal{K}$ is a suitable kernel for random unit vector u. The parameter κ is the scale parameter associated with the kernel $\mathcal{K}$ and the weights π_i are chosen such that $\sum_{i=1}^{n} \pi_i = 1$ and $\pi_i \geq 0, \ \forall i$.

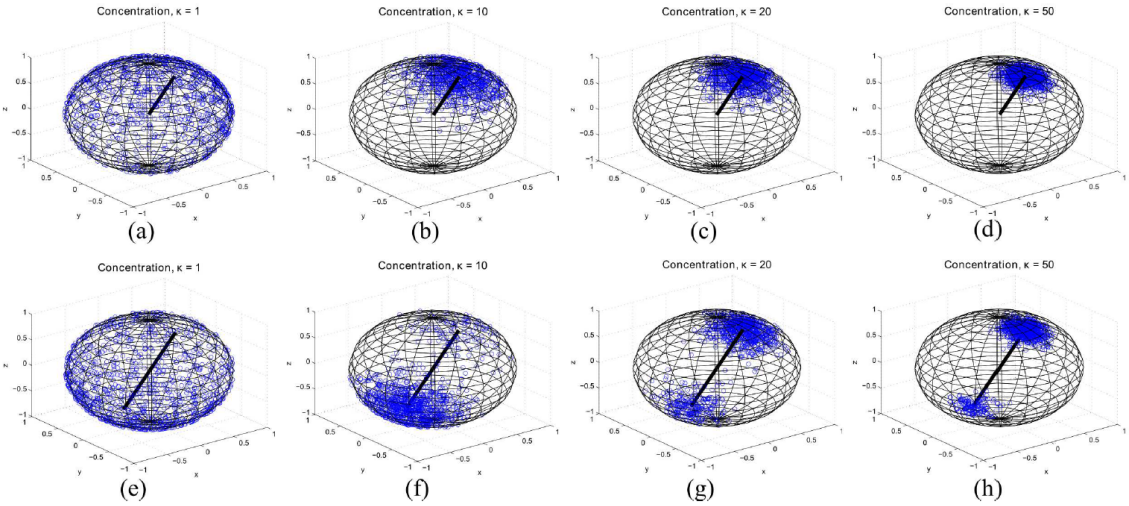


Figure 5.1: Vectors $u \in \mathbb{S}^2$ sampled from the von Mises-Fisher (top row) and Watson distributions (bottom row) for different values of the concentration parameter κ. Image sourced from [8].

5.2.1 Kernels for directional data

von Mises-Fisher kernel

The von Mises-Fisher probability density function for a random unit vector $u \in \mathbb{S}^{d-1}$ is defined as:

$$vMF(u; \mu, \kappa) = C_d(\kappa) \ \exp\left(\kappa \ \mu^T u\right) \tag{5.2}$$

with scalar parameter $\kappa \geq 0$, $\|\mu\| = 1$, and the normalising constant C_d defined as:

$$C_d(\kappa) = \frac{1}{\int_{\mathbb{S}^{d-1}} \exp\left(\kappa \ \mu^T u\right) \ du} = \frac{\kappa^{\frac{d}{2}-1}}{(2\pi)^{\frac{d}{2}} \ \mathcal{I}_{\frac{d}{2}-1}(\kappa)} \tag{5.3}$$

with $\mathcal{I}_{\frac{d}{2}-1}$ the modified Bessel function of order $\frac{d}{2}-1$. 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 $\mathbb{S}^2$ (as defined in Equation 3.26), and the normalising constant in the von Mises-Fisher distribution is [8]:

$$C_3(\kappa) = \frac{\kappa}{4\pi \sinh(\kappa)} \tag{5.4}$$

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

The value of the parameter κ determines the shape of the distribution, with high values of κ creating a distribution highly concentrated about the mean direction μ, and low values of κ creating an almost uniform distribution on $\mathbb{S}^{d-1}$. Figure 5.1 displays different von Mises-Fisher distributions simulated by varying the parameter κ.

Bimodal extension of von Mises-Fisher distribution

When modelling variables which are axially symmetric, in other words random vectors u which have the same distribution as their opposite $-u$, a bimodal extension of the von Mises-Fisher distribution could be used, defined as follows:

$$bvMF(u;\mu,\kappa) = C_d(\kappa)\left(\alpha \exp\left(\kappa\ \mu^T u\right) + (1-\alpha)\exp\left(\kappa\ (-\mu)^T u\right)\right) \tag{5.5}$$

with $\alpha \in [0,1]$ determining the strength of the modes at μ and $-\mu$. The line along $u = -\mu$ and $u = \mu$ is known as the axis direction.

Watson distribution

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

$$W_d(u;\mu,\kappa) = M_d(\kappa)\exp\left(\kappa\ (\mu^T u)^2\right) \tag{5.6}$$

with the normalising constant:

$$M_d(\kappa) = \frac{1}{\int_{\mathbb{S}^{d-1}} \exp\left(\kappa\ (\mu^T u)^2\right)\ du} \tag{5.7}$$

This can be computed as $M_d(\kappa) = M(\frac{1}{2}, \frac{d}{2}, \kappa)$, the confluent hyper-geometric function also known as the Kummer function, which is not directly available but can be approximated. Like the von Mises-Fisher distribution, the value of κ determines the shape of the distribution, however in the case of the Watson distribution κ can take both positive and negative values. Figure 5.1 displays different Watson distributions simulated by varying the parameter κ.

5.2.2 Scalar product of vMF kernels

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:

$$vMF_1 \times vMF_2 = C_d(\kappa_1)\ C_d(\kappa_2) \times \exp\left(\|\kappa_1\mu_1 + \kappa_2\mu_2\|\ \frac{u^T(\kappa_1\mu_1 + \kappa_2\mu_2)}{\|\kappa_1\mu_1 + \kappa_2\mu_2\|}\right) \tag{5.8}$$

In other words, the product $vMF_1 \times vMF_2$ is proportional to $vMF = V_d(u; \mu, \kappa)$ such that:

$$vMF_1 \times vMF_2 = \frac{C_d(\kappa_1)\ C_d(\kappa_2)}{C_d(\kappa)}\ vMF \tag{5.9}$$

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:

$$\langle vMF_1|vMF_2\rangle = \int_{u\in\mathbb{S}^{d-1}} vMF_1 \times vMF_2\ du = \frac{C_d(\kappa_1)\ C_d(\kappa_2)}{C_d(\kappa)} \tag{5.10}$$

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. equation (5.4) for $d = 3$). Alternatively numerical integration can be used as an approximation to equation (5.3) for any value $d > 1$.

When modelling axial symmetric data the bimodal form of the von Mises-Fisher distribution could be used. The scalar product of two such distributions follows from equation 5.10.

5.2.3 Mixtures of von Mises-Fisher and Dirac Kernels

Consider two KDEs, denoted p_1 and p_2 (cf. Eq. 5.1) computed with datasets $\{u_1^{(i)}\}_{i=1,\cdots,n_1}$ and $\{u_2^{(j)}\}_{j=1,\cdots,n_2}$, and kernels $\mathcal{K}_{u_1}$ and $\mathcal{K}_{u_2}$ respectively. Their scalar product is then:

$$\langle p_1|p_2\rangle = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \pi_1^{(i)}\ \pi_2^{(j)}\ \int_u \mathcal{K}_{u_1}(u; u_1^{(i)})\ \mathcal{K}_{u_2}(u; u_2^{(j)})du \tag{5.11}$$

Consider on the other hand the case where p_1 is computed using two types of kernels

$\mathcal{K}_{u_1}$ and $\mathcal{K}_{x_1}$ fitted to the points $\{u_1^{(i)}\}_{i=1,\cdots,n_1}$ and $\{x_1^{(i)}\}_{i=1,\cdots,n_1}$ as follows:

$$p_1(x,u) = \sum_{i=1}^{n_1} \pi_1^{(i)} \; \mathcal{K}_{u_1}(u; u_1^{(i)}) \; \mathcal{K}_{x_1}(x; x_1^{(i)}) \tag{5.12}$$

If p_2 is computed similarly with kernels $\mathcal{K}_{u_2}$ and $\mathcal{K}_{x_2}$, and data $\{u_2^{(j)}\}_{j=1,\cdots,n_2}$ and $\{x_2^{(j)}\}_{j=1,\cdots,n_2}$, then the cross product between p_1 and p_2 can be written as:

$$\langle p_1 | p_2 \rangle = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \pi_1^{(i)} \; \pi_2^{(j)} \int_u \mathcal{K}_{u_1}(u; u_1^{(i)}) \; \mathcal{K}_{u_2}(u; u_2^{(j)}) \; du \int_x \mathcal{K}_{x_1}(x; x_1^{(i)}) \; \mathcal{K}_{x_2}(x; x_2^{(j)}) \; dx \tag{5.13}$$

In the next section we propose modelling shape observations and their normals using a combination of Dirac, von Mises-Fisher and Normal kernels. Computing the $\mathcal{L}_2$ distance between the proposed KDEs relies on the scalar products between their associated kernels. All of these scalar products are summarised in Table 5.1.

	$x \in \mathbb{R}^d$		$u \in \mathbb{S}^{d-1}$	
	$\delta(x-\mu_1)$	$\mathcal{N}(x;\mu_1,h_1^2)$	$\delta(u-\mu_1)$	$vMF(u;\mu_1,\kappa_1)$
$\delta(x-\mu_2)$	✗	$\mathcal{N}(\mu_1;\mu_2,h_1^2)$, [49]	✗	✗
$\mathcal{N}(x;\mu_2,h_2^2)$	$\mathcal{N}(\mu_1;\mu_2,h_2^2)$, [49]	$\mathcal{N}(\mu_1;\mu_2,h_1^2 + h_2^2)$ [41]	✗	✗
$\delta(u-\mu_2)$	✗	✗	✗	$C_d(\kappa_1)\exp(\kappa_1\,\mu_1^T\mu_2)$
$vMF(u;\mu_2,\kappa_2)$	✗	✗	$C_d(\kappa_2)\exp(\kappa_2\,\mu_2^T\mu_1)$	$\frac{C_d(\kappa_1)\;C_d(\kappa_2)}{C_d(\lVert\kappa_1\mu_1+\kappa_2\mu_2\rVert)}$

Table 5.1: Scalar products for Gaussian ($\mathcal{N}$), von Mises-Fisher (vMF) and Dirac (δ) kernels.

5.3 Cost functions for Registration

Given two sets of observations $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\cdots,n_1}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,\cdots,n_2}$, 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 $\tilde{S}_1 = \{(\tilde{x}_1^{(i)}, \tilde{u}_1^{(i)})\}_{i=1,\cdots,n_1}$. 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 $x \in \mathbb{R}^d$, $u \in \mathbb{S}^{d-1}$, and $(x, u) \in \mathbb{R}^d \times \mathbb{S}^{d-1}$. The $\mathcal{L}_2$ distance between p_1 and p_2 can then be computed as $\mathcal{L}_2(p_1, p_2) = \|p_1 - p_2\|^2 = \|p_1\|^2 + \|p_2\|^2 - 2\langle p_1|p_2\rangle$. Minimising $\mathcal{L}_2(p_1, p_2)$ to estimate the transformation ϕ is equivalent to minimising $\|p_1\|^2 - 2\langle p_1|p_2\rangle$ since the distribution p_2 is independent of ϕ. When ϕ is a rigid transformation (ie. a transformation which preserves distances between points, such as a rotation or translation), the term $\|p_1\|^2$ can also be discarded as the $\mathcal{L}_2$ norm of a pdf is invariant under rigid transformation. Therefore, minimising the $\mathcal{L}_2$ distance between p_1 and p_2 is equivalent to minimising $-\langle p_1|p_2\rangle$, or maximising $\langle p_1|p_2\rangle$, when ϕ is a rigid transformation.

5.3.1 Registration using $u \in \mathbb{S}^{d-1}$

To model the first set of normals $\{u_1^{(i)}\}$ we propose a KDE p_1 with a von Mises-Fisher kernel $vMF(u; \tilde{u}_1^{(i)}, \kappa)$ fitted to each normal $\tilde{u}_1^{(i)}$ in $\tilde{S}_1$:

$$p_1(u) = \frac{1}{n_1} \sum_{i=1}^{n_1} vMF(u; \tilde{u}_1^{(i)}, \kappa_1) \tag{5.14}$$

For the second set of normal vectors $\{u_2^{(j)}\}$ we propose using either the Dirac or von Mises-Fisher distributions.

Dirac distribution

Using the empirical distribution we propose the model:

$$p_2(u) = \frac{1}{n_2} \sum_{j=1}^{n_2} \delta(u - u_2^{(j)}). \tag{5.15}$$

Using the definitions for $\langle p_1|p_2\rangle$ as given in Table 5.1, the cost function used to estimate θ by minimising $\|p_1\|^2 - 2\langle p_1|p_2\rangle$ can then be defined as follows:

$$\widehat{\theta} = \arg\min_{\theta} \left\{ \mathcal{C}_{\delta}^{u} = \frac{C_d(\kappa_1)C_d(\kappa_1)}{n_1 n_1} \sum_{i=1}^{n_1}\sum_{j=1}^{n_1} \frac{1}{C_d\left(\|\kappa_1 \tilde{u}_1^{(i)} + \kappa_1 \tilde{u}_1^{(j)}\|\right)} - 2\frac{C_d(\kappa_1)}{n_1 n_2} \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \exp(\kappa_1 \tilde{u}_1^{(i)T} u_2^{(j)}) \right\}, \quad (5.16)$$

When ϕ is a rigid transformation, maximising the scalar product $\langle p_1|p_2\rangle$ can be used to estimate θ since $\|p_1\|^2$ is invariant to changes in θ, and $\mathcal{C}_{\delta}^{u}$ can be redefined as

$$\widehat{\theta} = \arg\max_{\theta} \left\{ \mathcal{C}_{\delta}^{u} = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \exp(\kappa_1 \tilde{u}_1^{(i)T} u_2^{(j)}) \right\}. \quad (5.17)$$

and can be easily computed for all values of d. This is one of the main advantages of using the Dirac distribution to model one set of normal vectors.

Von Mises-Fisher distribution

Using the von Mises-Fisher distribution we propose the model:

$$p_2(u) = \frac{1}{n_2} \sum_{j=1}^{n_2} vMF(u; u_2^{(j)}, \kappa_2). \quad (5.18)$$

Using the definitions for $\langle p_1|p_2\rangle$ as given in Table 5.1, the cost function used to estimate θ by minimising $\|p_1\|^2 - 2\langle p_1|p_2\rangle$ can then be defined as follows:

$$\widehat{\theta} = \arg\min_{\theta} \left\{ \mathcal{C}^{u} = \frac{C_d(\kappa_1)C_d(\kappa_1)}{n_1 n_1} \sum_{i=1}^{n_1}\sum_{j=1}^{n_1} \frac{1}{C_d\left(\|\kappa_1 \tilde{u}_1^{(i)} + \kappa_1 \tilde{u}_1^{(j)}\|\right)} - 2\frac{C_d(\kappa_1)C_d(\kappa_2)}{n_1 n_2} \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \frac{1}{C_d\left(\|\kappa_1 \tilde{u}_1^{(i)} + \kappa_2 u_2^{(j)}\|\right)} \right\}. \quad (5.19)$$

Unlike $\mathcal{C}_{\delta}^{u}$, both terms in $\mathcal{C}^{u}$ depend on the normalising constant $C_d(\kappa)$, and the computation of $\mathcal{C}^{u}$ requires numerical integration when $d \neq 3$.

5.3.2 Registration using $x \in \mathbb{R}^d$

Next we propose to model a KDE with a Gaussian kernel $\mathcal{N}(x; \tilde{x}_1^{(i)}, h)$ fitted to each normal $\tilde{x}_1^{(i)}$ in $\tilde{S}_1$:

$$p_1(x) = \frac{1}{n_1} \sum_{i=1}^{n_1} \mathcal{N}(x; \tilde{x}_1^{(i)}, h_1) \tag{5.20}$$

We also propose to model the second set of points $\{x_2^{(j)}\}$ using the Gaussian distribution:

$$p_2(x) = \frac{1}{n_2} \sum_{j=1}^{n_2} \mathcal{N}(x; x_2^{(i)}, h_2) \tag{5.21}$$

Using the definition for the scalar product between two Gaussian kernels (cf. Table 5.1), the following cost function is proposed to estimate θ:

$$\begin{aligned} \widehat{\theta} = \arg\min_{\theta} \Bigg\{ \mathcal{C}^x = & \frac{1}{n_1 n_1} \sum_{j=1}^{n_1} \sum_{i=1}^{n_1} \frac{1}{\sqrt{4h_1^2 \pi}} \exp\left(\frac{-\|\tilde{x}_1^{(j)} - \tilde{x}_1^{(i)}\|^2}{4h_1^2} \right) \\ & - \frac{2}{n_1 n_2} \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} \frac{1}{\sqrt{2(h_1^2 + h_2^2)\pi}} \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2(h_1^2 + h_2^2)} \right) \Bigg\}. \end{aligned} \tag{5.22}$$

This cost function is equivalent to that proposed for shape registration by Jian et al. [41] and as it takes the form of a summation of Gaussians, fast approximation schemes such as the fast Gauss Transform can be used to speed up its computation [145]. 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:

$$\begin{aligned} \widehat{\theta} = \arg\min_{\theta} \Bigg\{ \mathcal{C}_\delta^x = & \frac{1}{n_1 n_1} \sum_{j=1}^{n_1} \sum_{i=1}^{n_1} \frac{1}{\sqrt{4h_1^2 \pi}} \exp\left(\frac{-\|\tilde{x}_1^{(j)} - \tilde{x}_1^{(i)}\|^2}{4h_1^2} \right) \\ & - \frac{2}{n_1 n_2} \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} \frac{1}{\sqrt{2h_1^2 \pi}} \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2h_1^2} \right) \Bigg\}, \end{aligned} \tag{5.23}$$

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

5.3.3 Entangled Registration using $x \in \mathbb{R}^d$ and $u \in \mathbb{S}^{d-1}$

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

$$p_1(x,u) = \frac{1}{n_1}\sum_{i=1}^{n_1} vMF(u;\tilde{u}_1^{(i)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(i)},h_1) \tag{5.24}$$

For the second set of observation we again propose two methods for modelling the point and normal vectors.

Dirac distribution

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:

$$p_2(x,u) = \frac{1}{n_2}\sum_{j=1}^{n_2} \delta(u-u_2^{(j)})\mathcal{N}(x;x_2^{(j)},h_2). \tag{5.25}$$

Then in order to optimise for the parameter θ we minimise the following:

$$\begin{aligned}\hat{\theta} = \arg\min_{\theta}\Bigg\{\mathcal{C}_\delta^{x,u} = &\frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1}\langle vMF(u;\tilde{u}_1^{(i)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(i)},h_1)|vMF(u;\tilde{u}_1^{(j)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(j)},h_1)\rangle \\ &- \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1}\langle vMF(u;\tilde{u}_1^{(i)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(i)},h_1)|\delta(u-u_2^{(j)})\mathcal{N}(x;x_2^{(j)},h_2)\rangle\Bigg\}\end{aligned} \tag{5.26}$$

Using the appropriate scalar product definitions given in Table 5.1, the proposed cost function can be written explicitly as:

$$\begin{aligned}\hat{\theta} = \arg\min_{\theta}\Bigg\{\mathcal{C}_\delta^{x,u} = &\frac{C_d(\kappa_1)C_d(\kappa_1)}{n_1 n_1\sqrt{4h_1^2\pi}}\sum_{i=1}^{n_1}\sum_{j=1}^{n_1}\frac{1}{C_d\left(\|\kappa_1\tilde{u}_1^{(i)}+\kappa_1\tilde{u}_1^{(j)}\|\right)}\exp\left(\frac{-\|\tilde{x}_1^{(j)}-\tilde{x}_1^{(i)}\|^2}{4h_1^2}\right) \\ &- \frac{2C_d(\kappa_1)}{n_1 n_2\sqrt{2(h_1^2+h_2^2)\pi}}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\exp(\kappa_1\ {u_2^{(j)}}^T\tilde{u}_1^{(i)})\exp\left(\frac{-\|x_2^{(j)}-\tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right)\Bigg\}\end{aligned} \tag{5.27}$$

In $\mathcal{C}_\delta^{x,u}$, the term $\langle p_1|p_2\rangle$ is independent of the normalising constant $C_d(\kappa)$ and can be computed for any dimension d. Therefore, when ϕ is a rigid transformation, θ can be estimated for any dimension d by maximising the scalar product $\langle p_1|p_2\rangle$ as follows:

$$\hat{\theta} = \arg\max_{\theta} \left\{ \mathcal{C}_\delta^{x,u} = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \exp(\kappa_1 \ {u_2^{(j)}}^T \tilde{u}_1^{(i)}) \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right)\right\}. \quad (5.28)$$

Von Mises-Fisher

We also propose an alternate KDE with vMF kernels fitted to the normal vectors $\{u_2^{(j)}\}$ as in Equation 5.24:

$$p_2(x,u) = \frac{1}{n_2}\sum_{j=1}^{n_2} vMF(u; u_2^{(j)}, \kappa_2)\mathcal{N}(x; x_2^{(j)}, h_2). \quad (5.29)$$

Then in order to optimise for the parameter θ we minimise the following:

$$\hat{\theta} = \arg\min_{\theta} \left\{ \mathcal{C}^{x,u} = \frac{1}{n_1 n_1}\sum_{j=1}^{n_1}\sum_{i=1}^{n_1} \langle vMF(u;\tilde{u}_1^{(i)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(i)},h_1)|vMF(u;\tilde{u}_1^{(j)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(j)},h_1)\rangle \right.$$
$$\left. - \frac{2}{n_1 n_2}\sum_{j=1}^{n_2}\sum_{i=1}^{n_1} \langle vMF(u;\tilde{u}_1^{(i)},\kappa_1)\mathcal{N}(x;\tilde{x}_1^{(i)},h_1)|vMF(u;u_2^{(j)},\kappa_2)\mathcal{N}(x;x_2^{(j)},h_2)\rangle \right\} \quad (5.30)$$

Using the appropriate scalar product definitions given in Table 5.1, the proposed cost function can be written explicitly as:

$$\hat{\theta} = \arg\min_{\theta} \left\{ \mathcal{C}^{x,u} = \frac{C_d(\kappa_1)C_d(\kappa_1)}{n_1 n_1 \sqrt{4h_1^2\pi}}\sum_{i=1}^{n_1}\sum_{j=1}^{n_1} \frac{1}{C_d\left(\|\kappa_1\tilde{u}_1^{(i)} + \kappa_1\tilde{u}_1^{(j)}\|\right)} \exp\left(\frac{-\|\tilde{x}_1^{(j)} - \tilde{x}_1^{(i)}\|^2}{2h_1^2}\right) \right.$$
$$\left. - 2\frac{C_d(\kappa_1)C_d(\kappa_2)}{n_1 n_2\sqrt{2(h_1^2+h_2^2)\pi}}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \frac{1}{C_d\left(\|\kappa_1\tilde{u}_1^{(i)} + \kappa_2 u_2^{(j)}\|\right)} \exp\left(\frac{-\|x_2^{(j)} - \tilde{x}_1^{(i)}\|^2}{2(h_1^2+h_2^2)}\right) \right\} \quad (5.31)$$

Unlike $\mathcal{C}_\delta^{x,u}$, both terms in $\mathcal{C}^u$ depend on the normalising constant $C_d(\kappa)$, and the computation of $\mathcal{C}^{x,u}$ requires numerical integration when $d \neq 3$.

5.4 Implementation Details

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.

5.4.1 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 $\mathcal{C}^u_\delta$ are invariant to translation. Therefore, when estimating a rigid transformation, to ensure all cost functions are evaluated equally, we omit a translational transformation and only consider data differing by a rotation. When estimating the 3D rotation transformation we use the quaternion representation $\phi(x, \theta) = (q_1, q_2, q_3, q_4)$ as described in Section 3.2.1.

For shapes differing by a non-rigid deformation, we estimate a Thin Plate Spline transformation to register the shapes, as defined in Equation 3.16. In this case, we do not include any regularisation terms when estimating the TPS transformation, although a term such as that given in Equation 3.22 can be added by the user if necessary. The N control points c_j used to control the TPS transformations in our non-rigid experiments are chosen uniformly on a grid spanning the bounding box of the model shape.

5.4.2 Algorithm

Given two point sets $\{x_1^{(i)}\}_{i=1,..n_1}$ and $\{x_2^{(j)}\}_{j=1,..n_2}$ representing the model and target shapes, our strategy for estimating the transformation $\phi(x, \theta)$ is summarised in Algorithm 2.

Algorithm 2 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 $\{u_1^{(i)}\}_{i=1,..n_1}$ and $\{u_2^{(j)}\}_{j=1,..n_2}$ from $\{x_1^{(i)}\}_{i=1,..n_1}$ and $\{x_2^{(j)}\}_{j=1,..n_2}$.

Choose m points $\{x_1^{(i)}\}_{i=1,..m}$ and $\{x_2^{(j)}\}_{j=1,..m}$ and their associated unit normal vectors $\{u_1^{(i)}\}_{i=1,..m}$ and $\{u_2^{(j)}\}_{j=1,..m}$ 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 $\kappa_1 = \kappa_2 = \kappa$. In order to avoid local minima, we implement a simulated annealing strategy which involves gradually decreasing the value of h from h_{init} to h_{final} by multiplying by h_{step}, and similarly increasing the value of κ from κ_{init} to κ_{final} by multiplying by κ_{step}. Starting the optimisation at several initial guesses can also help avoid local minima, however in all of the results presented we use only the identity transformation as the initial guess. The values chosen for each of these parameters can be found in Appendix B. As the computation time of our proposed algorithms are dependent on the number of points in S_1 and S_2, we reduce the number of points processed by choosing a random sample of m points and their associated unit normal vectors from both S_1 and S_2.

5.4.3 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 at each sampled point by calculating the first derivative of the curve. For 3D shapes in the form of meshes, with edge and vertex information available, 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 can also compute the normal vectors without exploiting the connectivity of a vertex. This method involves using the nearest neighbourhood structure of the vertex instead, and can also be used to compute the normal vector of a point when no edge or face information is available. This method fits a plane to the N_k nearest neighbours about the point $x^{(i)}$ and the vector perpendicular to this plane is set as the normal vector of $x^{(i)}$ [146, 147].

5.4.4 Optimisation details

When estimating a rotation we use the Matlab optimisation function *fmincon* to minimise the cost functions. This optimisation function minimises constrained, non-linear multivariate functions and allows us to impose constraints on the estimated parameter $\hat{\theta}$. For 3D rotations, upper and lower bounds are added to ensure that each value q_i in the estimated quaternion (q_1, q_2, q_3, q_4) lies in the range $[-1,\ 1]$. The minimization algorithm used is the interior point algorithm [148, 149] which is a gradient ascent technique.

When estimating a TPS transformation we used the Matlab function *fminunc*, which minimises unconstrained multivariate functions. In this case the minimisation algorithm used is the Quasi-Newton method [136], which is also a gradient ascent algorithm. In the case of the cost functions $\mathcal{C}^x$, $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$, no analytical gradients are computed. Instead, numerical methods are used by *fmincon* and *fminunc* to estimate the gradient of the cost functions.

5.4.5 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, rather than computing the double sum $\sum_i^{n_1} \sum_j^{n_2}$ over all point pairs, this summation reduces to $\sum_i^n$ for all cost functions. To compute correspondences we used the method proposed by Yang et al.[33]. Their correspondence estimation method uses a combination of both global and local distance features. First, a global distance between the model and target point sets $\{x_1^{(i)}\}$ and $\{x_2^{(i)}\}$ is computed, which is based on the squared Euclidean distance between each pair of points in the point sets. They then estimate the local distance which measures the difference in

neighbourhood structure between each pair of points in the point sets. They combine both the local and global distance into a cost matrix, and use this to estimate a set of point correspondences.

They also implement an annealing scheme which is designed to slowly change the cost minimisation from local to global. This ensures that in the first annealing steps the correspondences estimated are based mostly on the local distance and in the final annealing steps more emphasis is given to the global distance. When implementing this correspondence estimation method in our algorithm, we also implemented this transition from global to local distance minimisation, incorporating it into our simulated annealing strategy.

5.4.6 Comparisons

To evaluate our algorithm we compared our results to several state of the art registration techniques [41, 150, 151, 33]. The parameters chosen for each of these techniques is given in Appendix B. To ensure that a fair comparison between Jian'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 et al.[1] so that they coincided with those implemented with our proposed cost functions. For example, we implemented the same simulated annealing framework for $\mathcal{C}^x$ as for $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}_\delta$ and $\mathcal{C}^{x,u}$. The optimisation changes made enhanced the results achievable by C^x by allowing the function to avoid local minima and converge to a good solution.

5.4.7 Evaluation

In all of our experiments we chose the model and target point sets $\{x_1^{(i)}\}_{i=1,..n_1}$ and $\{x_2^{(j)}\}_{j=1,..n_2}$ to be of equal size with $n_1 = n_2 = n$. The ground truth point correspondences between the model and target shapes $\{x_1^{(i)}, x_2^{(i)}\}_{i=1,..n}$ are also known in all cases. Therefore when evaluating the results of all algorithms tested, we computed the mean square error (MSE) between corresponding points in the transformed model ($\{\tilde{x}_1^{(i)}\}$)

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

and target point sets as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\|\tilde{x}_1^{(i)} - x_2^{(i)}\| \tag{5.32}$$

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 compute $\phi(x, \hat{\theta})$.

5.5 Experimental Results

In this section we evaluate our proposed cost functions and show how they compare to other state of the art registration techniques. In Sections 5.5.1 and 5.5.2 we apply the cost functions to the problem of registering both 2D data and 3D data differing by a rotation transformation. In this case we maximise the scalar product $\langle p_1|p_2\rangle$. In Section 5.5.3 and 5.5.4 we also evaluate the cost functions on 2D and 3D data differing by a non-rigid transformation and minimise $\|p_1\|^2 - 2\langle p_1|p_2\rangle$ to estimate θ. In Section 5.5.5 we give details about the computational cost of our algorithm and in Section 5.5.6 we give a short summary of the findings in this section.

For our 2D experiments we chose to register point clouds consisting of points sampled from parametric curves. The normal vectors at each point can then be computed analytically as the first derivative of the parametric curve at the given point. For 3D data, we chose to register point clouds consisting of the vertices of 3D meshes. In this case, the edge and face information of the mesh can be used to compute the normal vectors of each vertex. Using data sets with ground truth normal vectors ensures that we can assess how the proposed cost functions perform when there is no noise or error associated with either the points or their normal vectors. We also added noise and removed points from the point clouds in order to simulate the errors and occlusions which are typically associated with real world data sets.

5.5.1 2D Rotation Registration

In this section we consider two curves $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,..n}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,..n}$, which are represented by their point locations $\{x^{(i)}\} \in \mathbb{R}^2$ and normal vectors $\{u^{(i)}\} \in \mathbb{S}$, as in Figure 5.2. Curves S_1 and S_2 differ by a rotation ϕ which is defined as

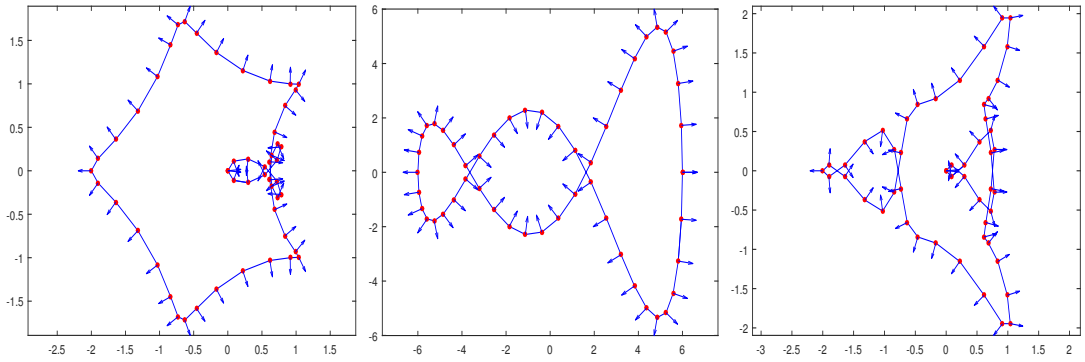


Figure 5.2: Three different 2D parametric curves used in our experiments, sampled at 50 locations, along with their normal vectors.

$\phi(x, \theta) = \mathrm{R}x$, where R is a 2D rotation matrix controlled by the angle θ, as in Equation 3.8. We assess the estimation of the rotation angle θ using the cost functions $\mathcal{C}^x$ [41], $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. When using $\mathcal{C}^{x,u}$ and $\mathcal{C}^u$, which depend on the von Mises-Fisher normalising constant $C_d(\kappa)$, we artificially define u on $\mathbb{S}^2$ instead of $\mathbb{S}$ by adding a third dimensional coordinate to the normal vector which is set to zero. The normalising constant $C_3(\kappa)$ can be computed analytically. The point locations $\{x^{(i)}\}$ remain unchanged. When testing our results we found that $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ as well as $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ perform similarly, so for ease of comparison we only present results for $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ in the following section. The similarity in results is due to the fact that fitting a dirac kernel to each normal vector associated with a given shape, as proposed by the cost functions $\mathcal{C}^{x,u}_\delta$ and $\mathcal{C}^u_\delta$, creates a similar model to that generated by fitting a von Mises-Fisher kernel to each normal vector, when the sample of normal vectors used is large and randomly selected, as is the case in our proposed experiment. Further comparisons with $\mathcal{C}^u_\delta$ and $\mathcal{C}^{x,u}_\delta$ can be found in Appendix C.

Experimental Design

1. Rotation: To create the shapes for our first experiment, we sampled a parametric curve at 50 locations $\{x_1^{(i)}\}_{i=1,..50}$ and computed their corresponding normal vectors $\{u_1^{(i)}\}_{i=1,..50}$ to create $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,\cdots,50}$, as in Figure 5.2. We rotated the points and normal vectors of S_1 by θ degrees, creating a second curve S_2. Examples of the curves S_1 and S_2 can be seen in Figure 5.5. Using the curve S_1 as the model, we

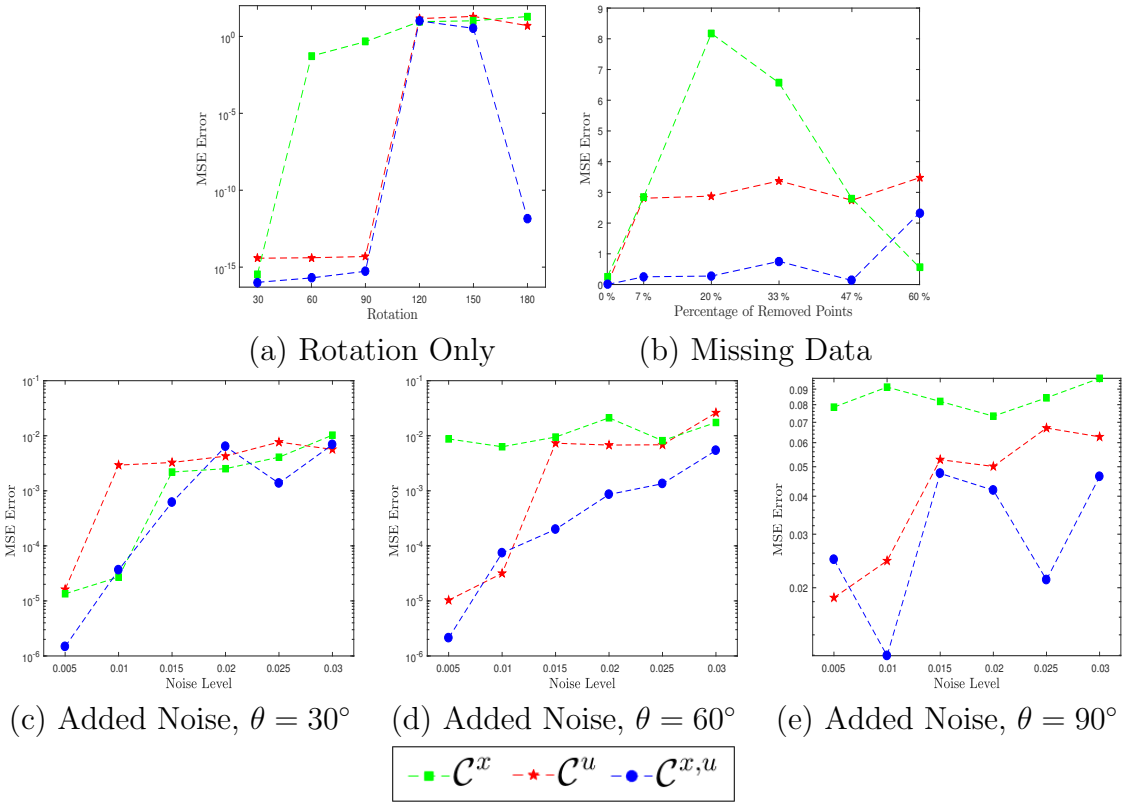
estimate θ by registering it to S_2. Several values of θ were tested, and for each value we created and registered 10 pairs of curves S_1 and S_2.

2. Rotation with missing data: In our second experiment we sampled a parametric curve at 150 locations $\{x_1^{(i)}\}_{i=1,..150}$ and computed their corresponding normal vectors $\{u_1^{(i)}\}_{i=1,..150}$ to create $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,..150}$. We then rotated S_1 by $\theta = 60$ degrees to create the curve S_2. Then we removed a percentage of points, in order, from the end of the curve S_2. Some examples of the curves S_1 and S_2 can be seen in Figure 5.6. Again we estimate θ by registering S_1 to S_2. 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.

3. Rotation with added noise: In our third experiment we sampled a parametric curve at 150 locations and computed the corresponding normal vectors analytically using the parametric equation of the curve, creating the shape $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,..150}$. To create the shape S_2, we first rotated S_1 by θ degrees. We then added Gaussian noise with mean 0 and standard deviation σ to each of the points, creating the point set $\{x_2^{(j)}\}_{j=1,..150}$. To compute the corresponding normal vectors $\{u_2^{(i)}\}_{i=1,..150}$, we fit a spline to the noisy data points, using a smoothing parameter with the fitting algorithm to ensure that the spline does not interpolate the noisy data exactly, but instead approximates it. We use the Matlab function '*csaps*' to fit the spline, and set the smoothing parameter to 0.999. We used the derivative of this spline to estimate the normal vectors of the noisy points. Using a spline approximation, as opposed to a spline interpolation, to compute the normal vectors ensures that the normal vectors still contain some useful information and are not too noisy. Some examples of the curves S_1 and S_2 can be seen in Figure 5.7. Again we estimate θ by registering S_1 to S_2. For each level of noise tested ($\sigma = 0.005, 0.01, 0.015, 0.02, 0.025, 0.03$) we generating 150 pairs of curves S_1 and S_2 on which we tested the estimation of θ. We also tested several values of θ to see how the registration results differ as the rotation increases.

Results

In Figure 5.3 we present the average MSE errors computed for $\mathcal{C}^x$, $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ in all three experiments. The results of our first experiment can be seen in Figure 5.3(a). In



(a) Rotation Only (b) Missing Data

(c) Added Noise, $\theta = 30°$ (d) Added Noise, $\theta = 60°$ (e) Added Noise, $\theta = 90°$

Figure 5.3: MSE results for each of our experiments on 2D data differing by a 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). The graphs in (c), (d) and (e) represent the results on noisy data when $\theta = 30°, 60°$ and $90°$ respectively. The MSE value given at each noise level is the average over 150 curve registration results.

this case we can see that $\mathcal{C}^{x,u}$ performs the best, followed by $\mathcal{C}^u$ and then $\mathcal{C}^x$. As some of the shapes tested are symmetric, the cost functions can fall into, or out of, local minima depending on the shape tested, which is why $C^{x,u}$ seems to perform better at $180°$ than $150°$. Figure 5.5 gives a sample of some of the registration results as well as a graph of the corresponding cost functions. From these results we can see that in the cases where only one of $\mathcal{C}^u$ and $\mathcal{C}^x$ performs well, $\mathcal{C}^{x,u}$ still gives a good result as it combines the information from both cost functions.

The results of our second experiment can be seen in Figure 5.3(b). Again we found that $\mathcal{C}^{x,u}$ performs the best, followed by $\mathcal{C}^u$ and then $\mathcal{C}^x$. We also found that as the

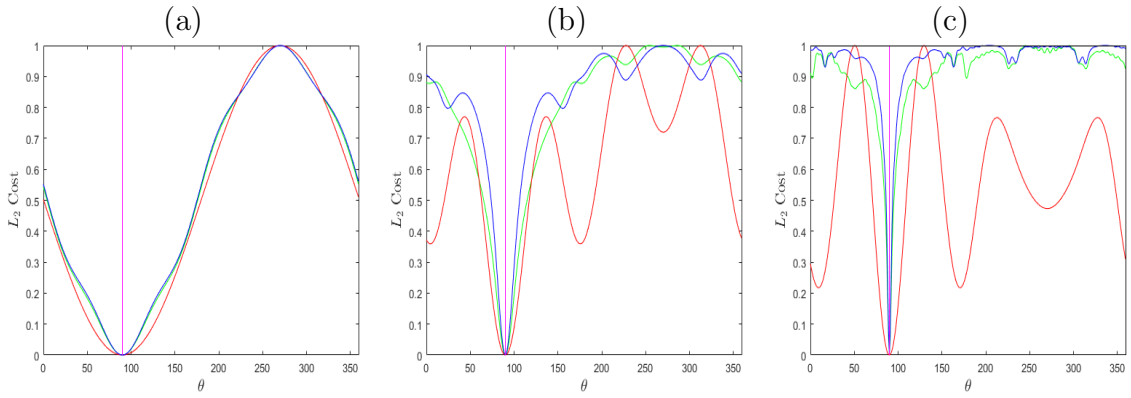


Figure 5.4: The effect that our simulated annealing strategy has on the cost functions as θ ranges from 1° to 360°. To avoid local minima, h is gradually decreased and κ is gradually increased until $h = h_{final}$ and $\kappa = \kappa_{final}$. Here the curve shown in Figure 5.2(a) is S_1, and was rotated by an angle of $\theta_{GT} = 90°$ to generate S_2. Green: $\mathcal{C}^x$; Red: $\mathcal{C}^u$; Blue: $\mathcal{C}^{x,u}$; Pink: θ_{GT}.

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. The rise in error associated with $\mathcal{C}^x$ as θ approaches 20%, and the fall in error afterwards as the number of points removed increases further, is again due to the symmetry of the shapes tested. Similar results were found for other values of θ tested.

The results of our third experiment can be seen in Figure 5.3(c),(d) and (e) when estimating $\theta = 30°$, $\theta = 60°$, and $\theta = 90°$ respectively. In this case we found that when θ is relatively small, around 30°, $\mathcal{C}^{x,u}$ and $\mathcal{C}^x$ perform similarly and slightly outperform $\mathcal{C}^u$. However, as θ increases to 60° and 90°, the performance of $\mathcal{C}^x$ gets worse and both $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ outperform it, with $\mathcal{C}^{x,u}$ performing the best in both cases, for all levels of noise added to the data.

As explained in Section 5.4.2, we use a simulated annealing strategy to avoid local minima when registering the shapes S_1 and S_2. The effect of this strategy on the cost functions can be seen in Figure 5.4.

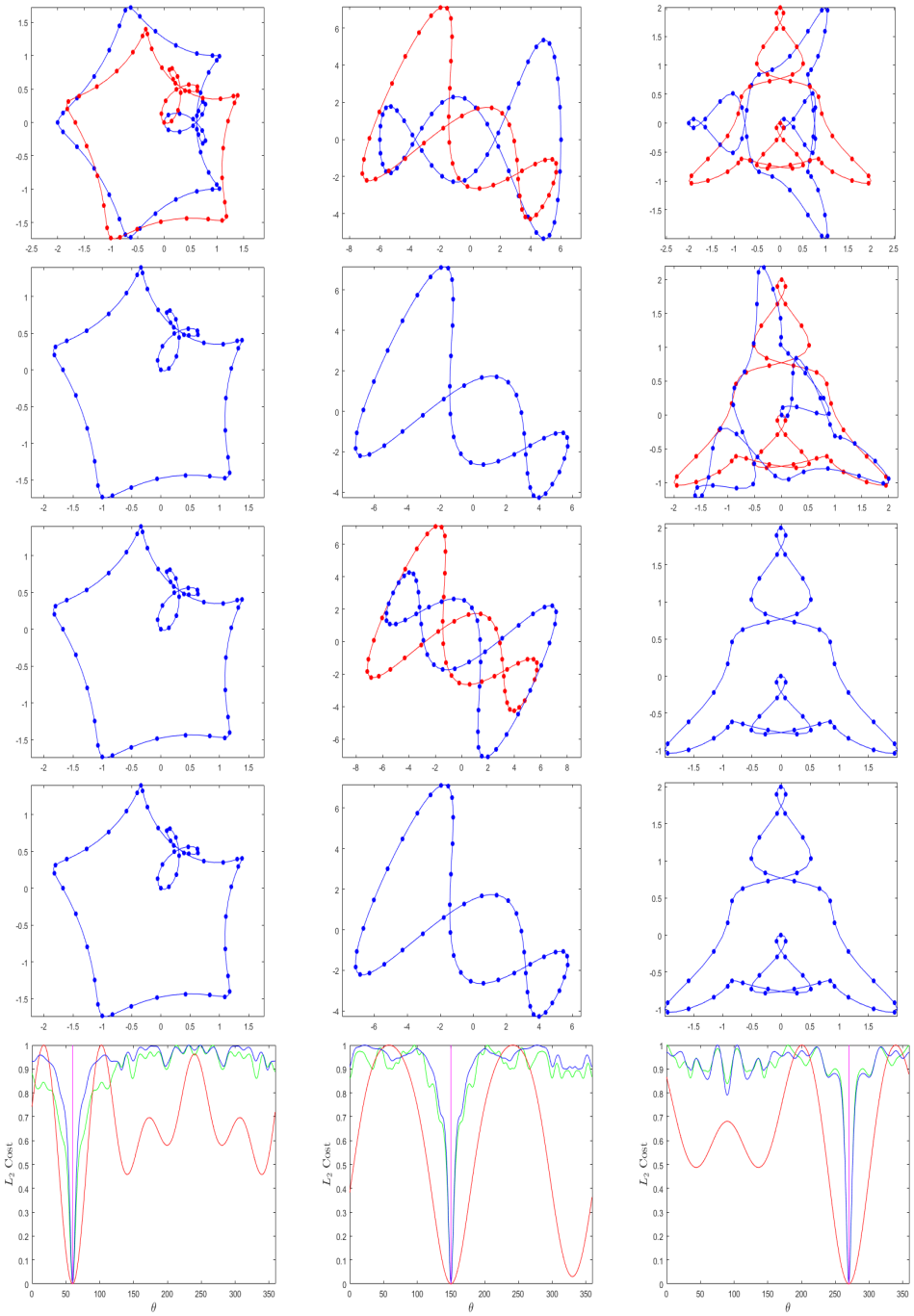


Figure 5.5: Sample registration results for rotation estimation with 2D data. Row 1: Model Curve (blue) and Target Curve (red). Row 2 - 4: Registration result using $\mathcal{C}^x$, $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ respectively. Row 5 shows the value of each of the cost functions when θ ranges from 1°to 360°. In these graphs red represents $\mathcal{C}^u$, green represents $\mathcal{C}^x$ and blue represents $\mathcal{C}^{x,u}$. The pink line indicates the value of θ_{GT}.

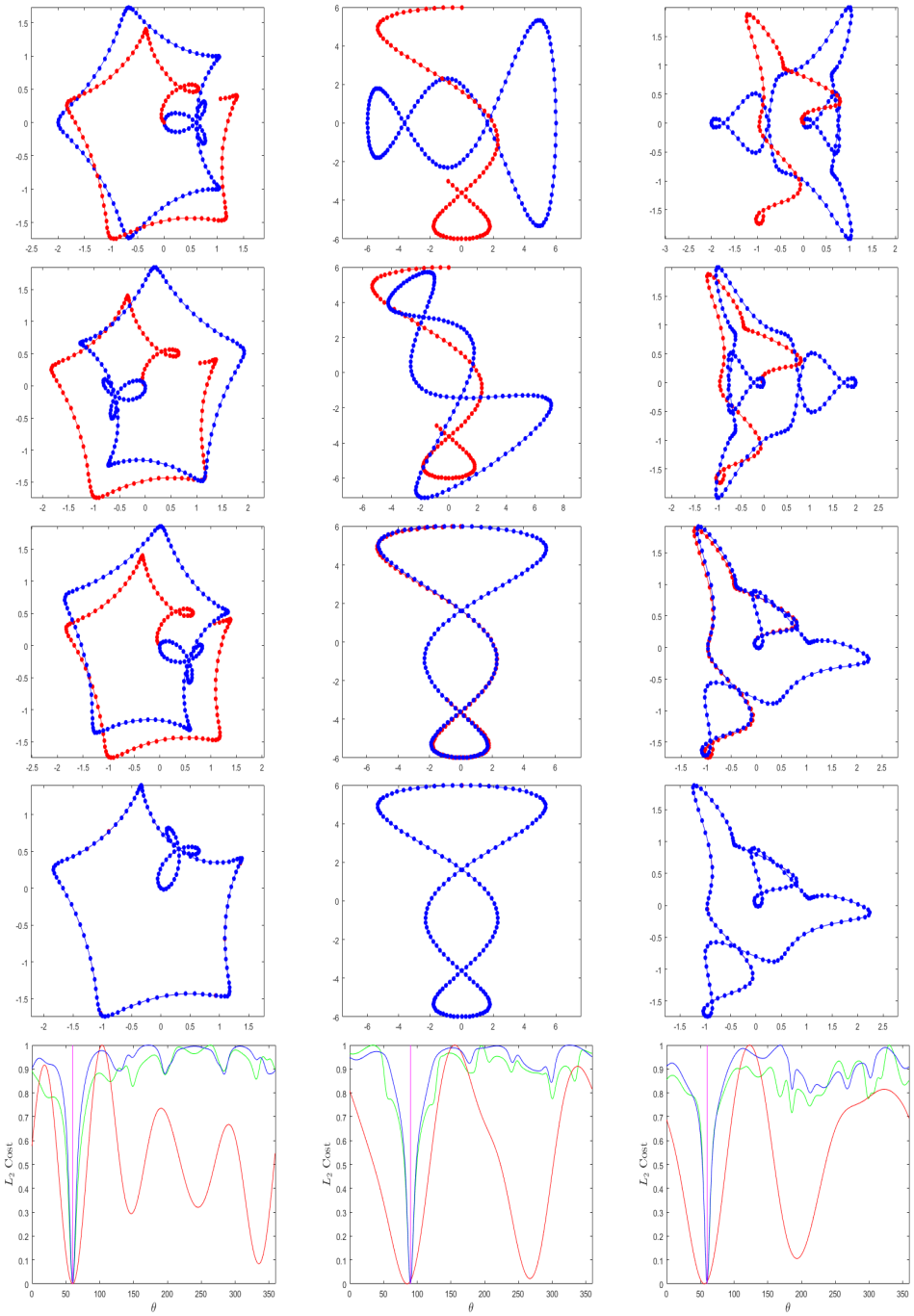


Figure 5.6: Sample registration results for rotation estimation with missing data. Row 1: Model Curve (blue) and Target Curve (red). Row 2 - 4: Registration result using $\mathcal{C}^x$, $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ respectively. Row 5 shows the value of each of the cost functions when θ ranges from 1°to 360°. In these graphs red represents $\mathcal{C}^u$, green represents $\mathcal{C}^x$ and blue represents $\mathcal{C}^{x,u}$. The pink line indicates the value of θ_{GT}.

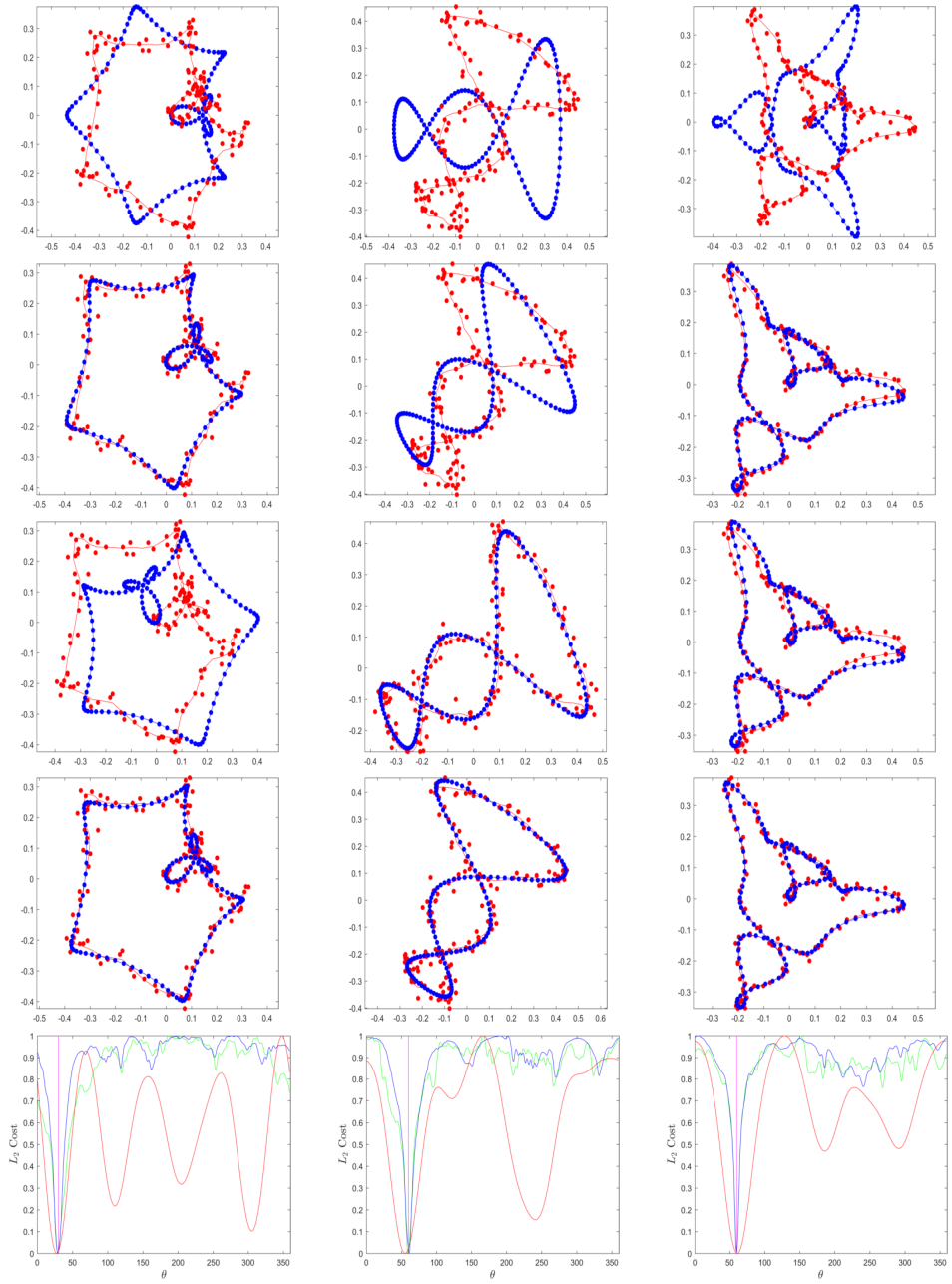


Figure 5.7: Sample registration results for rotation estimation with added noise. Row 1: Model Curve (blue) and Target Curve (red). The target curve is computed by fitting a spline to the noisy data. The red curve does not pass through the noise data points exactly, as it is not an interpolation of the data, but an approximation. Row 2 - 4: Registration result using $\mathcal{C}^x$, $\mathcal{C}^u$ and $\mathcal{C}^{x,u}$ respectively. Row 5 shows the value of each of the cost functions when θ ranges from 1°to 360°. In these graphs red represents $\mathcal{C}^u$, green represents $\mathcal{C}^x$ and blue represents $\mathcal{C}^{x,u}$. The pink line indicates the value of θ_{GT}.

5.5.2 3D Rotation Registration

Next we 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 $\phi(x, \theta) = \mathrm{R}x$. Rather than estimating the rotation matrix R, which has 9 unknowns, we estimate its quaternion representation $q \in \mathbb{S}^3$, as detailed in Section 5.4.4. In this case our latent space is of dimension 4. We compared our results to those obtained using Jian et al's method $\mathcal{C}^x$ [41], CPD [150] and Go-ICP [151]. The shapes used in this experiment were the Stanford Bunny, Dragon and Buddha meshes provided by the Stanford University Computer Graphics Laboratory [2], and the Horse mesh provided by Sumner et al.[152]. Each of these shapes is stored in .ply format with both vertex and edge information available, from which normal vectors are easily calculated, as described in Section 5.4.3. The point clouds of each mesh used, along with the distribution of their normal vectors, can be seen in Figure 5.8. These meshes have between 5000 and 40000 vertices each, and a subsample 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 Appendix B.

Experimental Design

1. Rotation with same sampling for S_1 and S_2: First, taking the vertices $\{x_1^{(i)}\}$ from a given mesh we computed their corresponding normal vectors $\{u_1^{(i)}\}$ using the edge information provided in the .ply, creating the shape S_1. We then transformed the points in S_1, as well as their corresponding normal vectors, by some rotation R to create S_2. Corresponding samples of 1000 points and normal vectors were then drawn from S_1 and S_2 so that point to point correspondences exist between the subsampled target and model shapes. The rotation R was then estimated by registering the sub sampled shape S_1 to the sub sampled shape S_2. Note that the correspondences are not used to enhance the registration process in this case. We tested this registration

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

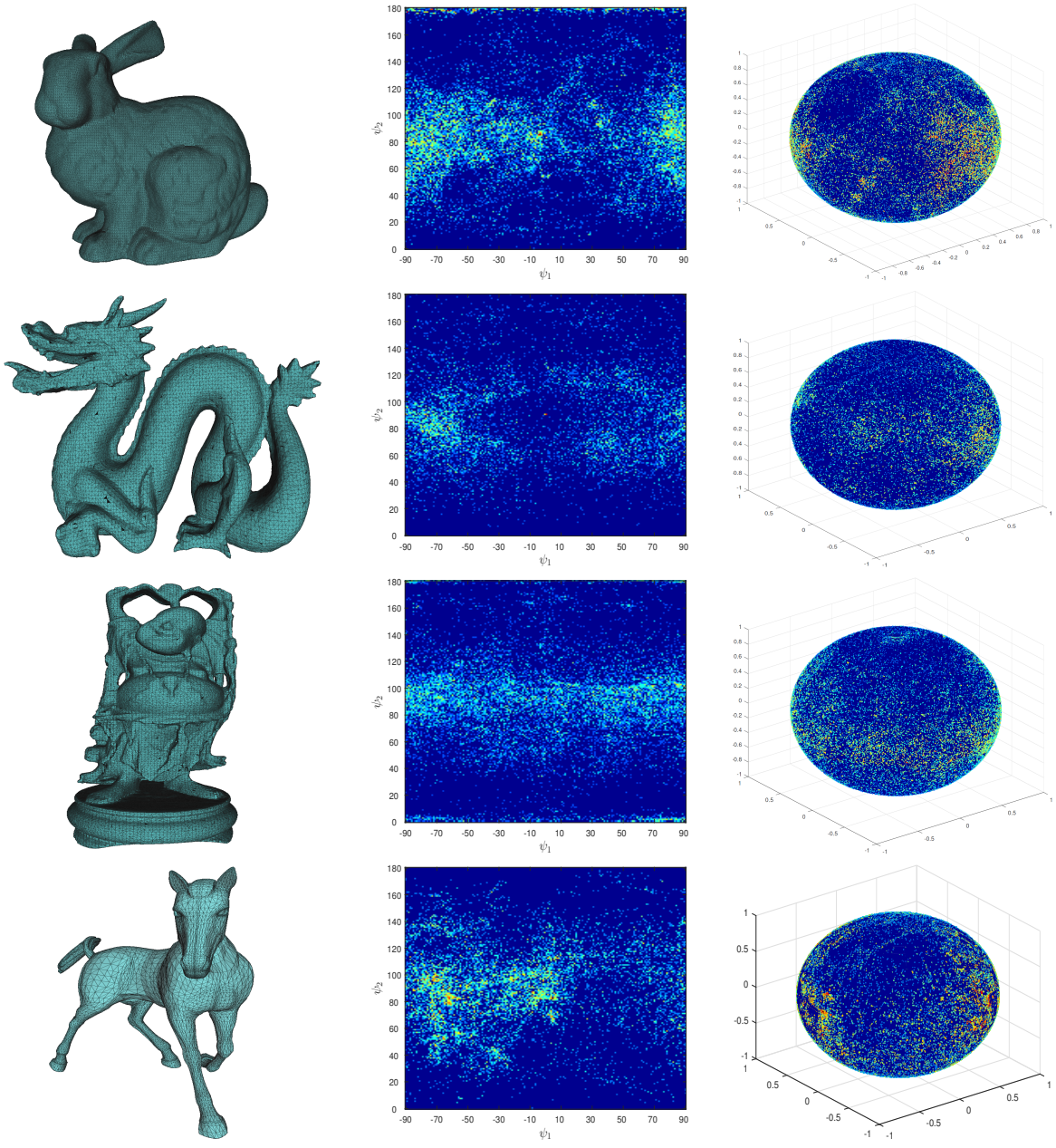


Figure 5.8: Column (a) shows the Bunny, Dragon, Buddha and Horse meshes. Column (b) shows the distribution of normal vectors for each mesh in 2D. The normals can be parametrised as $u(\psi_1, \psi_2) = (\cos(\psi_1)\cos(\psi_2), \sin(\psi_1)\cos(\psi_2), \sin(\psi_2))$. Column (c) shows the distribution of normal vectors $u^{(i)}$ on the sphere.

process for different levels of rotation magnitude $\|\mathrm{R}_\theta\|$, defined as:

$$\|\mathrm{R}\| = \|(\alpha, \beta, \gamma)\|. \tag{5.33}$$

where (α, β, γ) is the Euler angle representation of the rotation R, as described in Section 3.2.1. At each level of rotation magnitude, 15 different pairs of shapes S_1 and S_2 were registered.

2. Rotation with different sampling of S_1 and S_2: Our next experiment followed a similar procedure, however in this case 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. Again, at each level of rotation magnitude, 15 different pairs of shapes S_1 and S_2 were registered.

3. Rotation with added Noise: To assess how our proposed cost functions perform when noise is present in the data, we added three levels of Gaussian noise to each of the points $\{x_2^{(i)}\}$ in the shape S_2, which differs from S_1 by a rotation of magnitude 30°. The positions $\{x_1^{(i)}\}$ in S_1 remain noise free. The noise on S_2 is defined with mean zero and standard deviation varying from 0.001 to 0.003 and Figure 5.9 shows the noisy point positions for the Bunny, Dragon and Buddha shapes at all 3 levels of noise.

When computing the normal vectors of the noisy shape S_2 we used the nearest neighbours approach implemented by Meshlab, as described in Section 5.4.3. 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 the results can be seen in Figure 5.10.

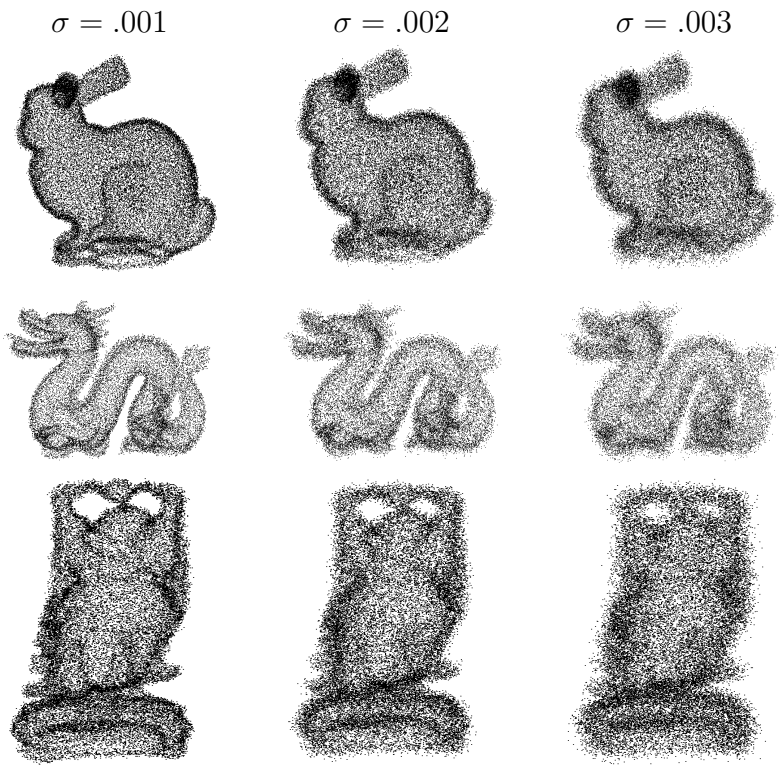


Figure 5.9: Here we show the noisy points $\{x_2^{(i)}\}$ when S_2 corresponds to the Bunny, Dragon and Buddha shapes. Gaussian noise of mean 0 and standard deviation σ is added to each point $x^{(i)}$. Column 1: $\sigma = 0.001$, Column 2: $\sigma = 0.002$, Column 3: $\sigma = 0.003$.

Results

The MSE errors for each experiment are presented in Figure 5.10. The results of the first experiment can be seen in column 1 and show that overall CPD performs best, followed by Go-ICP and $\mathcal{C}_\delta^u$, while $\mathcal{C}_\delta^{x,u}$ and Jian et al'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 $\mathcal{C}_\delta^u$, $\mathcal{C}_\delta^x$ and $\mathcal{C}_\delta^{x,u}$ continue to estimate good solutions.

Column 2 of Figure 5.10 presents the results when different sampling of the target and model shape is used. We can see that $\mathcal{C}_\delta^{x,u}$ performed the best in this case, followed

by Jian et al'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.

For our final experiment with added noise, the results in column 3 of Figure 5.10 show that 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.

5.5.3 2D Non-rigid Registration

In this section we again consider two 2D shapes $S_1 = \{(x_1^{(i)}, u_1^{(i)})\}_{i=1,..n}$ and $S_2 = \{(x_2^{(j)}, u_2^{(j)})\}_{j=1,..n}$, which are represented by their point locations $\{x^{(i)}\} \in \mathbb{R}^2$ and normal vectors $\{u^{(i)}\} \in \mathbb{S}$. In this case S_1 and S_2 differ by a non-rigid deformation which we estimate using the transformation ϕ, defined as a TPS transformation as given in Equation 3.16. This transformation is controlled by $N = 12$ control points and our latent space is of dimension $(12 \times 2) + 6 = 30$.

We assess the estimation of ϕ using the cost functions $\mathcal{C}^x$ [41] and $\mathcal{C}^{x,u}$ but omit $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ as we found that normal information alone is not sufficient when estimating a non-rigid transformation. We also omit $\mathcal{C}^{x,u}_\delta$ as it generates similar results to $\mathcal{C}^{x,u}$. Again, when using $\mathcal{C}^{x,u}$, which depends on the von Mises-Fisher normalising constant $C_d(\kappa)$, we artificially define u on $\mathbb{S}^2$ instead of $\mathbb{S}$ by adding a third dimensional coordinate to the normal vector which is set to zero. This ensures that the normalising constant $C_3(\kappa)$ is easy to compute.

As in Section 5.5.1 we register parametric curves sampled at several locations, from which the normal vectors $\{u^{(i)}\}$ can be computed. As well as comparing $\mathcal{C}^x$ [41] and $\mathcal{C}^{x,u}$, we also compare to other state of the art non-rigid registration techniques: CPD [150] and GLMD [33]. For comparison we implement a similar experimental framework as that presented by Yang et al.[33] and for this reason we also normalise all curves so that they lie within $[0, 1] \times [0, 1]$.

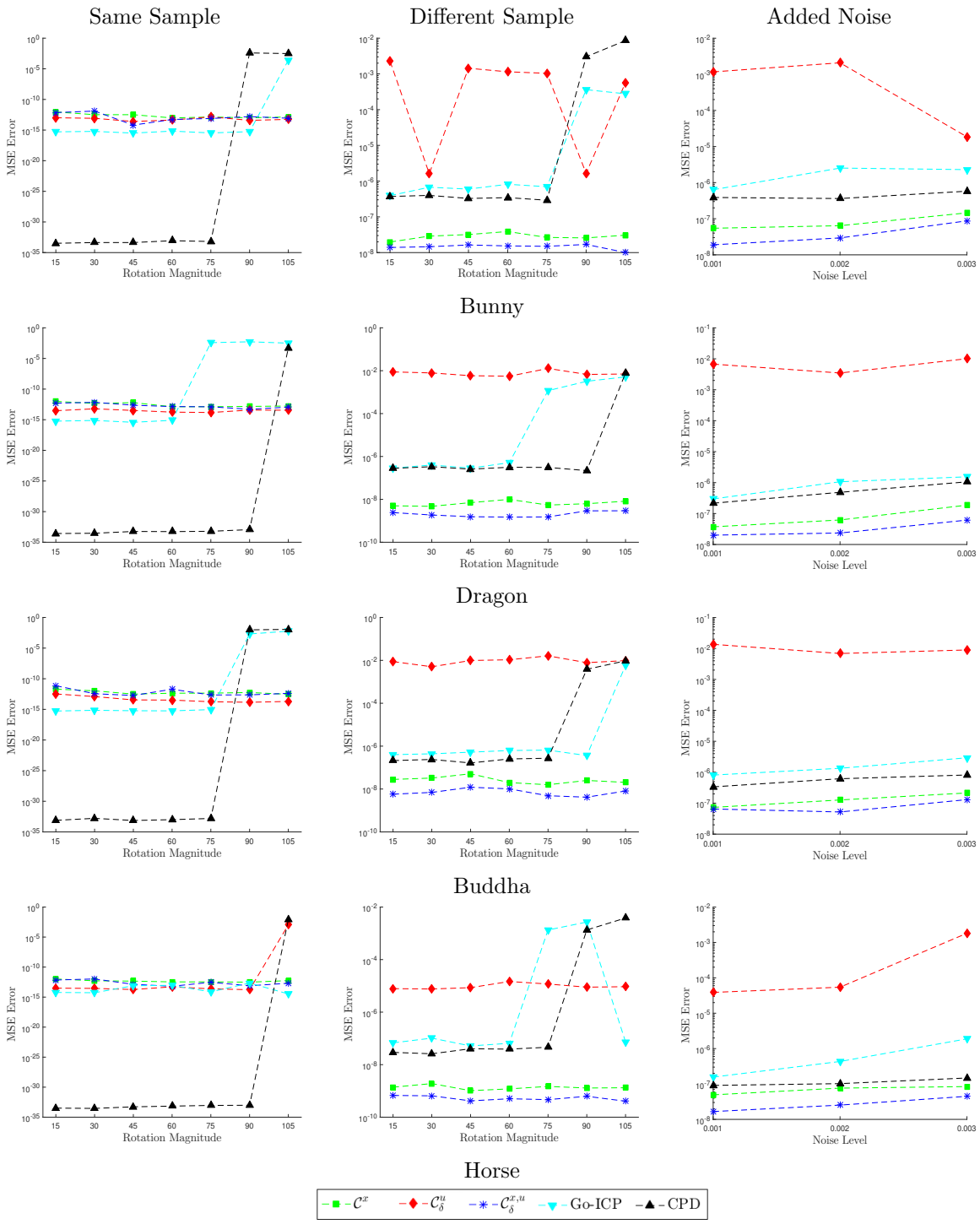
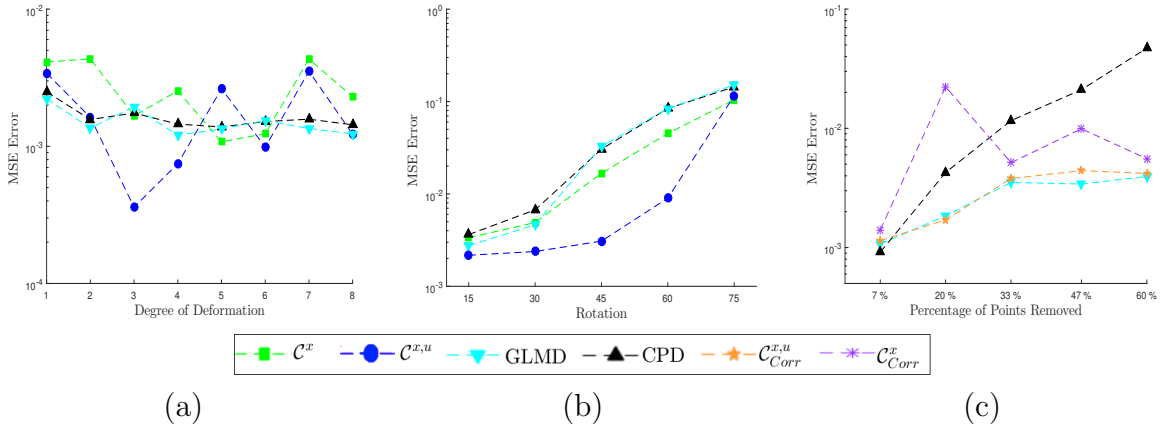


Figure 5.10: Error results obtained when registering shapes S_1 and S_2 with the same sampling (column 1), different sampling (column 2) and added noise (column 3), for the Bunny (row 1), Dragon (row 2), Buddha (row 3) and Horse (row 4) meshes.



(a) (b) (c)

Figure 5.11: Mean Square Error 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

Experimental Design

1. Deformation: Taking a parametric curve, we sampled it at 50 locations $\{x_2^{(i)}\}$ and computed the corresponding normal vectors $\{u_2^{(i)}\}$, creating the shape S_2. Then, using the same technique proposed by Yang et al.[33] in their experimental section, we deformed S_2 using a non-rigid transformation to create the shape S_1. To deform S_2 we took 8 control points on the boundary of the curve, as in Figure 5.12, and moved a number of them in a particular direction by a displacement of 0.2. Each control point is allowed to move in one of four directions and the direction is randomly determined. The number of control points that are displaced determines the degree of deformation and can vary from 1 to 8. The TPS transformation which maps the original control points to their new displaced locations is computed and applied to the points $\{x_2^{(i)}\}$ to create $\{x_1^{(i)}\}$. The normals vectors $\{u_2^{(i)}\}$ are then computed, creating the shape S_2. For each level of deformation we register 120 pairs of shapes S_1 and S_2 and present the average MSE results in Figure 5.11(a).

2. Deformation and Rotation: For our second experiment we used the method described in the previous section to deform S_2, creating the shape S_1. In this case we also rotated S_1 so that S_1 and S_2 differ by both a non-rigid deformation and a rotation.

Setting the degree of deformation to 4 in all cases, we tested all methods for rotations of $\pm 15°, \pm 30°, \pm 45°, \pm 60°$ and $\pm 75°$. At each rotation value we registered 240 pairs of deformed curves for each method and the mean square errors computed can be seen in Figure 5.11 (b).

3. Deformation with missing data: In our third experiment we sampled a parametric curve at 150 locations to create S_2, and deformed it by a deformation of degree 4 to create S_1. There was no rotation added in this case. Instead we removed a percentage of points in order from the end of the deformed curve S_1 so that it is missing data.

For this experiment correspondences were estimated using Yang et al's technique, as described in Section 5.4.5, and were used when optimising both $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x}$. We use $\mathcal{C}^{x,u}_{corr}$ and $\mathcal{C}^{x}_{corr}$ to denote the use of correspondences with these cost functions. We tested all methods with several levels of missing data $(7\%, 20\%, 33\%, 47\%, 60\%)$. At each level we registered 120 pairs of curves and the average MSE results computed can be seen in Figure 5.11 (c).

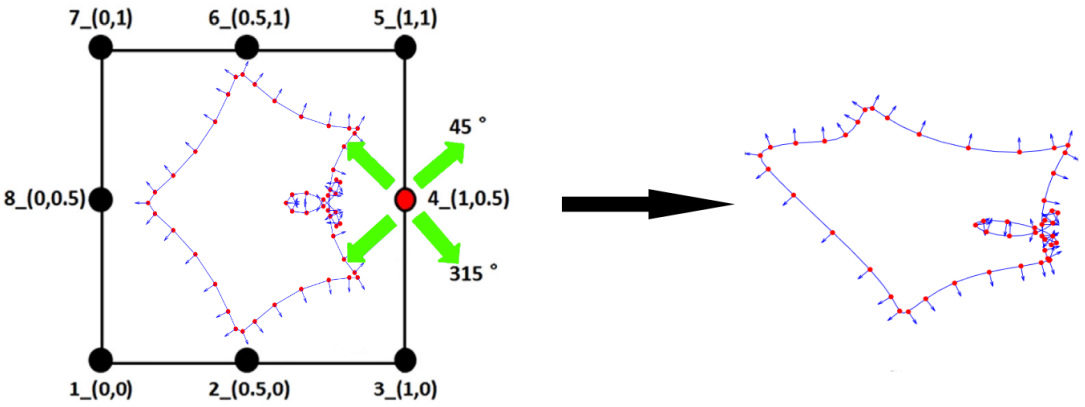


Figure 5.12: On the left we show the 8 control points chosen on the boundary of the shape S_2 which can move in any four directions (shown in green). The deformed control points are used to estimate a transformation which deforms S_2, creating S_1 (right).

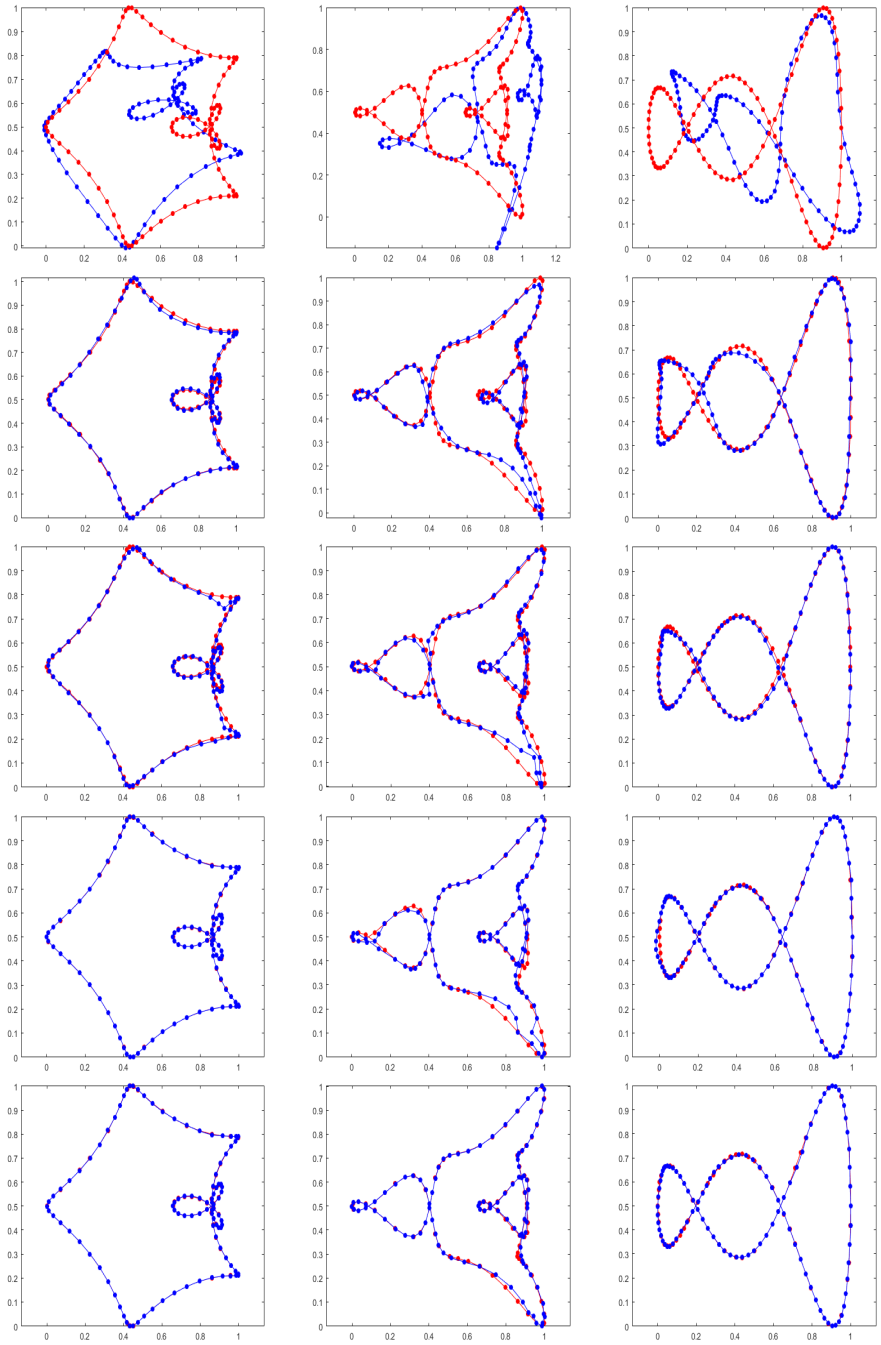


Figure 5.13: Sample registration results for deformation estimation. Row 1: Model Curve (blue) and Target Curve (red). Row 2 - 5: Registration result using CPD, GLMD, $\mathcal{C}^x$ and $\mathcal{C}^{x,u}$ respectively. Column 1: Deformation degree = 5; Column 2: Deformation degree = 6; Column 3: Deformation degree = 8;

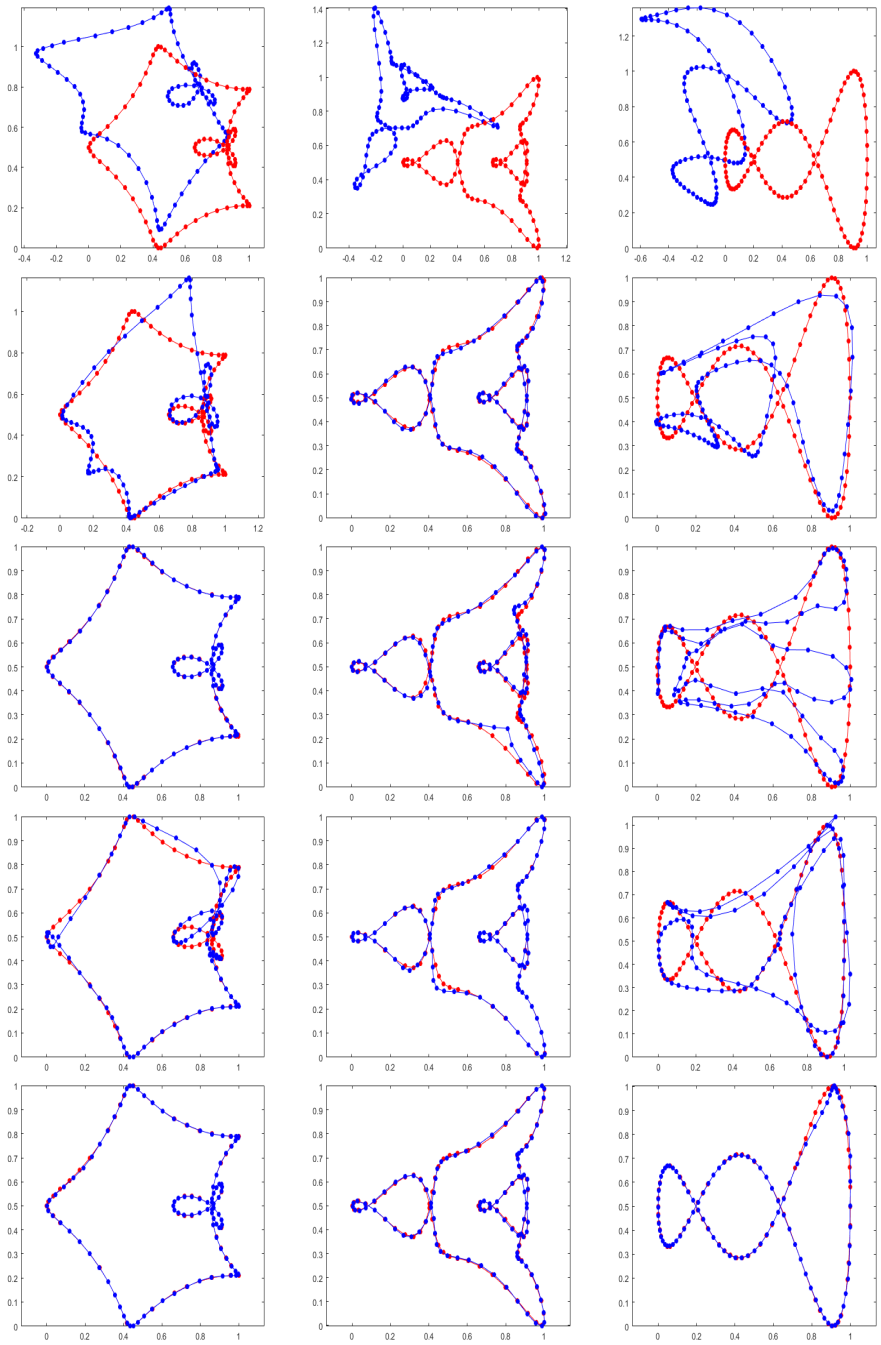


Figure 5.14: Sample registration results for rotation and deformation estimation. Row 1: Model Curve (blue) and Target Curve (red). Row 2 - 5: Registration result using CPD, GLMD, $\mathcal{C}^x$ and $\mathcal{C}^{x,u}$ respectively. Column 1: Rotation = 30 °; Column 2: Rotation = 45 °; Column 3: Rotation = 60 °.

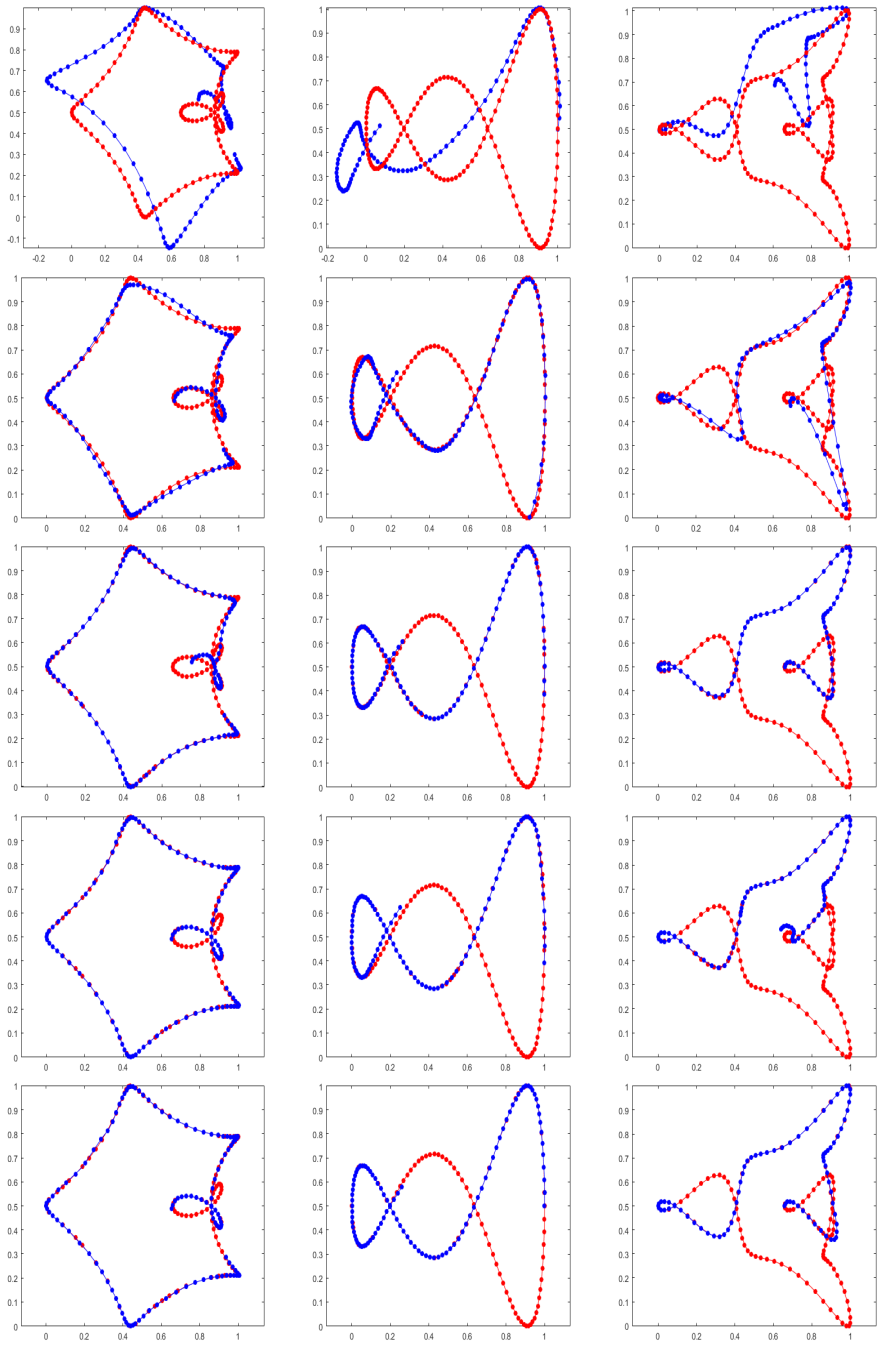


Figure 5.15: Sample registration results for deformation estimation with missing data. Row 1: Model Curve (blue) and Target Curve (red). Row 2 - 5: Registration result using CPD, GLMD, $\mathcal{C}^x$ and $\mathcal{C}^{x,u}$ respectively. Column 1: Missing pts = 30; Column 2:Missing pts = 50; Column 3:Missing pts = 70.

Results

For our first experiment the MSE results for each cost function can be seen in Figure 5.11 (a), and Figure 5.13 gives a sample of some of the registration results. We found that in general $\mathcal{C}^{x,u}$ performs well, however at times it fails to estimate a good result, creating spikes in the average MSE at deformations of degree 5 and 7. Similar spikes appear in the results for $\mathcal{C}^{x}$. Both CPD and GLMD seem to generate consistent results over all deformations.

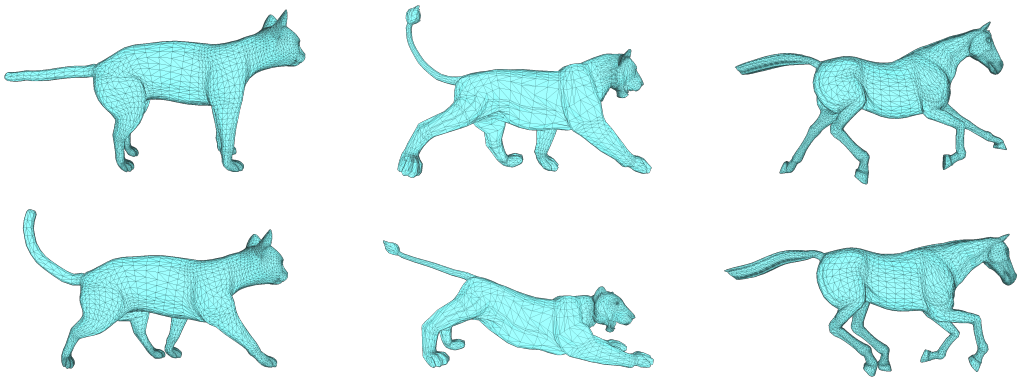
For our second experiment the MSE results can be seen in Figure 5.11 (b) and some sample registration results can also be seen in Figure 5.14. These results show that $\mathcal{C}^{x,u}$ performed best, followed by $\mathcal{C}^{x}$, GLMD and CPD. The addition of the normal information in the cost function 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.

The results of our third experiment can be seen in Figure 5.11(c) and 5.15. In this case we found that $\mathcal{C}^{x,u}_{corr}$ performed as well as GLMD, followed by $\mathcal{C}^{x}_{corr}$ and CPD. Without correspondences we found that both $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x}$ tried to maximise the amount of overlap between the curves and rarely estimated the correct parameters.

5.5.4 3D Non-rigid Registration

Finally we 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 $\{u^{(i)}\} \in \mathbb{S}^2$ and differ by a non-rigid deformation. We register these shapes by estimating a non-rigid TPS transformation, as defined in Equation 3.16. We choose the number of control points as $N = 125$ so our latent space has $(125 \times 3) + 12 = 387$ dimensions. In this section we will incorporate point correspondences into the cost functions $\mathcal{C}^{x}$ and $\mathcal{C}^{x,u}$, notating them as $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$. Again we omit $\mathcal{C}^{u}$ and $\mathcal{C}^{u}_{\delta}$ as we found that 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 $\mathcal{C}^{x,u}$.

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 [150] and GLMD [33] when registering shapes with unknown correspondences that must be estimated.



(a) Cat (b) Lion (c) Horse

Figure 5.16: The meshes used to generate the shapes S_1 and S_2 in our non-rigid registration experiment on 3D data. Meshes representing the same animal (eg. cat, lion, horse) have the same number of vertices and exact vertex correpondences. The cat, lion and horse meshes have 7202, 5000 and 8431 vertices respectively.

Experimental Design

1. Deformation with ground truth correspondences: In this experiment we use the dataset of shapes provided by Sumner et al. [152] which contains 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. Some examples of these meshes can be seen in Figure 5.16. 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.

2. Deformation with estimated correspondences: In this experiment we use a scan taken of the Stanford Bunny with 1000 points, shown in Figure 5.19, 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 discussed in Section 5.4.3. Then using the same deformation technique proposed by Yang et al.[33] and implemented in Section 5.4.3, we used 9 control points on the boundary of the points $\{x_2^{(i)}\}$ to deform them,

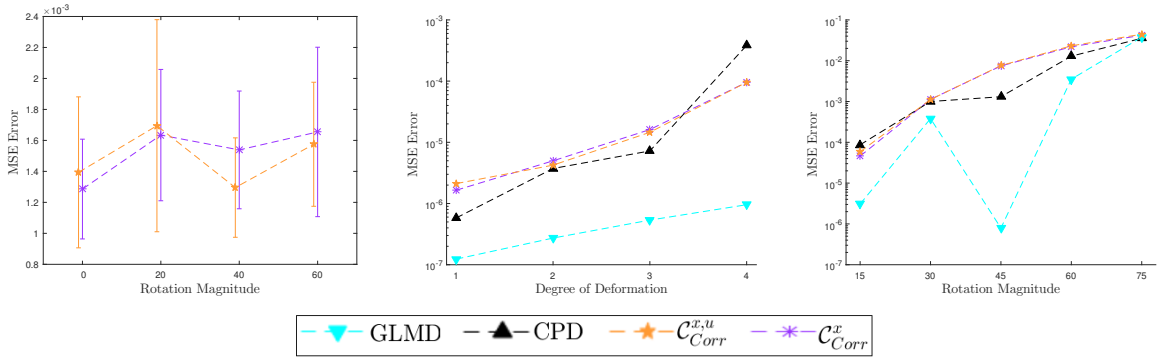


Figure 5.17: Mean Square Error results for non-rigid registration with 3D data. (a) Comparison between $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ when registering meshes with exact correspondences. The meshes differ by a deformation and rotation varying from 0° to 60°. The standard error bars are included and emphasise the similarity between the cost functions; (b) Non-rigid 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°.

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.[33] and detailed in Section 5.4.5. 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 $(15°, 30°, 45°, 60°, 75°)$, with 15 pairs of shapes registered for each rotation.

Results

The results of our first experiment, comparing $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ when exact point correspondences are known, can be seen in Figure 5.17(a) and Figure 5.18. In this experiment we found that 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 we set a limit of 50,000 on

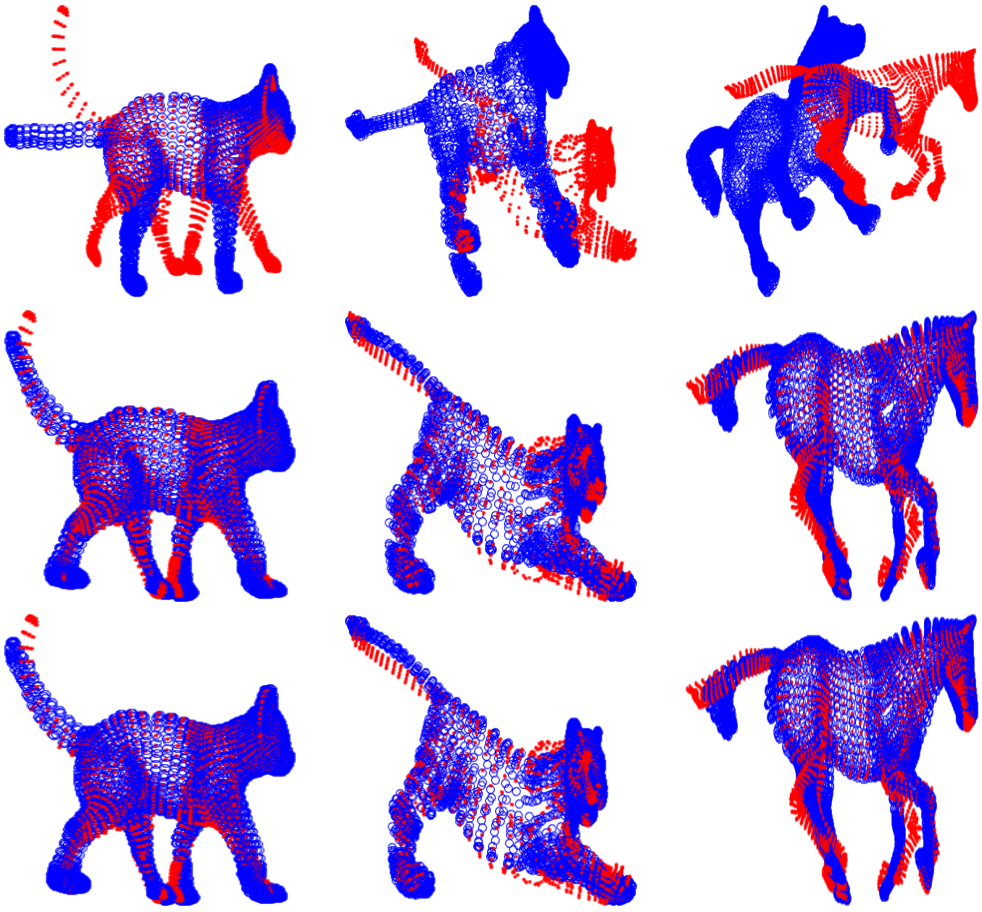
the number of function evaluations computed during optimisation (at each simulated annealing step). In Figure 5.17(a), the MSE results show that there is very little difference between the cost functions and in Figure 5.18 we can see that even with 50,000 functions evaluations, both $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ failed to converge to a good solution and do not match the model shape S_1 to the target shape S_2 exactly.

The MSE results from our second set of experiments, which registered two Bunny shapes, can be seen in Figures 5.17(b), 5.17(c) and 5.19. From Figure 5.17(b) we can see that for all degrees of deformation applied to the model shape S_1, GLMD performs the best. Again we can see that $\mathcal{C}^{x}_{corr}$ 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 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.

The results of registering Bunny shapes differed by both a non rigid deformation and rotation can be seen in Figure 5.17(c) and row 3 of 5.19. 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 $\mathcal{C}^{x,u}_{corr}$.

5.5.5 Computation Complexity

Due to the double sum in all of the cost functions, the computational complexity of $\mathcal{C}^{x}$, $\mathcal{C}^{u}$, $\mathcal{C}^{u}_{\delta}$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_{\delta}$ depends on the number of points n_1 and n_2 in the shapes S_1 and S_2. When no point correspondences are chosen the computational complexity is of order $\mathcal{O}(n_1 \times n_2)$. Choosing to use n point correspondences instead reduces this to $\mathcal{O}(n)$. The computation time needed by the gradient ascent technique to converge to a good solution also depends on the dimension of the latent space. The latent space dimension is determined by the transformation being estimated and the dimension d



(a)Rotation 0° (b)Rotation 20° (c) Rotation 40°

Figure 5.18: Some of the registration results for $\mathcal{C}^x_{Corr}$ and $\mathcal{C}^{x,u}_{Corr}$ for shapes with exact point correspondences. Row 1: The model shape S_1 (blue) and target shape S_2 (red); Row 2: Target shape (red) and $\mathcal{C}^x_{Corr}$ registration results (blue); Row 3: Target shape (red) and $\mathcal{C}^{x,u}_{Corr}$ registration results (blue).

of the space $\mathbb{R}^d$ in which the shapes are defined.

We have not provided analytical gradients to the gradient ascent algorithm for any of the cost functions $\mathcal{C}^x$, $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. Instead, the gradient ascent algorithm uses numerical methods at each iteration to approximate the gradient. This adds an additional cost to the cost functions, requiring a larger number of function evaluations at each iteration. An analytical gradient could be computed in the case of $\mathcal{C}^x$ to speed up computation time, as was the case in the code provided by Jian et al. For the

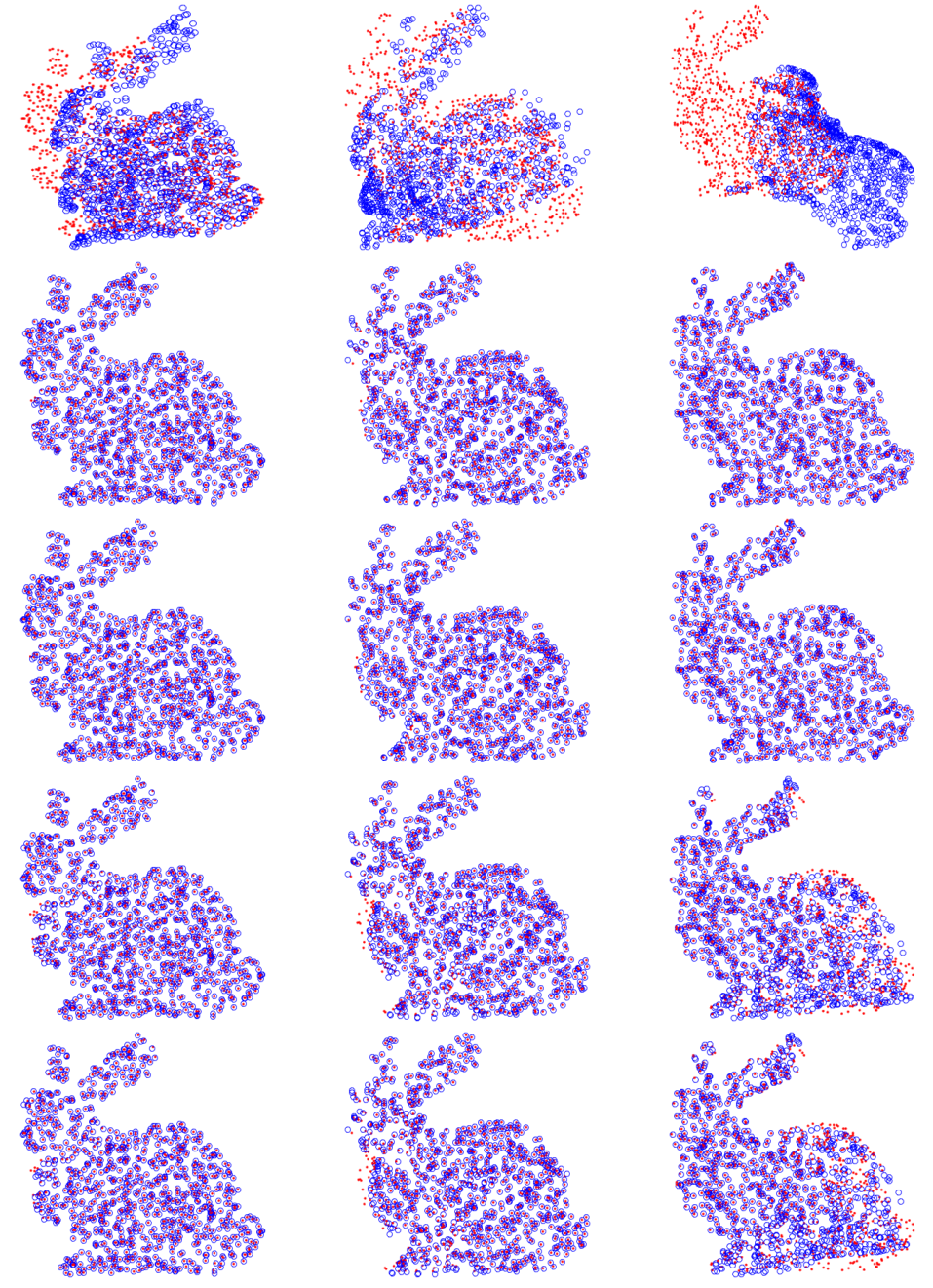


Figure 5.19: The results of several registration methods applied to two 3D shapes differing by a non-rigid deformation and a rotation. Row 1 shows the model shape S_1 (blue) and target shape S_2 (red). Rows (2 - 5) show the transformed model after registration (blue) computed using CPD (Row 2), GLMD (Row 3), $\mathcal{C}^{x}_{corr}$ (Row 4) and $\mathcal{C}^{x,u}_{corr}$ (Row 5).

	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	.4695
2D TPS	12	30	100	100	2.2s	✗	✗	2.9s	✗
3D TPS	125	387	100	100	16s	✗	✗	30s	✗

Table 5.2: The time taken by each of the cost functions to compute 100 iterations of the gradient ascent algorithm. In each case the shapes S_1 and S_2 have 100 points each. The number of control points used by the TPS functions is shown in column 2 and column 3 gives the dimension of the latent space in each case.

cost functions $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$, the analytical derivative could be supplied when estimating a rotation transformation. However, when estimating a TPS transformation using either $\mathcal{C}^{x,u}$ or $\mathcal{C}^{x,u}_\delta$, a simple analytical derivative of the cost functions is not available. Although the normal vectors of a shape change when a TPS transformation is applied to it, the computation of the normal vectors cannot be computed directly using only the TPS transformation. Instead, the normal vectors of the transformed shape need to be recomputed using either the neighbourhood structure of the points $\{x^{(i)}\}$ or the vertex and edge information, if available. Therefore computing the analytical derivative of the cost functions with respect to the TPS transformation can be very complex, and using a numerical approximation to the gradient is preferable.

In Table 5.2 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 5.3 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 5.3. In Table 5.4 we also present the computation times needed by the CPD, Go ICP and GLMD algorithms to register shapes S_1 and S_2.

5.5.6 Summary

In this section we have presented several experiments exploring the performance of each of the proposed cost functions when registering 2D and 3D shapes differing by a rotation or non-rigid transformation. We found that in the case of 2D and 3D rotation

	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 5.3: The number of iterations typically taken by each algorithm to register two point clouds with 100 points each. These figures are computed using our full simulated annealing strategy, with the number of simulated annealing steps used given in column 5. *Note that due to the high dimension of the latent space, we limited the number of function evaluations in this case, thus limiting the number of iterations allowed. Although a good solution was reached after this many iterations, the cost functions still had not fully converged.

	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

Table 5.4: The time taken, on average, by the Go ICP, CPD and GLMD methods to converge to the correct solution. For CPD, the MSE tolerance chosen for 3D rotation, 2D TPS and 3D TPD registration was the same as that used in the demo code provided by authors. It was set to e^{-8} for 3D rotation, e^{-8} for 2D TPS and e^{-3} for 3D TPS, hence the difference in computation times.

estimation, the cost function $\mathcal{C}^{x,u}$, which takes into account both point positions and directional information, performs best overall in terms of accuracy, outperforming Jian et al's cost function $\mathcal{C}^x$ as well as the CPD and Go ICP methods. For 2D shapes differing by a non-rigid transformation we found that while all techniques perform similarly when two shapes differ by only a non-rigid deformation, when they differ by a rotation and non-rigid deformation, $\mathcal{C}^{x,u}$ outperforms $\mathcal{C}^x$ as well as CPD and GLMD. When partial curves are registered and correspondences are used, $\mathcal{C}^{x,u}_{corr}$ also outperforms CPD and $\mathcal{C}^x_{corr}$, giving similar results to GLMD.

However, in the case of 3D shapes differing by a non-rigid deformation we found that the high dimensional latent space and the small number of control points used reduced the accuracy of $\mathcal{C}^{x,u}_{corr}$ and $\mathcal{C}^x_{corr}$. As correspondences were incorporated into both cost functions to reduce computational cost, the accuracy of the results also depended on

the quality of the correspondences estimated. The need to compute derivatives and normals vectors at each iteration when using $\mathcal{C}_{corr}^{x,u}$ also increased the computational cost of the algorithm.

5.6 Conclusion

In this chapter we have proposed to include both positional and directional information when modelling shapes for shape registration. We used the von Mises-Fisher distribution to model the unit normal vectors of a shape and propose KDEs using a combination of Dirac, Gaussian and von Mises-Fisher kernels. We show that computing the $\mathcal{L}_2$ distance between two KDEs with von Mises-Fisher kernels has an explicit expression when $d = 3$, while using Dirac kernels ensures that the $\mathcal{L}_2$ distance has an explicit expression when estimating a rotation for any dimension d.

We also present experimental results which validate that using both point positions and directional information can enhance the accuracy of the registration results, especially when the shapes differ by a rotation transformation. However, the gradient ascent technique used to optimise the cost functions is very time consuming when the latent space that needs to be explored has a high dimension. Implementing a new optimisation technique which is less time consuming and could explore the latent space quickly would ensure that this type of cost functions could be used when the dimension of the space is high. 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 before estimating the non-rigid transformation. In the next chapter we will explore the application of optimal transport techniques to the shape registration problem to see if these methods prove advantageous in this area.

Chapter 6

Optimal Transport for Shape Registration

In Chapter 4 we proposed a colour transfer technique which was based on minimising a divergence between PDFs, a strategy often used in shape registration. In this chapter we explore the application of optimal transport, a popular approach in colour transfer, to 3D shape registration. We present a brief exploration of how two optimal transport based colour transfer techniques perform when applied directly to shape data. The objective of this chapter is to draw attention to some of the initial problems that arise when taking this approach, and highlight that some additional constraints need to be introduced to make such an approach successful for shape registration.

6.1 Introduction

We will investigate the application of two optimal transport methods - the Iterative Distribution Transfer (IDT) and Sliced Wasserstein Distance (SWD) methods [6, 77] - to 3D shape registration. While complex in higher dimensional spaces, optimal transportation is straightforward in one dimensional space. Both the IDT and SWD approaches take advantage of this and estimate a non-parametric transformation $\hat{\phi}$ by reducing the problem to several 1-dimensional optimal transportation problems. In the previous chapter we found that the optimisation of the $\mathcal{L}_2$ distance was quite computationally expensive when a non-rigid deformation was estimated in 3-dimensional space.

In this chapter we investigate whether projecting the 3D shape registration problem to lower dimensions could reduce optimisation cost while still allowing an appropriate transformation $\hat{\phi}$ to be estimated. In the next section we will present results illustrating the application of the IDT and SWD methods to 3D shape registration.

6.2 Illustrations

To test both approaches we have taken the implementations provided by Pitié et al.[1] and Bonneel et al.[2] for colour transfer and applied them directly to 3D shape data. We apply both techniques to shapes differing by a rigid rotation and non-rigid deformation (Section 6.2.1) and investigate the transformation $\hat{\phi}$ that is estimated (Sections 6.2.2 and 6.2.3).

6.2.1 Registration

First we apply both methods to the problem of registering 3D shapes differing by a rotation and present the results in Figure 6.2. In this case we found that the IDT method transformed the model to the target point clouds almost exactly, while the results generated by the SWD method contained some errors. Next we used both methods to register point clouds differing by a non-rigid deformation and present the results in Figure 6.3. In all cases the number of points in the model and target point clouds is the same. Again, we found that the IDT method transformed the model to the target point clouds almost exactly while the SWD results contained some errors. The parameters used for both algorithms to generate these results are presented in Table 6.1. In Figure 6.1 we present several steps in the estimation of the transformation $\hat{\phi}$ using the IDT algorithm, and show how the algorithm converges to the registration solution. We can see that although the model point set matches the target point set well after many iterations, in the early stages of the registration process the structure of the model cat shape is lost.

[1] `https://github.com/frcs/colour-transfer`
[2] `https://github.com/gpeyre/2014-JMIV-SlicedTransport`

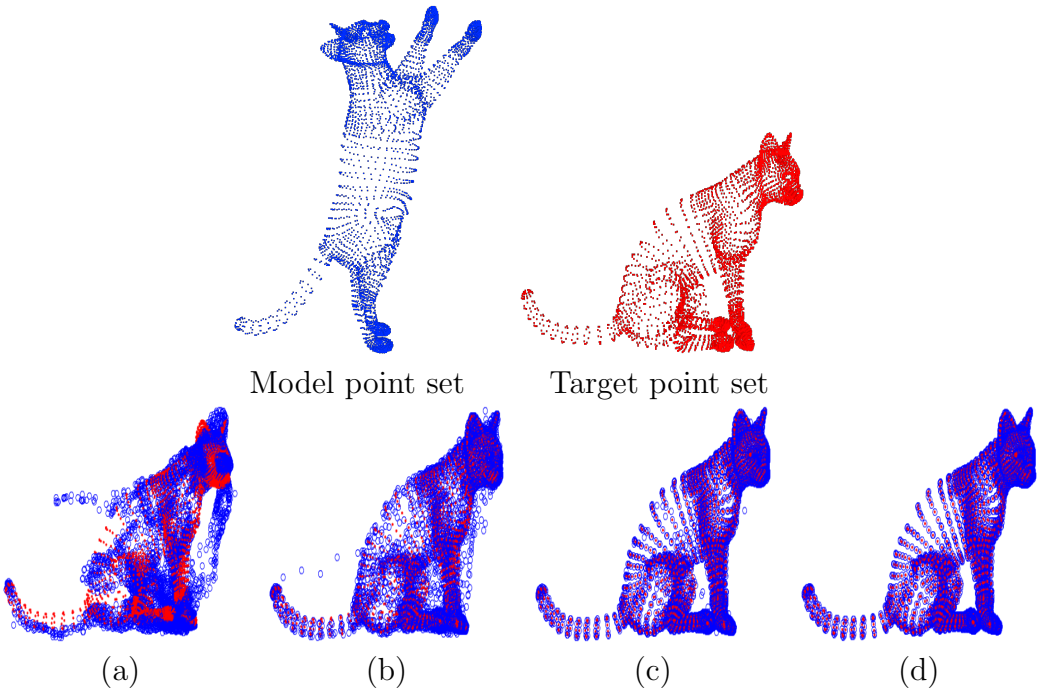


Figure 6.1: The transformation $\hat{\phi}$ that is estimated by the IDT algorithm when registering the model point set (blue) to the target point set (red). Here we show the estimated transformation $\hat{\phi}$ at different stages of convergence: after (a) 10, (b) 100, (c) 1000 and (d) 5000 iterations. We can see that at the early stages of registration, the structure of the cat shape is lost.

	Iterations	Projections per iteration
IDT	1000	6
SWD	500	1000

Table 6.1: The parameters used for the IDT and SWD methods when generating the results presented in this chapter.

6.2.2 Correspondences created by $\hat{\phi}$

In this section we investigate the correspondences $(x, \hat{\phi}(x))$ between the model and target point sets that were created by the IDT and SWD methods when the transformation $\hat{\phi}$ is estimated. Here the point x lies in the model point set and the point $\hat{\phi}(x)$ lies in the transformed model point set, which is equivalent to the target point set.

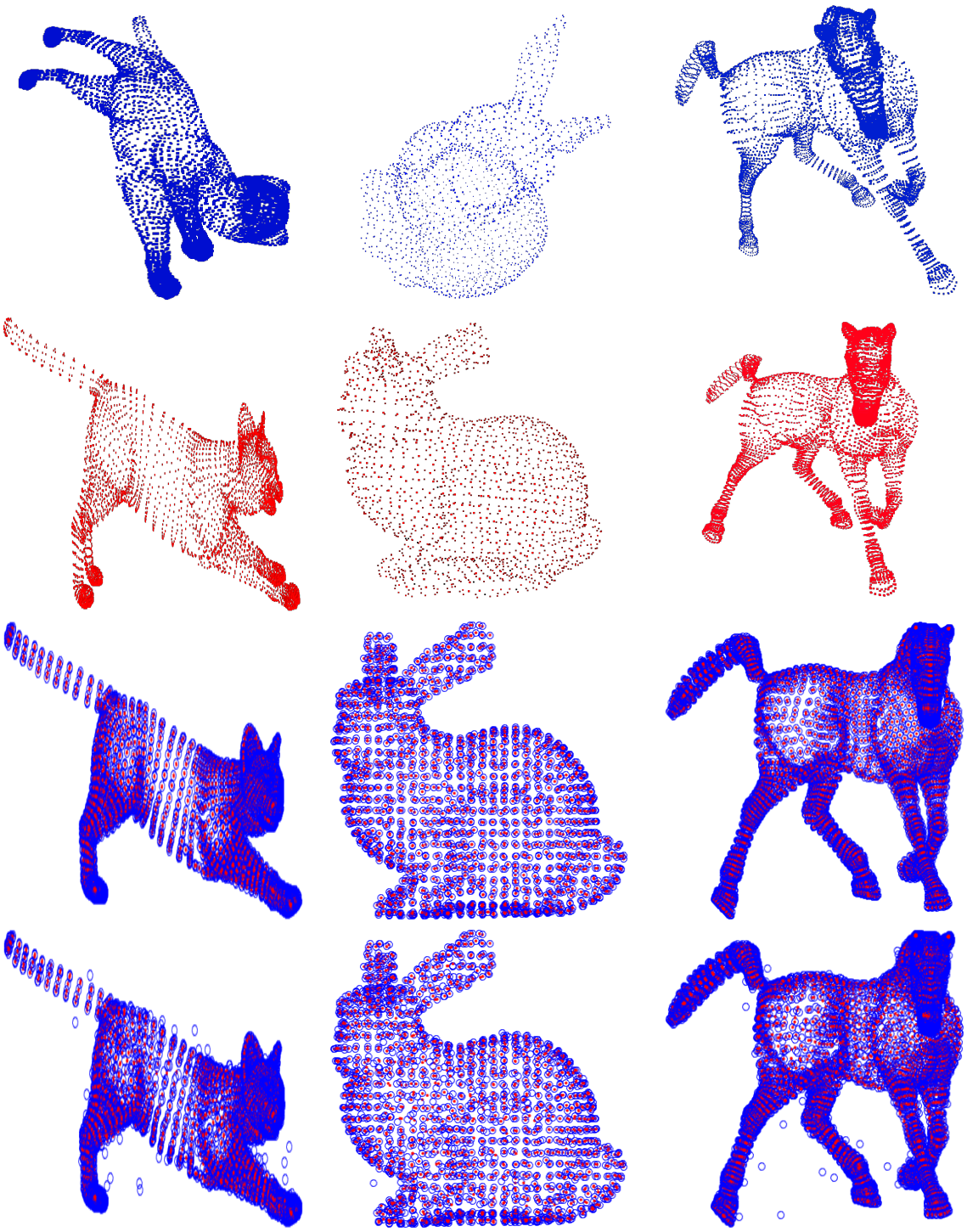


Figure 6.2: Registration results for point clouds differing by a rotation. Row 1: Model point clouds; Row 2: Target point clouds; Row 3: Results of IDT algorithm; Row 4: Results of SWD algorithm.

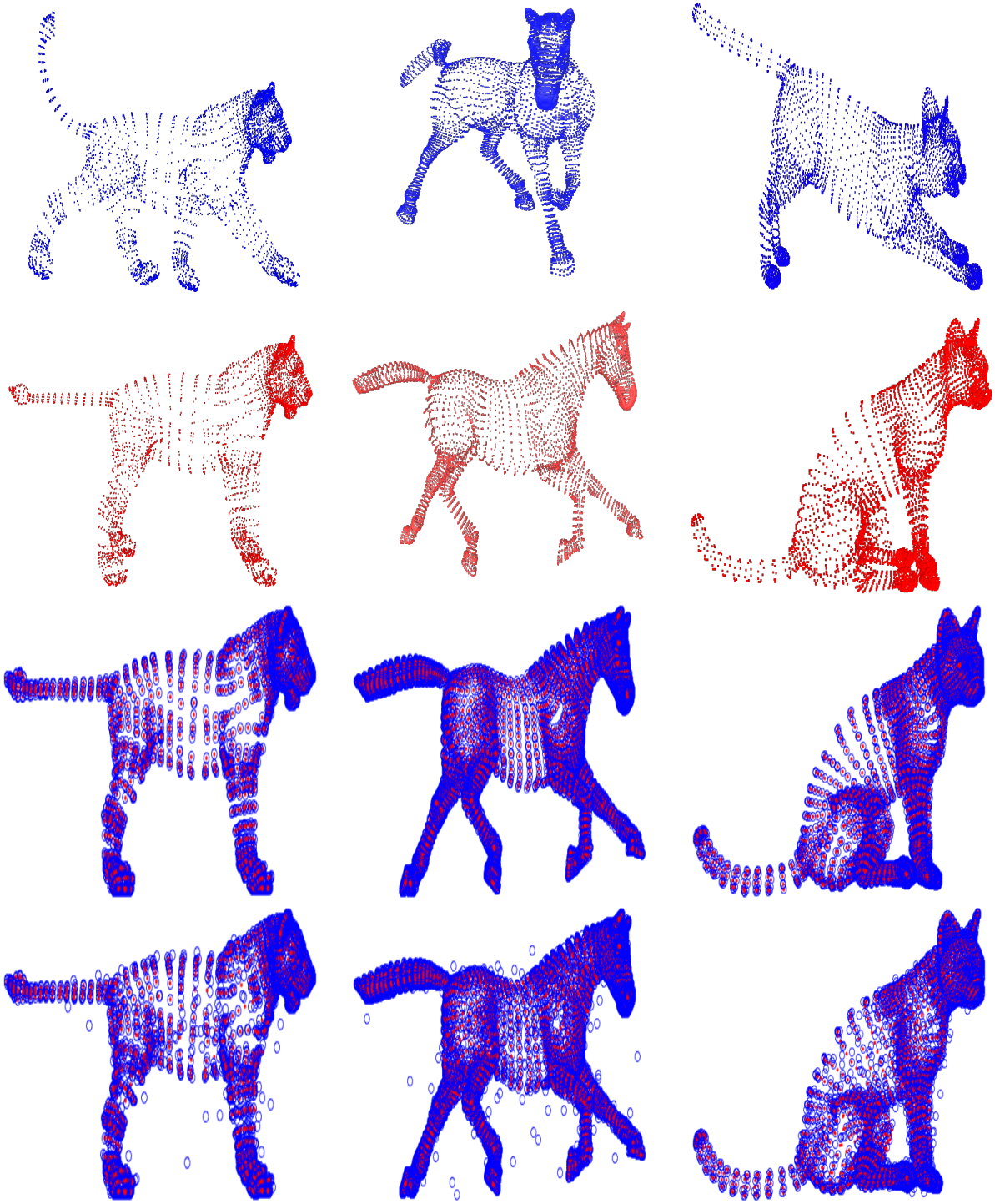


Figure 6.3: Registration results for point clouds differing by a non-rigid deformation. Row 1: Model point clouds; Row 2: Target point clouds; Row 3: Results of IDT algorithm; Row 4: Results of SWD algorithm.

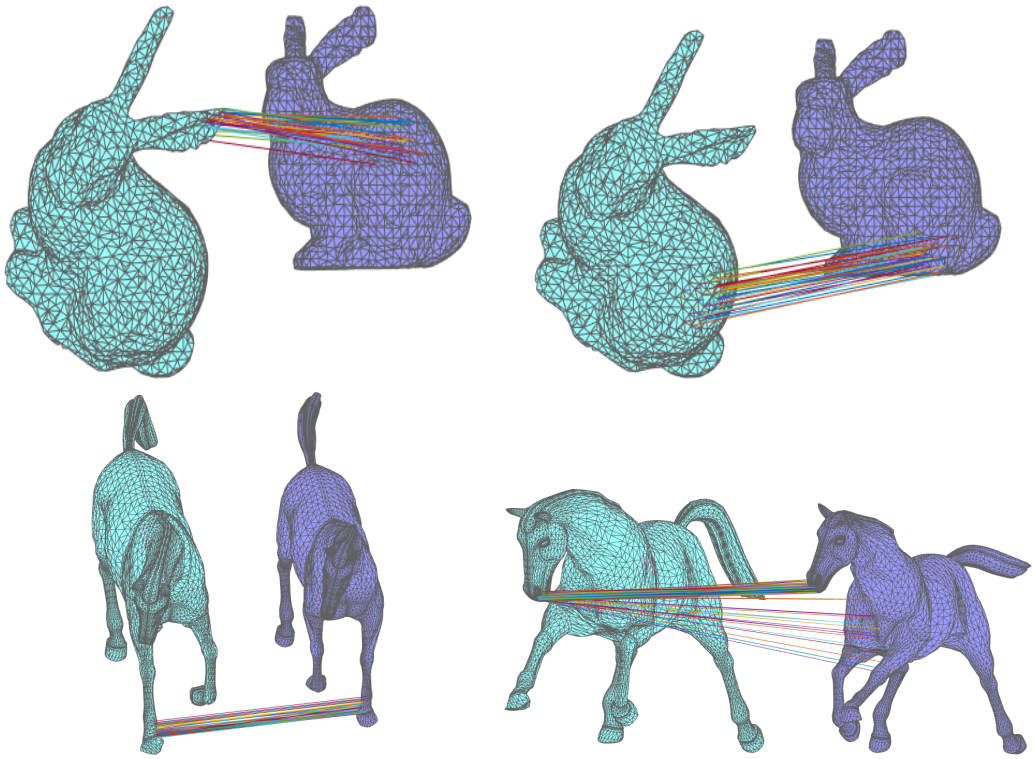


Figure 6.4: The correspondences $(x, \hat{\phi}(x))$ created by the transformation $\hat{\phi}$ using the IDT algorithm. Row 1: Two subsets of correspondences estimated between the model (green) and target (purple) bunny point clouds differing by a rotation; Row 2: Two subsets of correspondences estimated between the model and target horse point clouds differing by a non rigid deformation. The same pair of horse shapes are shown on both the right and left sides, but from different viewpoints for better illustration of the point correspondences.

In Figure 6.4 we present a sample of point correspondences estimated using the IDT algorithm, although we found similar results when we applied the SWD method. In these figures we plot the model and target shapes as meshes rather than point clouds for clarity. Note that the optimal transport framework is not able to maintain useful information regarding the edges in the meshes, therefore the connectivity shown in Figure 6.4 corresponds to the original edges of the target mesh.

In the first row of Figure 6.4 we present the correspondences estimated when the model bunny (green) is registered to the target bunny (purple), which differs from the model by a 3D rotation. Here we can see that the correspondences estimated by the

transformation $\hat{\phi}$ do not match similar parts of the bunny point sets. Points on the ear of the model bunny have been mapped to points on the back of the target bunny (in the left image), and points around the back of the model bunny have been mapped to points along the bottom of the target bunny (in the right image). These correspondences have been estimated as the points lie close together in some 1D projection space in which $\hat{\phi}$ was estimated.

In the second row of Figure 6.4 we show the correspondences created by the IDT algorithm when the model horse (green) is transformed to the target horse (purple), which differs from the model by a non-rigid deformation. Here we can see that points on the front left leg of the model horse have been mapped to points on the front right leg of the target horse. Similarly, points around the mouth of the model horse have been mapped to points on the body of the target horse. Again, these points have become correspondences as they lie close together in some 1-dimensional space used to estimate $\hat{\phi}$.

6.2.3 Effect of Axis Sequence

The transformation function $\hat{\phi}$ estimated by both the IDT and SWD methods is also dependent on the sequence and order of projection axes used when computing the 1D optimal transport functions [153, 77]. This means that if the estimation is repeated using a different sequence of 1D projection axes, the correspondences $(x, \hat{\phi}(x))$ will be different. This can be seen in Figure 6.5. Here we show three subsets of point correspondences for each pair of shapes, estimated by the IDT algorithm using the same parameters given in Table 6.1, but changing the sequence of projection axes used to compute $\hat{\phi}$. We can see that in each case, a different set of correspondences are estimated. Similar results were generated using the SWD algorithm.

6.3 Conclusion

In this chapter we provided a short investigation into the application of optimal transport methods to 3D shape registration to see if they provide any benefits to this area. We investigated the optimal transport techniques proposed by Pitié et al.[6] and Bonneel et al. [77] which are based on estimating the optimal transportation in several 1D

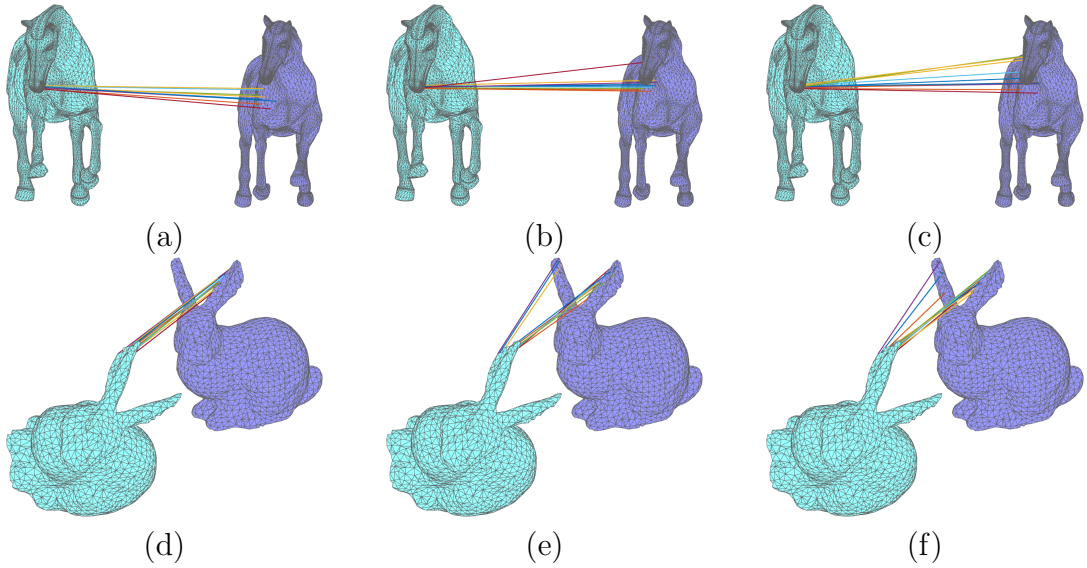


Figure 6.5: For both the horse shapes and bunny shapes we present three sets of correspondences $(x, \hat{\phi}_1(x))$, $(x, \hat{\phi}_2(x))$ and $(x, \hat{\phi}_3(x))$ created by three different transformations $\hat{\phi}_1$, $\hat{\phi}_2$ and $\hat{\phi}_3$. Each transformation is estimated using a different sequence of projection axes on which the 1D optimal transport functions are computed. We can see that in each case the same points in the model shapes are mapped to different points in the target shapes.

subspaces rather than in the higher dimensional space. We have found that both techniques can successfully transform one shape into another, with Pitié et al's technique providing very accurate results with little error. However, in order to estimate the transformation, the neighbourhood structure of the model point set is lost. When the shapes are similar or represent the same object, the point correspondences estimated often do not match corresponding areas, which can be undesirable. We also note that estimating the transformation $\hat{\phi}$ using two different sequences of projection axes can result in two different transformations being estimated, which can be undesirable in some applications.

In many 3D shape registration applications a smooth transformation is desired, which estimates the correct correspondences between model and target point sets of the same object. Using optimal transport techniques such as that proposed by Pitié et al. and Bonneel et al. would therefore not be suitable in this case. However, if the point clouds correspond to different objects and the correspondences estimated are

unimportant, these methods should be considered.

In order to reduce the computational cost of our optimisation algorithm, a similar method of dimension reduction could be implemented. Instead of minimising the $\mathcal{L}_2$ distance between two point sets in 3D, the point sets could be projected into several 1-dimensional subspaces in which the $\mathcal{L}_2$ distance could be minimised. However, unless additional constraints were added to the estimated transformation $\hat{\phi}$, the same issues arising in the IDT and SWD methods may occur. Investigating possible constraints that would allow estimation of a more suitable transformation $\hat{\phi}$ could be explored in future.

Chapter 7

Conclusions

The $\mathcal{L}_2$ distance is a robust metric which has been previously proposed for the registration of point sets. In this thesis, we explored the application of the $\mathcal{L}_2$ distance to the colour transfer problem and investigated the advantages that this framework can bring to the area. For shape registration, we also explored how directional data can enhance registration results when estimating a rotation or non-rigid deformation. In this chapter, we give an overview of the work presented in this thesis and the contributions achieved. We also summarise potential avenues of future work that could be explored.

7.1 Summary

In this thesis, we have proposed a method for colour transfer which modifies the colour feel of a target image or video using a colour palette provided by another image or video. The main contributions of this method are reviewed here:

1. **Application of the $\mathcal{L}_2$ distance to the colour transfer problem:** We presented a method which minimises a divergence between PDFs to register the colour distribution of images. Similar techniques have been previously proposed in the field of shape registration and we investigated how this type of technique performed when applied to the colour transfer problem. Using a similar method to that proposed by Jian et al. [41] we model the colour distribution of two images using GMMs and minimise their $\mathcal{L}_2$ distance to match their colour palettes.

Our formulation uses continuous PDFs as opposed to discrete histograms and has been shown to reduce the artifacts typically created by the discrete nature of other colour transfer methods, such as optimal transport.

2. **Framework which can be enhanced by correspondences:** Colour transfer techniques applied to images with similar content typically rely on feature correspondences, and many cannot be applied to images of different content where no corresponding feature points are available [71, 66, 91]. On the other hand, methods which do not take into account feature correspondences when they are available often cannot produce results comparable to those that do. The colour transfer method proposed in this thesis can be applied both to images with similar and different content. It can be easily enhanced by correspondences when they are available and we have shown that the results generated are comparable to other state of the art methods, with the robust nature of the $\mathcal{L}_2$ distance ensuring that outlier correspondences do not effect the results.

3. **Exploration of parametric transformations:** We have proposed using a parametric TPS transformation to register the colour distribution of two images and show that it creates smooth colour transfer results. The parametric framework also allows the recolouring step to be parallelised, since the estimated transformation can be applied to each pixel independently, and we present results on the speed achievable when this parallelisation is implemented. We have also shown that the parametric transformation allows for the recolouring of video sequences and the generation of interesting special effects in both image and video.

In Chapter 5, we also proposed a new method to tackle the shape registration problem. The main contributions that we have made to this field include:

1. **Adding directional data to the shape model for shape registration:** We have proposed a method for shape registration which explicitly models both the point positions and normal vector information of a shape, and show that in many cases our framework gives improved accuracy over other state of the art techniques. We also proposed modelling the directional data using the von Mises-

Fisher distribution, a framework which was previously proposed for clustering directional data but has not been applied to the shape registration problem.

2. **Using the $\mathcal{L}_2$ distance with the non-Euclidean von Mises-Fisher distribution:** We have shown that the scalar product between two von Mises-Fisher kernels has an explicit expression when $d = 3$, allowing us to explicitly compute the $\mathcal{L}_2$ distance between two KDEs with von Mises-Fisher kernels when our normal vectors are 3-dimensional. We have also shown that using Dirac kernels for one of the mixture models, and von Mises-Fisher kernels for the other, ensures that an explicit expression exists for the cross product between the two mixtures for any dimension d.

3. **Exploring the application of optimal transport to shape registration:** We have also investigated the application of two optimal transport techniques, which have been previously proposed for colour transfer, to the shape registration problem. We have shown that these techniques can successfully transform one point cloud to another. However, the correspondences estimated between shapes of the same object are incorrect, which can be disadvantageous in some applications.

7.2 Future Work

Learning user preferences for colour transfer: One of the advantages of our proposed colour transfer technique is the parametric transformation that we estimate which, once computed, can be easily stored and reused. This type of transformation could also be used to create an interface which allows the user to generate several colour transfer results, and interpolate between them until their preference is found, giving the user more control over the final result. As a good colour transfer result is very objective, learning techniques could also be applied to discover the type of colour transformation each user prefers. This information could then be used to tailor the colour transfer results to the user's own preference. Dimension reduction techniques such as PCA could also be applied to reduce the space needed to store the transformations.

Modelling colours in cylindrical colour space using von Mises-Fisher: In Chapter 4, we tested our colour transfer technique on colours represented in the RGB

and CIELab colour spaces. We modelled their colour distributions as GMMs in 3-dimensional space and registered them using the $\mathcal{L}_2$ distance. Other colour spaces such as HSL and HSV are cylindrical in nature, and the hue colour component can be represented as a unit vector in $\mathbb{S}$. In this case, the von Mises-Fisher distribution could be used to model the hue component, while Gaussian kernels could be used to model the remaining colour components. Modelling the colour distribution of images in these colour spaces, and the effect this has on our colour transfer technique, could be investigated, with the von Mises-Fisher distribution ensuring that the hue colour component is appropriately modelled.

Exploring new optimisation techniques: In terms of computational cost, we found that computing the $\mathcal{L}_2$ distance between GMMs with many mixtures when no correspondences are available can be computationally expensive, as can searching a high dimensional latent space using a gradient ascent technique. In order to reduce the number of mixtures in the GMM, a smaller number of Gaussians with differing weights and non-isotropic covariances could instead be used to model the shape. A faster optimisation algorithm could also be explored to further reduce the computational cost. A technique which performs the optimisation in lower dimensional spaces could be implemented, as discussed in Chapter 6, with added constraints to ensure that the estimated transformation computes correct correspondences when similar objects are registered.

Including more information in the density function: We have shown that adding normal vector information to the shape model can improve the accuracy of the shape registration result. Depending on the problem being solved and the data available, other shape information could also be added, including colour information, shape descriptors, or point confidence. This additional information would create a more detailed density, that could prove be more powerful in other applications than one containing only point information.

Appendix A

Additional Colour Transfer Results

In this Appendix we provide additional results generated using our proposed colour transfer approach, described in Chapter 4.

A.1 Images with similar content $P \simeq T$

Here we present some additional results similar to those presented in Figure 4.4 for the affine transformation, TPS transformation and the Gaussian, Inverse Multiquadric and Inverse Quadric RBF transformations. In all cases we present results for both the RGB and CIELab colour spaces. For notation details see Section 4.1.7.

In Figures A.1 and A.3 we compare results generated using the TPS and affine transformations and can see that in general a TPS transformation performs better than an affine one. When considering the results of Aff_{rgb}^{Corr} and Aff_{lab}^{Corr} in Figure A.1, we can see that in Column 1, the yellow slide has not been recoloured correctly, in Column 3 the shadows in the trees do not appear dark enough, and in Column 4 the light on the ceiling is not as bright as that in the palette image. Both TPS_{rgb}^{Corr} and TPS_{lab}^{Corr} create better results in these cases. Again in Figure A.3, when considering the results of Aff_{rgb}^{Corr} and Aff_{lab}^{Corr} we can see that in Column 3, the green appearing in the girl's dress does not match the palette image, and in Column 4 many of the colours appearing in the clothes do not match those in the palette. Again both TPS_{rgb}^{Corr} and TPS_{lab}^{Corr} create better results in these cases.

In Figures A.2 and A.4 we compare results generated using the Gaussian, Mul-

tiquadric and Inverse Quadric RBFs. From these results we can see that all RBFs (including TPS) perform similarly, with very little perceivable difference between the results.

When comparing results generated in RGB and CIELab colour space we found that in general the results were very similar, although in some cases computing the colour transfer function in CIELab space generated better results (the blue roof in Column 1 of Figures A.3 and A.4). As the RBF functions are smooth transformations, in certain cases they may not be able to transfer some colours from the palette to the target image due to the smoothness constraint. Changing the colour space, however, may allow these colours to be mapped more successfully.

A.2 Images with different content $P \neq T$

We also present additional results similar to those presented in Figure 4.7 for the affine transformation, TPS transformation and the Gaussian, Inverse Multiquadric and Inverse Quadric RBF transformations. In Figures A.5 and A.7 we compare the results generated using the TPS and affine transformations. While some results generated by these methods are quite similar, using a TPS transformation outperforms an affine transformation in others (Fig.A.5 Column 3). In Figures A.6 and A.8 we compare the results generated using the RBF transformation functions. We found that in some cases using the RGB colour space created better results than those generated using the CIELab colour space (Fig.A.6 Columns 1,2 and 4). The visual difference between the different RBF functions (including TPS) on the other hand is quite small. However, in some cases the Gaussian, Inverse Multiquadric and Inverse Quadric RBF functions can get caught in local minima when estimating θ (Fig. A.6 Column 3, row 3,5,7). In this case changing the colour space may improve the colour transfer result.

A.3 TPS versus Affine

In Figure A.9 we present some results comparing Aff_{rgb}^{KM} and TPS_{rgb}^{KM}. We can see that the TPS transformation creates good colour transfer results while the affine transformation can fail to transfer the variety of colours from the palette to the target (the

yellow and deep blues of the palette image in Column 2 appear more in the TPS result (row 4) than the affine result(row 3)). An affine transformation is also more likely to transform colours in the target to values outside of the RGB cube, creating blocky artifacts and gradient artifacts (area of sunlight in row 3, column 3).

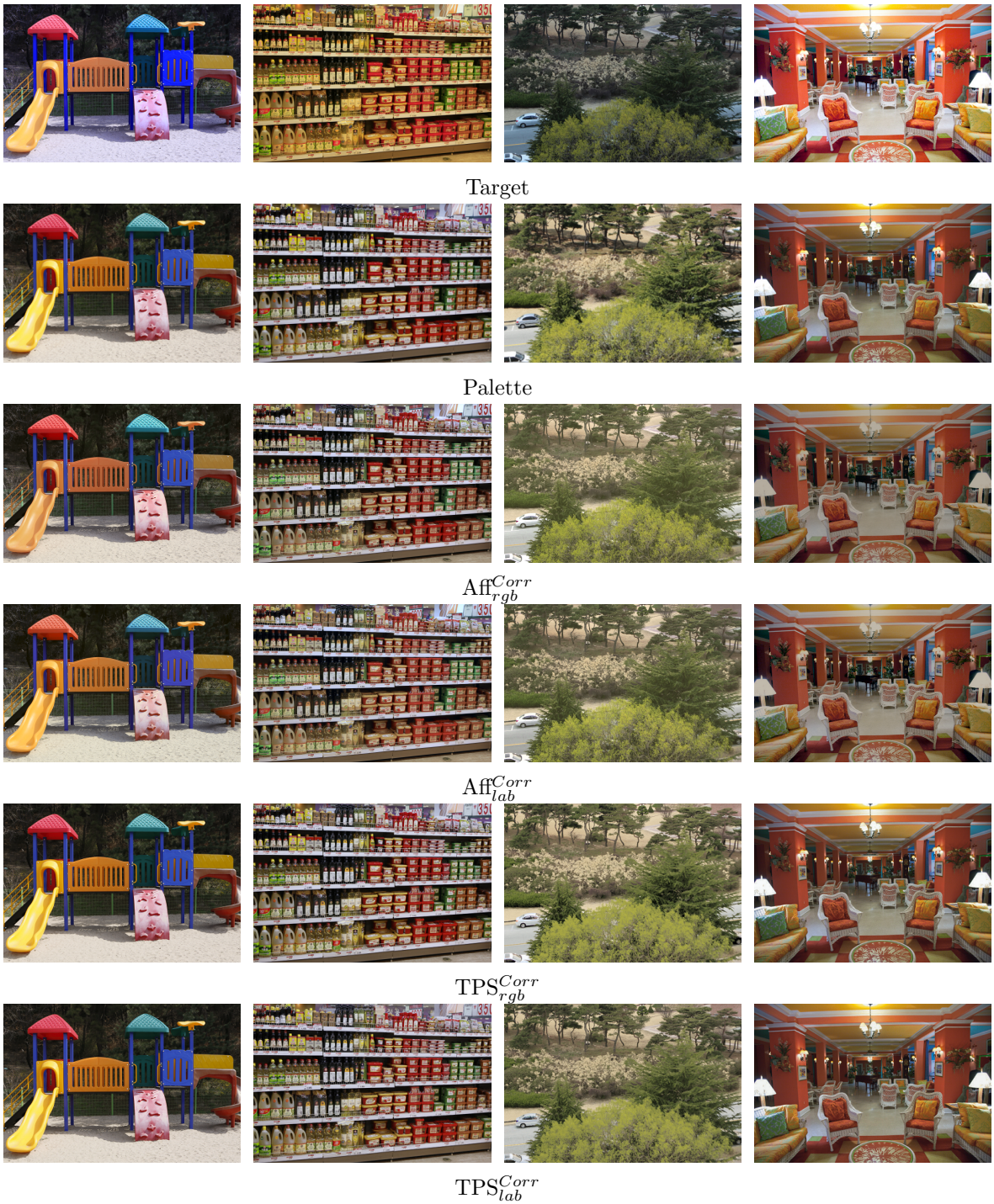


Figure A.1: Additional results on images with similar content. Row 1: Target images; Row 2: Palette images; Rows 3-4: Affine transformation results using correspondences in RGB and CIELab spaces; Rows 5-6: TPS transformation results using correspondences in RGB and CIELab spaces.

Palette

G_{rgb}^{Corr}

G_{lab}^{Corr}

$InMQ_{rgb}^{Corr}$

$InMQ_{lab}^{Corr}$

InQ_{rgb}^{Corr}

InQ_{lab}^{Corr}

Figure A.2: Row 1: Palette images (Target images can be seen in Row 1 of Figure A.1; Row (2-7) Results in RGB and CIELab spaces for the Gaussian RBF (Rows 2-3), Inverse Multiquadric RBF (Rows 4-5) and Inverse Quadric RBF(Row 6-7). 132

Figure A.3: Additional results on images with similar content. Row 1: Target images; Row 2: Palette images; Rows 3-4: Affine transformation results using correspondences in RGB and CIELab spaces; Rows 5-6: TPS transformation results using correspondences in RGB and CIELab spaces.

Palette

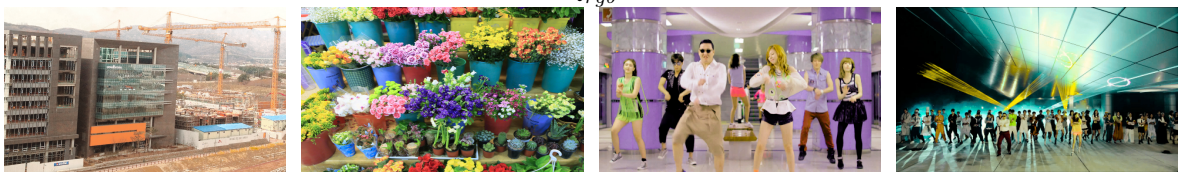
G_{rgb}^{Corr}

G_{lab}^{Corr}

$InMQ_{rgb}^{Corr}$

$InMQ_{lab}^{Corr}$

InQ_{rgb}^{Corr}



InQ_{lab}^{Corr}

Figure A.4: Row 1: Palette images (Target images can be seen in Row 1 of Figure A.3; Row (2-7) Results in RGB and CIELab spaces for the Gaussian RBF (Rows 2-3), Inverse Multiquadric RBF (Rows 4-5) and Inverse Quadric RBF(Row 6-7). 134

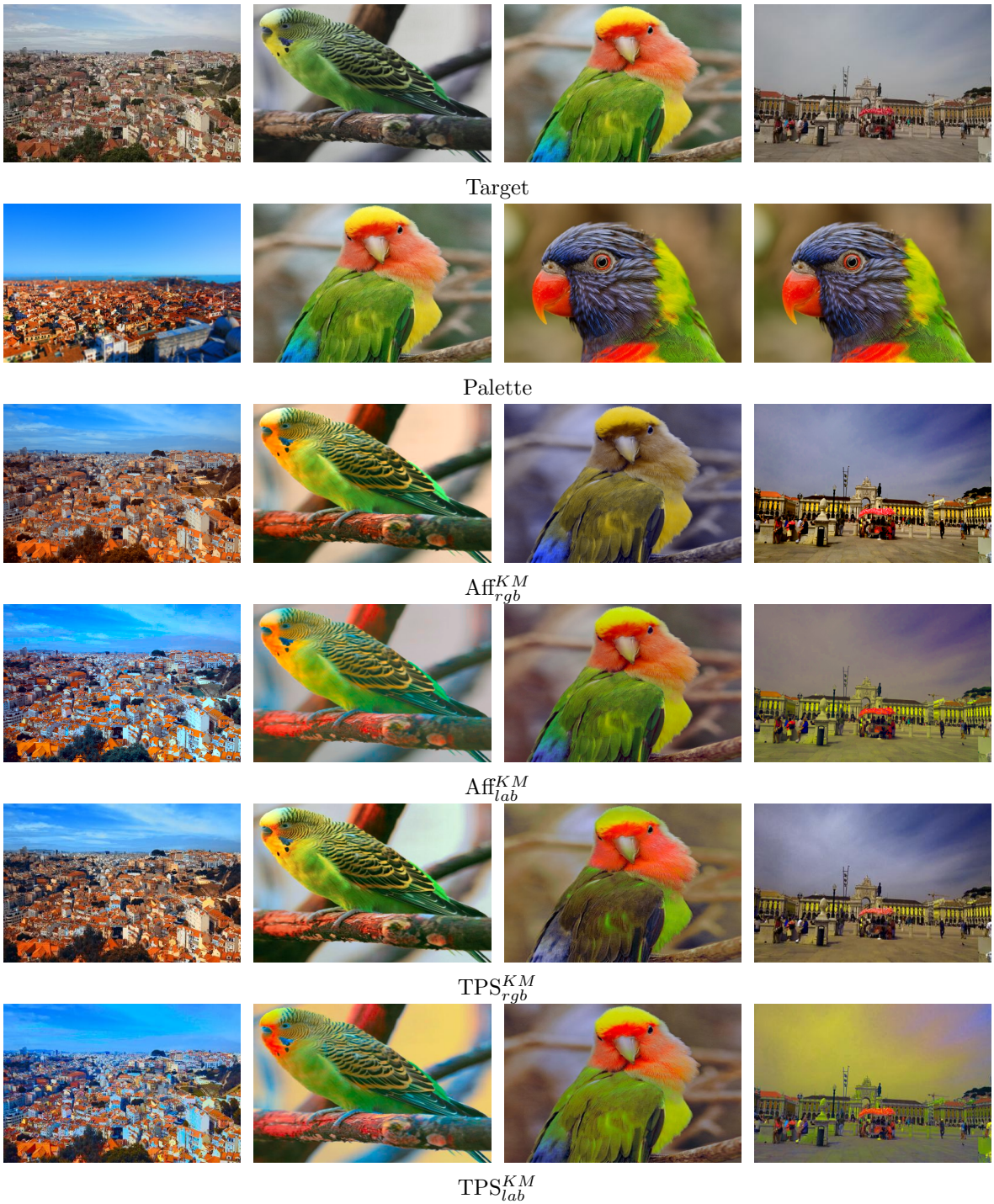


Figure A.5: Results when using an affine and TPS transformation function in RGB and CIELab colour space. In columns 1,2 and 4 the affine and TPS methods perform similarly, while in column 3 TPS outperforms the affine transformation.

Figure A.6: Results when using different RBF functions in RGB and CIELab colour space.

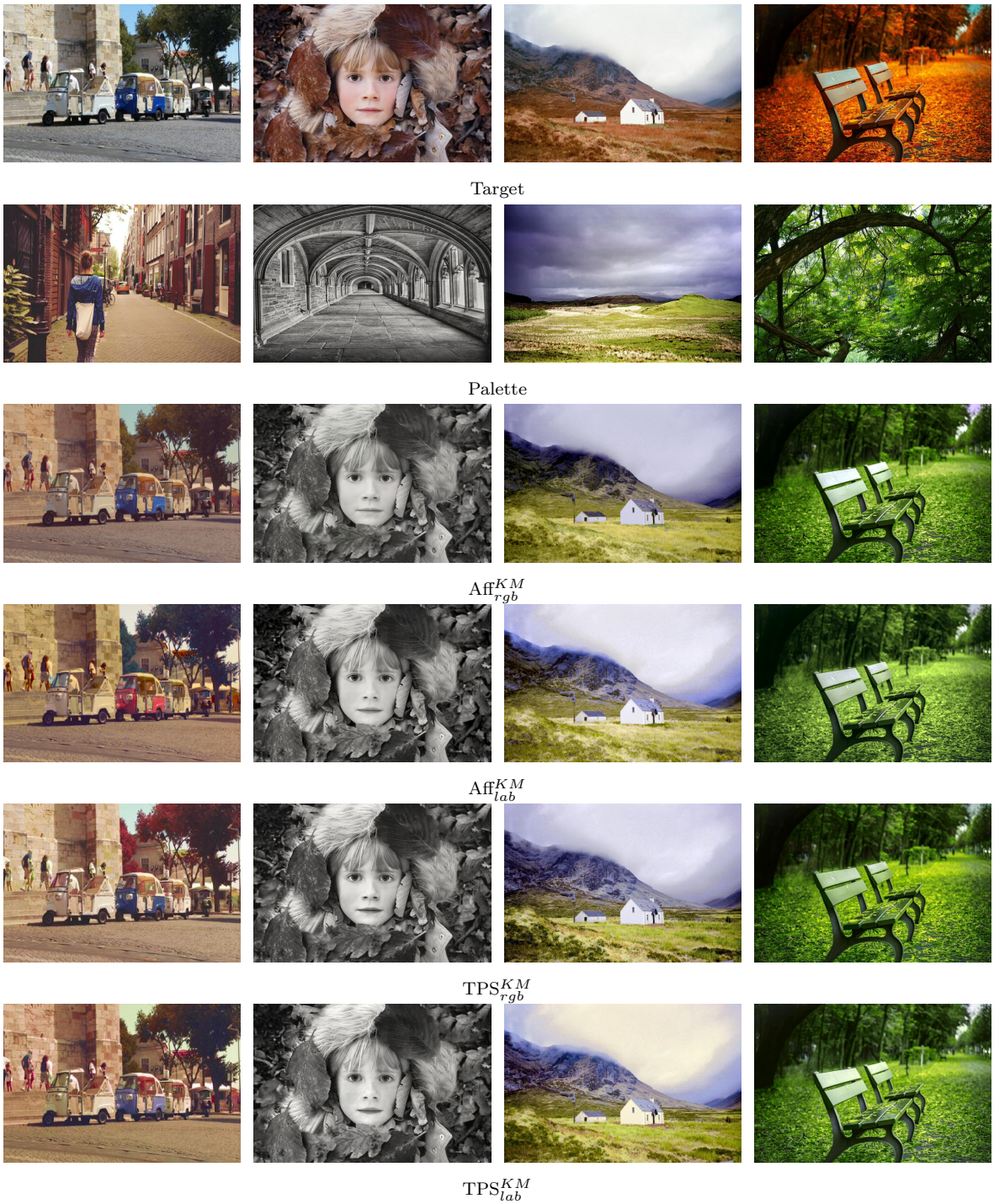


Figure A.7: Results when using an affine and TPS transformation function in RGB and CIELab colour space.

Figure A.8: Results when using different RBF functions in RGB and CIELab colour space.

Figure A.9: Affine versus TPS colour transfer results. Row 1: Target images; Row 2: Palette images; Row 3: Affine results; Row 4: TPS results. We can see that the TPS transformation creates good colour transfer results while the affine transformation can fail to transfer the variety of colours from the palette to the target (the yellow and deep blues of the palette image in Column 2 appear more in the TPS result (row 4) than the affine result(row 3)). An affine transformation is also more likely to transform colours in the target to values outside of the RGB cube, creating blocky artifacts and gradient artifacts (area of sunlight in row 3, column 3).

Appendix B

Parameters

In this appendix we provide the parameters used for all methods tested in Section 5.5. When comparing with state of the art registration methods (Go ICP [151], CPD [150] and GLMD[33]) we use the parameters provided in the code supplied by the authors.

B.1 2D Rotation Registration

For 2D rigid registration presented in Section 5.5.1 we compared $\mathcal{C}^x$, $\mathcal{C}^u_\delta$, $\mathcal{C}^u$, $\mathcal{C}^{x,u}_\delta$, and $\mathcal{C}^{x,u}$. Table B.1 details the parameters used in each case.

	$\mathcal{C}^x$	$\mathcal{C}^u_\delta$	$\mathcal{C}^u$	$\mathcal{C}^{x,u}_\delta$	$\mathcal{C}^{x,u}$
κ_{init}	✗	$\frac{5}{2^5}$	$\frac{20}{2^5}$	$\frac{5}{2^5}$	$\frac{20}{2^5}$
κ_{step}	✗	2	2	2	2
κ_{final}	✗	5	20	5	20
h_{init}	$4^5 \times 0.007 \times c$	✗	✗	$4^5 \times 0.007 \times c$	$4^5 \times 0.007 \times c$
h_{step}	$\frac{1}{4}$	✗	✗	$\frac{1}{4}$	$\frac{1}{4}$
h_{final}	$0.007 \times c$	✗	✗	$0.007 \times c$	$0.007 \times c$

Table B.1: Parameters used for $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^x$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ in our 2D rigid registration experiments. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the model shape S_1.

	$\mathcal{C}^x$	$\mathcal{C}^u_\delta$	$\mathcal{C}^u$	$\mathcal{C}^{x,u}_\delta$	$\mathcal{C}^{x,u}$
κ_{init}	✗	$\frac{50}{2^{14}}$	$\frac{25}{2^{14}}$	$\frac{50}{2^7}$	$\frac{25}{2^7}$
κ_{step}	✗	2	2	2	2
κ_{final}	✗	50	25	50	25
h_{init}	$2^7 \times 0.01\times$ c	✗	✗	$2^7 \times 0.01\times$ c	$2^7 \times 0.01\times$ c
h_{step}	$\frac{1}{2}$	✗	✗	$\frac{1}{2}$	$\frac{1}{2}$
h_{final}	$0.01\times$ c	✗	✗	$0.01\times$ c	$0.01\times$ c

Table B.2: Parameters used for $\mathcal{C}^u$, $\mathcal{C}^u_\delta$ $\mathcal{C}^x$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ in our 3D rigid registration experiments when the same sampling of S_1 and S_2 is used. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the model shape S_1.

B.2 3D Rotation Registration

In Section 5.5.2, when testing several registration techniques on 3D data differing by a rigid transformation, we first explored the case in which the points sampled from S_1 and S_2 had exact correspondence pairs. The parameters used by our proposed cost functions can be seen in Table B.2.

Next we registered shapes S_1 and S_2 that had no exact point correspondences. The parameters used in this case can be found in Table B.3. When registering shapes S_1 and S_2 when noise is added to S_2, we use the same parameters.

In Section 5.5.2 we compared our results to the Go ICP and CPD methods. In both cases, the code and parameters provided by the authors was used and can be found in Table B.4.

B.3 2D Non-Rigid Registraton

In Section 5.5.3 we registered 2D shapes differing by a non-rigid deformation and rotation and the parameters used for $\mathcal{C}^x$, $\mathcal{C}^{x,u}$, $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ can be seen in Table B.5. In this section we also compared our results to the CPD and GLMD algorithms. The parameters used for the CPD method were the ones used by the authors in the code provided and can be seen in Table B.6. There were no parameters to set in the GLMD

	$\mathcal{C}^x$	$\mathcal{C}^u_\delta$	$\mathcal{C}^u$	$\mathcal{C}^{x,u}_\delta$	$\mathcal{C}^{x,u}$
κ_{init}	✗	$\frac{10}{2^7}$	$\frac{25}{2^7}$	$\frac{10}{2^7}$	$\frac{25}{2^7}$
κ_{step}	✗	2	2	2	2
κ_{final}	✗	10	25	10	25
h_{init}	$2^7 \times 0.06\times$ c	✗	✗	$2^7 \times 0.06\times$ c	$2^7 \times 0.06\times$ c
h_{step}	$\frac{1}{2}$	✗	✗	$\frac{1}{2}$	$\frac{1}{2}$
h_{final}	$0.06\times$ c	✗	✗	$0.06\times$ c	$0.06\times$ c

Table B.3: Parameters used for $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^x$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ in our 3D rigid registration experiments with different sampling of S_1 and S_2, and with added noise to the points S_2. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the shape S_1.

CPD	
Outliers (ω)	0.5
Max Iter	100
Tolerance	$1e^{-8}$

Go ICP	
MSE Threshold	0.0001
Trim Fraction	0
Nodes per dim	300

Table B.4: The CPD and Go ICP parameters used when registering 3D shapes differing by a rigid rotation, as given in the code provided by the authors.

algorithm.

	$\mathcal{C}^x$	$\mathcal{C}^{x,u}$	$\mathcal{C}^x_{corr}$	$\mathcal{C}^{x,u}_{corr}$
κ_{init}	✗	$\frac{15}{2^4}$	✗	$\frac{10}{1.5^4}$
κ_{step}	✗	2	✗	1.5
κ_{final}	✗	15	✗	10
h_{init}	$4^4 \times 0.007\times$	$4^4 \times 0.007\times$	$2^4 \times 0.007\times$ c	$2^4 \times 0.007\times$ c
h_{step}	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{2}$
h_{init}	$0.007\times$ c	$0.007\times$ c	$0.007\times$ c	$0.007\times$ c

Table B.5: Parameters used for $\mathcal{C}^x$, $\mathcal{C}^{x,u}$ $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ in our experiments. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the shape S_1.

CPD	
Outliers (ω)	0
λ	2
β	2
Max Iter	100
Tolerance	e^{-10}

Table B.6: The CPD parameters used when registering 2D shapes differing by a non-rigid deformation, as given in the code provided by the authors.

B.4 3D Non-Rigid Registraton

In Section 5.5.4 we tested $\mathcal{C}^{x,u}_{corr}$, $\mathcal{C}^{x}_{corr}$, CPD and GLMD when registering 3D shapes that differed by a non-rigid deformation. In our first experiment we used shapes that had exact point correspondences, and the parameters used for $\mathcal{C}^{x,u}_{corr}$ and $\mathcal{C}^{x}_{corr}$ are given in Table B.7. We also tested these cost functions on 3D shapes with unknown correspondences that had to be estimated and the parameters used in this case can be seen in Table B.8. For both experiments, the CPD parameters used are those given in Table B.9. Again, the GLMD setup used was that provided by the authors and no parameters needed to be set.

	$\mathcal{C}^{x}_{corr}$	$\mathcal{C}^{x,u}_{corr}$
κ_{init}	✗	$\frac{15}{2^5}$
κ_{step}	✗	2
κ_{final}	✗	15
h_{init}	$4^5 \times 0.0005\times$ c	$4^5 \times 0.0005\times$ c
h_{step}	$\frac{1}{4}$	$\frac{1}{4}$
h_{init}	$0.0005\times$ c	$0.0005\times$ c

Table B.7: Parameters used for $\mathcal{C}^{x}_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ in our experiments with 3D shapes differing by a non-rigid deformation with known correspondences. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the shape S_1.

	$\mathcal{C}^x_{corr}$	$\mathcal{C}^{x,u}_{corr}$
κ_{init}	✗	$\frac{15}{2^5}$
κ_{step}	✗	2
κ_{final}	✗	15
h_{init}	$2^5 \times 0.001\times$ c	$2^5 \times 0.001\times$ c
h_{step}	$\frac{1}{2}$	$\frac{1}{2}$
h_{init}	$0.001\times$ c	$0.001\times$ c

Table B.8: Parameters used for $\mathcal{C}^x_{corr}$ and $\mathcal{C}^{x,u}_{corr}$ in our experiments with 3D shapes differing by a non-rigid deformation with unknown correspondences that needed to be estimated. Here $c = \det(\frac{X^T X}{n_1})^{\frac{1}{2d}}$, where X is the $n_1 \times d$ matrix of vertices in the shape S_1.

CPD	
Outliers (ω)	0.1
λ	3
β	2
Max Iter	100
Tolerance	e^{-3}

Table B.9: The CPD parameters used when registering 3D shapes differing by a non-rigid deformation, as given in the code provided by the authors.

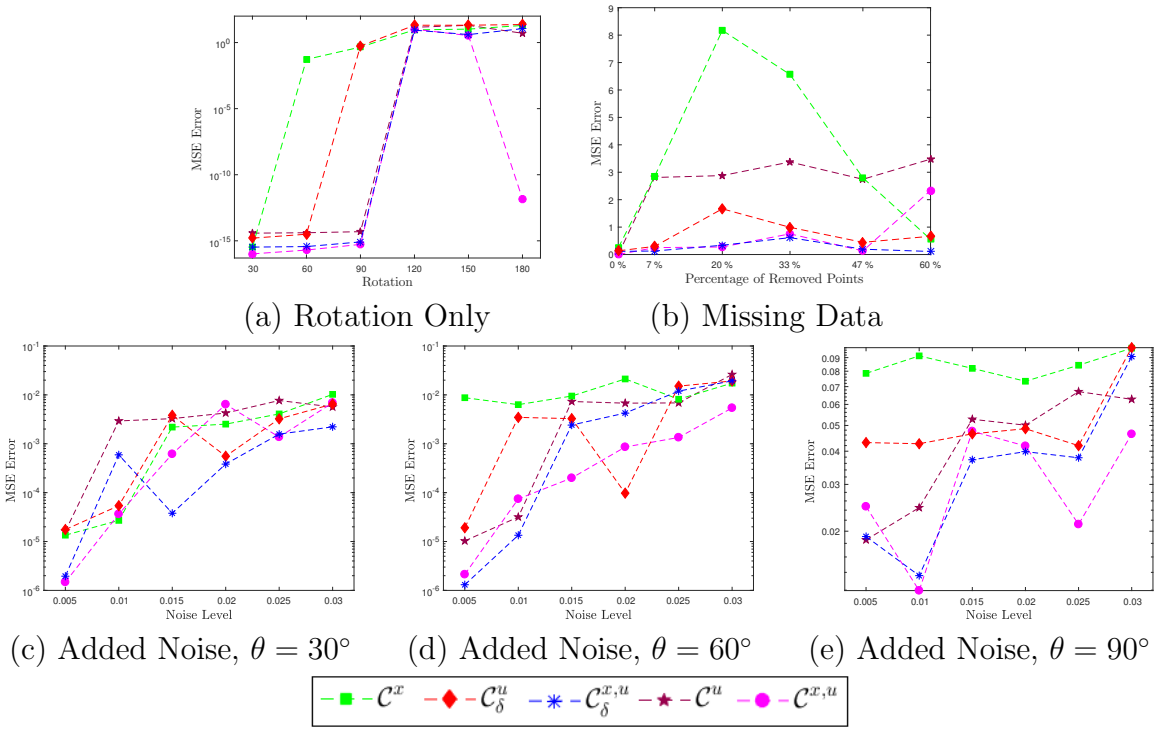
Appendix C

Additional Shape Registration Results

In Chapter 5 we propose cost functions $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ and when presenting the experimental results for rigid registration in Sections 5.5.1 and 5.5.2 we omit the results of some of these cost functions for clarity as we found that both $\mathcal{C}^u_\delta$ and $\mathcal{C}^u$ performed similarly, as did $\mathcal{C}^{x,u}_\delta$ and $\mathcal{C}^{x,u}$. In this appendix we provide similar results to those presented for rigid shape registration in Sections 5.5.1 and 5.5.2 for all four cost functions $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$.

C.1 2D Rotation Registration

In Figure C.1 we present results similar to those in Figure 5.3 for all cost functions $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. In Figure C.1(a) we present the results when a rotation is estimated using the full curves, and we can see that $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ perform similarly, as do $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. In Figure C.1(b) we present the results when a rotation is estimated using partial curves, with a percentage of points removed from the curve S_1. Again we can see that $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ perform similarly. While $\mathcal{C}^u_\delta$ seems to perform better than $\mathcal{C}^u$ in this case, both still perform better than $\mathcal{C}^x$ and worse than $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. In Figure C.1(c), (d) and (e) we present the registration results obtained when noise is added to the data. Again we can see that in general, $\mathcal{C}^{x,u}_\delta$ and $\mathcal{C}^{x,u}$ perform similarly, as do $\mathcal{C}^u_\delta$ and $\mathcal{C}^u$.



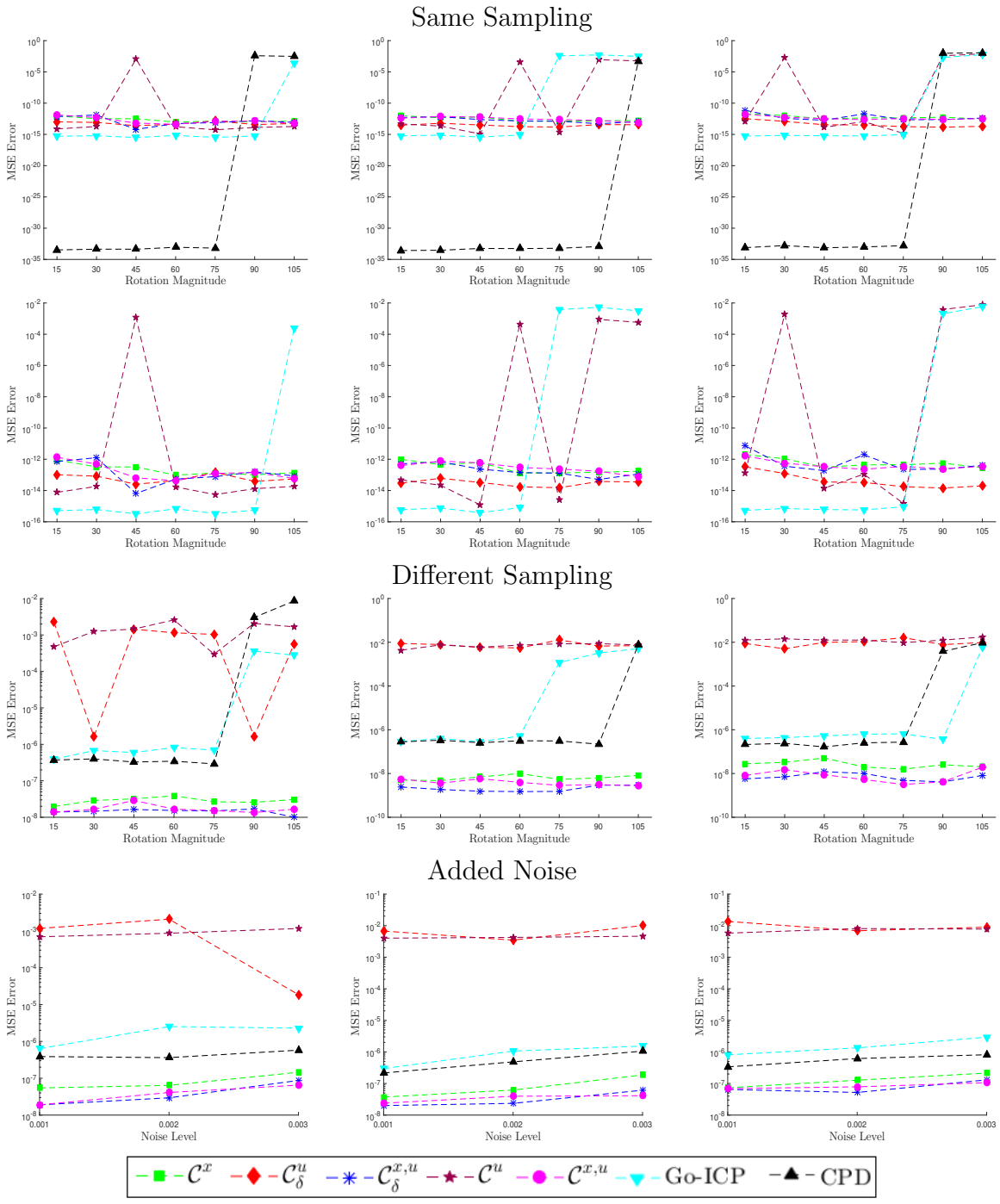
(a) Rotation Only (b) Missing Data

(c) Added Noise, $\theta = 30°$ (d) Added Noise, $\theta = 60°$ (e) Added Noise, $\theta = 90°$

Figure C.1: MSE results for each of our experiments on 2D data differing by a 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). In (c), (d) and (e) the MSE result is the average over 150 curve registration results for each level of noise.

C.2 3D Rotation Registration

In Figure C.2 we present results similar to those in Figure 5.10 for all cost functions $\mathcal{C}^u$, $\mathcal{C}^u_\delta$, $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$. From row 1 of Figure C.2 we can see that CPD performs best when estimating a rotation when there are exact point correspondences between S_1 and S_2. 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 C.2, row 1 and 2). When different samples are chosen from S_1 and S_2 (Fig C.2, row 3) both $\mathcal{C}^{x,u}$ and $\mathcal{C}^{x,u}_\delta$ perform similarly, and both outperform $\mathcal{C}^x$. Both $\mathcal{C}^u$ and $\mathcal{C}^u_\delta$ also give similar results, performing the worst. The same results were found when registering shapes with added noise (Fig C.2, row 4).



(a) Bunny (b) Dragon (c) Buddha

Figure C.2: Error results obtained when registering shapes S_1 and S_2 with the same sampling (row 1), omitting CPD for clarity (row 2), registering shapes with different sampling (row 3) and with added noise (row 4). Columns 1-3 give the error results for the Bunny, Dragon and Buddha meshes respectively.

Bibliography

[1] U. Castellani and A. Bartoli, *3D Shape Registration*, pp. 221–264. London: Springer London, 2012.

[2] R. Gutierrez-Osuna, "Kernel density estimation." `http://research.cs.tamu.edu/prism/lectures/pr/pr_l7.pdf`. Accessed: 2016-09-5.

[3] H. S. Faridul, T. Pouli, C. Chamaret, J. Stauder, A. Tremeau, and E. Reinhard, "A Survey of Color Mapping and its Applications," in *Eurographics 2014 - State of the Art Reports* (S. Lefebvre and M. Spagnuolo, eds.), The Eurographics Association, 2014.

[4] P. Bourke, "Ply - polygon file format." `http://paulbourke.net/dataformats/ply/`. Accessed: 2010-09-30.

[5] Y. Hwang, J.-Y. Lee, I. S. Kweon, and S. J. Kim, "Color transfer using probabilistic moving least squares," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 3342–3349, June 2014.

[6] F. Pitié, A. C. Kokaram, and R. Dahyot, "Automated colour grading using colour distribution transfer," *Comput. Vis. Image Underst.*, vol. 107, pp. 123–137, July 2007.

[7] F. Pitie, A. Kokaram, and R. Dahyot, "N-dimensional probability density function transfer and its application to color transfer," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1434–1439 Vol. 2, Oct 2005.

[8] A. Hasnat, *Unsupervised 3D image clustering and extension to joint color and depth segmentation.* PhD thesis, Université Jean Monnet de Saint-Etienne, France, 2014.

[9] M. Grogan and R. Dahyot, "Robust registration of gaussian mixtures for colour transfer," *Submitted to: ACM Transaction on Image Processing*, 2016.

[10] M. Grogan and R. Dahyot, "L2 registration for colour transfer in videos," in *Proceedings of the 12th European Conference on Visual Media Production*, CVMP '15, (New York, NY, USA), pp. 16:1–16:1, ACM, 2015.

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

[12] A. Bulbul, M. Grogan, and R. Dahyot, "3D reconstruction of reflective spherical surfaces from multiple images," in *Irish Machine Vision and Image Processing conference*, (Dublin, Ireland), August 2015.

[13] M. Grogan and R. Dahyot, "Mesh from depth images using GR2T," in *Irish Machine Vision and Image Processing Conference*, (Derry-Londonderry, Northern Ireland), pp. 15–20, August 2014.

[14] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, (New York, NY, USA), pp. 559–568, ACM, 2011.

[15] J. Maintz and M. A. Viergever, "A survey of medical image registration," *Medical Image Analysis*, vol. 2, no. 1, pp. 1 – 36, 1998.

[16] K. W. Bowyer, K. Chang, and P. Flynn, "A survey of approaches and challenges in 3d and multi-modal 3d + 2d face recognition," *Computer Vision and Image Understanding*, vol. 101, no. 1, pp. 1 – 15, 2006.

[17] J. Ma, J. Zhao, J. Tian, Z. Tu, and A. Yuille, "Robust estimation of nonrigid transformation for point set registration," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 2147–2154, June 2013.

[18] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. Langbein, Y. Liu, A. D. Marshall, R. Martin, X. Sun, and P. Rosin, "Registration of 3d point clouds and meshes: A survey from rigid to nonrigid," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, pp. 1199–1217, July 2013.

[19] P. J. Besl and H. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239–256, Feb 1992.

[20] L. G. Brown, "A survey of image registration techniques," *ACM Comput. Surv.*, vol. 24, pp. 325–376, Dec. 1992.

[21] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," in *Proceedings of the British Machine Vision Conference 2001, BMVC 2001, Manchester, UK, 10-13 September 2001*, pp. 1–10, 2001.

[22] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, pp. 2262–2275, Dec 2010.

[23] P. Besl and N. D. McKay, "A method for registration of 3-D shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, pp. 239–256, Feb 1992.

[24] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. J. Comput. Vision*, vol. 13, pp. 119–152, Oct. 1994.

[25] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," in *International Conference on 3-D Imaging and Modeling*, pp. 145–152, 2001.

[26] A. Censi, "An ICP variant using a point-to-line metric," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 19–25, May 2008.

[27] G. Sharp, S. Lee, and D. Wehe, "Maximum-likelihood registration of range images with missing data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, pp. 120–130, Jan 2008.

[28] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D registration efficiently and globally optimally," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, pp. 1457–1464, Dec 2013.

[29] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pp. 2724–2729 vol.3, Apr 1991.

[30] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 509–522, Apr 2002.

[31] S. Ge, G. Fan, and M. Ding, "Non-rigid point set registration with global-local topology preservation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 245–251, June 2014.

[32] J. Ma, J. Zhao, J. Tian, A. L. Yuille, and Z. Tu, "Robust point matching via vector field consensus," *IEEE Transactions on Image Processing*, vol. 23, pp. 1706–1721, April 2014.

[33] Y. Yang, S. H. Ong, and K. W. C. Foong, "A robust global and local mixture distance based non-rigid point set registration," *Pattern Recogn.*, vol. 48, pp. 156–173, Jan. 2015.

[34] Y. Zheng and D. Doermann, "Robust point matching for nonrigid shapes by preserving local neighborhood structures," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, pp. 643–649, April 2006.

[35] J. H. Lee and C. H. Won, "Topology preserving relaxation labeling for nonrigid point matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 427–432, Feb 2011.

[36] H. Chui and A. Rangarajan, "A new algorithm for non-rigid point matching," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 44–51 vol.2, 2000.

[37] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding*, vol. 89, no. 23, pp. 114 – 141, 2003. Nonrigid Image Registration.

[38] M. Sofka, G. Yang, and C. Stewart, "Simultaneous covariance driven correspondence (cdc) and transformation estimation in the expectation maximization framework," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pp. 1–8, June 2007.

[39] R. Horaud, F. Forbes, M. Yguel, G. Dewaele, and J. Zhang, "Rigid and articulated point registration with expectation conditional maximization," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, pp. 587–602, March 2011.

[40] G. Sanromí, R. AlquéZar, F. Serratosa, and B. Herrera, "Smooth point-set registration using neighboring constraints," *Pattern Recogn. Lett.*, vol. 33, pp. 2029–2037, Nov. 2012.

[41] B. Jian and B. Vemuri, "Robust point set registration using gaussian mixture models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, pp. 1633–1645, Aug 2011.

[42] Y. Tsin and T. Kanade, "A correlation-based approach to robust point set registration," in *Computer Vision - ECCV 2004* (T. Pajdla and J. Matas, eds.), vol. 3023 of *Lecture Notes in Computer Science*, pp. 558–569, Springer Berlin Heidelberg, 2004.

[43] C. Arellano and R. Dahyot, "Mean shift algorithm for robust rigid registration between gaussian mixture models," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pp. 1154–1158, Aug 2012.

[44] A. G. Bors and N. Nasios, "Kernel bandwidth estimation for nonparametric modeling," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, pp. 1543–1555, Dec 2009.

[45] D. Comaniciu, "An algorithm for data-driven bandwidth selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, pp. 281–288, Feb. 2003.

[46] D. W. Scott, "Parametric statistical modeling by minimum integrated square error," *Technometrics*, vol. 43, no. 3, pp. 274–285, 2001.

[47] J. Goldberger, S. Gordon, and H. Greenspan, "An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 487–493 vol.1, Oct 2003.

[48] F. Escolano, P. Suau, and B. Bonev, *Information theory in Computer Vision and Pattern Recognition*. Springer, 2009.

[49] D. W. Scott, "Parametric statistical modeling by minimum integrated square error," *Technometrics*, vol. 43, no. 3, pp. pp. 274–285, 2001.

[50] C. Arellano and R. Dahyot, "Robust ellipse detection with gaussian mixture models," *Pattern Recognition*, 2016.

[51] J. Ma, W. Qiu, J. Zhao, Y. Ma, A. Yuille, and Z. Tu, "Robust L_2E estimation of transformation for non-rigid registration," *Signal Processing, IEEE Transactions on*, vol. 63, pp. 1115–1129, March 2015.

[52] R. Sandhu, S. Dambreville, and A. Tannenbaum, "Point set registration via particle filtering and stochastic dynamics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1459–1473, Aug 2010.

[53] E. Hasanbelliu, L. Giraldo, and J. Principe, "A robust point matching algorithm for non-rigid registration using the cauchy-schwarz divergence," in *Machine Learning for Signal Processing (MLSP), 2011 IEEE International Workshop on*, pp. 1–6, Sept 2011.

[54] R. Dahyot and J. Ruttle, "Generalised relaxed radon transform (GR^2T) for robust inference," *Pattern Recognition*, vol. 46, no. 3, pp. 788 – 794, 2013.

[55] J. Ruttle, C. Arellano, and R. Dahyot, "Robust shape from depth images with GR^2T," *Pattern Recognition Letters*, vol. 50, no. 0, pp. 43 – 54, 2014. Depth Image Analysis.

[56] R. Dahyot, "Gr2t vs l2e with nuisance scale," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pp. 3857–3861, Aug 2014.

[57] W. Xu and J. Mulligan, "Performance evaluation of color correction approaches for automatic multi-view image and video stitching," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 263–270, June 2010.

[58] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, pp. 277–280, July 2002.

[59] Q. Wang, X. Sun, and Z. Wang, "A robust algorithm for color correction between two stereo images," in *Proceedings of the 9th Asian Conference on Computer Vision - Volume Part II*, ACCV'09, (Berlin, Heidelberg), pp. 405–416, Springer-Verlag, 2010.

[60] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, (Washington, DC, USA), pp. 1150–, IEEE Computer Society, 1999.

[61] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008.

[62] K. Yamamoto, T. Yendo, T. Fujii, M. Tanimoto, and D. Suter, "Color correction for multi-camera system by using correspondences," in *ACM SIGGRAPH 2006 Research Posters*, SIGGRAPH '06, (New York, NY, USA), ACM, 2006.

[63] K. Yamamoto and R. Oi, "Color correction for multi-view video using energy minimization of view networks," *International Journal of Automation and Computing*, vol. 5, no. 3, pp. 234–245, 2008.

[64] M. P. Tehrani, A. Ishikawa, S. Sakazawa, and A. Koike, "Iterative colour correction of multicamera systems using corresponding feature points," *Journal of Visual Communication and Image Representation*, vol. 21, no. 56, pp. 377 – 391, 2010. Special issue on Multi-camera Imaging, Coding and Innovative Display.

[65] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, June 1981.

[66] Y. Hwang, J.-Y. Lee, I. S. Kweon, and S. J. Kim, "Color transfer using probabilistic moving least squares," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 3342–3349, June 2014.

[67] Q. Wang, X. Sun, and Z. Wang, "A robust algorithm for color correction between two stereo images," in *Computer Vision ACCV 2009* (H. Zha, R.-i. Taniguchi, and S. Maybank, eds.), vol. 5995 of *Lecture Notes in Computer Science*, pp. 405–416, Springer Berlin Heidelberg, 2010.

[68] S. Kagarlitsky, Y. Moses, and Y. Hel-Or, "Piecewise-consistent color mappings of images acquired under various conditions," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2311–2318, Sept 2009.

[69] Y. Chen, K. K. Ma, and C. Cai, "Histogram-offset-based color correction for multi-view video coding," in *2010 IEEE International Conference on Image Processing*, pp. 977–980, Sept 2010.

[70] C. Doutre and P. Nasiopoulos, "Color correction preprocessing for multiview video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 1400–1406, Sept 2009.

[71] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Non-rigid dense correspondence with applications for image enhancement," *ACM Trans. Graph.*, vol. 30, pp. 70:1–70:10, July 2011.

[72] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *Proceedings of the 11th Euro-*

pean Conference on Computer Vision Conference on Computer Vision: Part III, ECCV'10, (Berlin, Heidelberg), pp. 29–43, Springer-Verlag, 2010.

[73] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Optimizing color consistency in photo collections," *ACM Trans. Graph.*, vol. 32, pp. 38:1–38:10, July 2013.

[74] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *Computer Graphics and Applications, IEEE*, vol. 21, pp. 34–41, Sep 2001.

[75] J. Shi, W. Zhang, and Y. Wang, "Shape analysis with hyperbolic wasserstein distance," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5051–5061, June 2016.

[76] F. Pitie and A. Kokaram, "The linear monge-kantorovitch linear colour mapping for example-based colour transfer," in *4th European Conference on Visual Media Production*, pp. 1–9, Nov 2007.

[77] N. Bonneel, J. Rabin, G. Peyr, and H. Pfister, "Sliced and radon wasserstein barycenters of measures," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, pp. 22–45, 2015.

[78] L. Neumann and A. Neumann, "Color style transfer techniques using hue, lightness and saturation histogram matching," in *Proceedings of the First Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, Computational Aesthetics'05, (Aire-la-Ville, Switzerland, Switzerland), pp. 111–122, Eurographics Association, 2005.

[79] N. Papadakis, E. Provenzi, and V. Caselles, "A variational model for histogram transfer of color images," *Image Processing, IEEE Transactions on*, vol. 20, pp. 1682–1695, June 2011.

[80] T. Pouli and E. Reinhard, "Progressive color transfer for images of arbitrary dynamic range," *Computers & Graphics*, vol. 35, no. 1, pp. 67 – 80, 2011. Extended Papers from Non-Photorealistic Animation and Rendering (NPAR) 2010.

[81] D. Freedman and P. Kisilev, "Object-to-object color transfer: Optimal flows and smsp transformations," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 287–294, June 2010.

[82] S. Ferradans, N. Papadakis, J. Rabin, G. Peyr, and J.-F. Aujol, "Regularized discrete optimal transport," in *Scale Space and Variational Methods in Computer Vision* (A. Kuijper, K. Bredies, T. Pock, and H. Bischof, eds.), vol. 7893 of *Lecture Notes in Computer Science*, pp. 428–439, Springer Berlin Heidelberg, 2013.

[83] X. Xiao and L. Ma, "Gradient-preserving color transfer," *Computer Graphics Forum*, vol. 28, no. 7, pp. 1879–1886, 2009.

[84] J. Rabin, J. Delon, and Y. Gousseau, "Removing artefacts from color and contrast modifications," *Image Processing, IEEE Transactions on*, vol. 20, pp. 3073–3085, Nov 2011.

[85] O. Frigo, N. Sabater, V. Demoulin, and P. Hellier, "Optimal transportation for example-guided color transfer," in *Computer Vision – ACCV 2014* (D. Cremers, I. Reid, H. Saito, and M.-H. Yang, eds.), vol. 9005 of *Lecture Notes in Computer Science*, pp. 655–670, Springer International Publishing, 2015.

[86] K. Jeong and C. Jaynes, "Object matching in disjoint cameras using a color transfer approach," *Machine Vision and Applications*, vol. 19, no. 5, pp. 443–455, 2007.

[87] Y. Xiang, B. Zou, and H. Li, "Selective color transfer with multi-source images," *Pattern Recognition Letters*, vol. 30, no. 7, pp. 682 – 689, 2009.

[88] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 603–619, May 2002.

[89] S. Xu, Y. Zhang, S. Zhang, and X. Ye, "Uniform color transfer," in *IEEE International Conference on Image Processing 2005*, vol. 3, pp. III–940–3, Sept 2005.

[90] Y.-W. Tai, J. Jia, and C.-K. Tang, "Local color transfer via probabilistic segmentation by expectation-maximization," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 747–754 vol. 1, June 2005.

[91] M. Oliveira, A. Sappa, and V. Santos, "A probabilistic approach for color correction in image mosaicking applications," *Image Processing, IEEE Transactions on*, vol. 24, pp. 508–523, Feb 2015.

[92] T. Oskam, A. Hornung, R. Sumner, and M. Gross, "Fast and stable color balancing for images and augmented reality," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pp. 49–56, Oct 2012.

[93] O. Dalmau-Cedeno, M. Rivera, and P. Mayorga, "Computing the α-channel with probabilistic segmentation for image colorization," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–7, Oct 2007.

[94] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski, "Interactive local adjustment of tonal values," *ACM Trans. Graph.*, vol. 25, pp. 646–653, July 2006.

[95] X. An and F. Pellacini, "Appprop: All-pairs appearance-space edit propagation," *ACM Trans. Graph.*, vol. 27, pp. 40:1–40:9, Aug. 2008.

[96] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 696–706, May 2002.

[97] T. Gevers and A. W. Smeulders, "Color-based object recognition," *Pattern Recognition*, vol. 32, no. 3, pp. 453 – 464, 1999.

[98] S. J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler, "Tracking groups of people," *Computer Vision and Image Understanding*, vol. 80, no. 1, pp. 42 – 56, 2000.

[99] M. Tkalcic and J. F. Tasic, "Colour spaces: perceptual, historical and applicational background," in *EUROCON 2003. Computer as a Tool. The IEEE Region 8*, vol. 1, pp. 304–308 vol.1, Sept 2003.

[100] A. Ford and A. Roberts, "Colour Space Conversions." `http://sites.biology.duke.edu/johnsenlab/pdfs/tech/colorconversion.pdf`, 1998. [Online; accessed 12-August-2016].

[101] R. Szeliski, *Computer Vision: Algorithms and Applications*. New York, NY, USA: Springer-Verlag New York, Inc., 1st ed., 2010.

[102] D. Zhang and G. Lu, "Review of shape representation and description techniques," *Pattern Recognition*, vol. 37, no. 1, pp. 1 – 19, 2004.

[103] d. F. M. Zhang, L. and F. A., "Survey on 3d shape descriptors," tech. rep., Fundacao para a Cincia e a Tecnologia: Lisboa, Portugal, 2007.

[104] X. Yang and Y. Tian, "Super normal vector for activity recognition using depth sequences," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 804–811, June 2014.

[105] S. Tang, X. Wang, X. Lv, T. X. Han, J. Keller, Z. He, M. Skubic, and S. Lao, *Histogram of Oriented Normal Vectors for Object Recognition with a Depth Sensor*, pp. 525–538. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[106] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," 2006.

[107] N. Mai-Duy and T. Tran-Cong, "Approximation of function and its derivatives using radial basis function networks," *Applied Mathematical Modelling*, vol. 27, no. 3, pp. 197 – 220, 2003.

[108] F. L. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 567–585, Jun 1989.

[109] B. Fornberg and C. Piret, "On choosing a radial basis function and a shape parameter when solving a convective {PDE} on a sphere," *Journal of Computational Physics*, vol. 227, no. 5, pp. 2758 – 2780, 2008.

[110] S. Y. Chun and J. A. Fessler, "A simple regularizer for b-spline nonrigid image registration that encourages local invertibility," *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, pp. 159–169, Feb 2009.

[111] R. Szeliski and J. Coughlan, "Spline-based image registration," *International Journal of Computer Vision*, vol. 22, no. 3, pp. 199–218, 1997.

[112] X. Huang, N. Paragios, and D. N. Metaxas, "Shape registration in implicit spaces using information theory and free form deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1303–1318, Aug 2006.

[113] M. Holden, "A review of geometric transformations for nonrigid body registration," *IEEE Transactions on Medical Imaging*, vol. 27, pp. 111–128, Jan 2008.

[114] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: application to breast mr images," *IEEE Transactions on Medical Imaging*, vol. 18, pp. 712–721, Aug 1999.

[115] T. Rohlfing, C. R. Maurer, D. A. Bluemke, and M. A. Jacobs, "Volume-preserving nonrigid registration of mr breast images using free-form deformation with an incompressibility constraint," *IEEE Transactions on Medical Imaging*, vol. 22, pp. 730–741, June 2003.

[116] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, p. 2012.

[117] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, June 2014.

[118] A. Selim, M. Elgharib, and L. Doyle, "Painting style transfer for head portraits using convolutional neural networks," *ACM Trans. Graph.*, vol. 35, pp. 129:1–129:18, July 2016.

[119] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[120] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," *ECCV*, 2016.

[121] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 1915–1929, Aug 2013.

[122] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *CoRR*, vol. abs/1508.06576, 2015.

[123] A. Goshtasby, "Piecewise linear mapping functions for image registration," *Pattern Recognition*, vol. 19, no. 6, pp. 459 – 466, 1986.

[124] A. Goshtasby, "Piecewise cubic mapping functions for image registration," *Pattern Recognition*, vol. 20, no. 5, pp. 525 – 533, 1987.

[125] L. Zagorchev and A. Goshtasby, "A comparative study of transformation functions for nonrigid image registration," *IEEE Transactions on Image Processing*, vol. 15, pp. 529–538, March 2006.

[126] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, pp. 3–15, Jan 2003.

[127] N. Amenta and Y. J. Kil, "Defining point-set surfaces," in *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, (New York, NY, USA), pp. 264–270, ACM, 2004.

[128] C. Shen, J. F. O'Brien, and J. R. Shewchuk, "Interpolating and approximating implicit surfaces from polygon soup," in *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, (New York, NY, USA), pp. 896–904, ACM, 2004.

[129] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Trans. Graph.*, vol. 25, pp. 533–540, July 2006.

[130] S. Haker, A. Tannenbaum, and R. Kikinis, *Mass Preserving Mappings and Image Registration*, pp. 120–127. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

[131] S. Haker, L. Zhu, A. Tannenbaum, and S. Angenent, "Optimal mass transport for registration and warping," *International Journal of Computer Vision*, vol. 60, no. 3, pp. 225–240, 2004.

[132] T. ur Rehman, E. Haber, G. Pryor, J. Melonakos, and A. Tannenbaum, "3d nonrigid registration via optimal mass transport on the {GPU}," *Medical Image Analysis*, vol. 13, no. 6, pp. 931 – 940, 2009. Includes Special Section on Computational Biomechanics for Medicine.

[133] F. Gustafsson and G. Hendeby, "On nonlinear transformations of stochastic variables and its application to nonlinear filtering," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3617–3620, March 2008.

[134] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, pp. 1–37, Dec. 2007.

[135] M. Carreira-Perpinan, "Gaussian mean-shift is an em algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 767 – 776, May 2007.

[136] D. F. Shanno, "Conditioning of quasi-newton methods for function minimization," *Mathematics of Computation*, vol. 24, no. 111, pp. 647–656, 1970.

[137] H. Hristova, O. Le Meur, R. Cozot, and K. Bouatouch, "Style-aware robust color transfer," in *Proceedings of the Workshop on Computational Aesthetics*, CAE '15, (Aire-la-Ville, Switzerland, Switzerland), pp. 67–77, Eurographics Association, 2015.

[138] N. Bonneel, K. Sunkavalli, S. Paris, and H. Pfister, "Example-based video color grading," *ACM Trans. Graph.*, vol. 32, pp. 39:1–39:12, July 2013.

[139] H. Chang, O. Fried, Y. Liu, S. DiVerdi, and A. Finkelstein, "Palette-based photo recoloring," *ACM Trans. Graph.*, vol. 34, pp. 139:1–139:11, July 2015.

[140] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu, "Automatic photo adjustment using deep neural networks," *ACM Trans. Graph.*, vol. 35, pp. 11:1–11:15, Feb. 2016.

[141] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, 2005.

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

[143] J. Traa and P. Smaragdis, "Multiple speaker tracking with the factorial von mises-fisher filter," in *IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING*, (REIMS, FRANCE), September 21-24 2014.

[144] R. Fisher, "Dispersion on a sphere," *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 217, no. 1130, pp. 295–305, 1953.

[145] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis, "Improved fast gauss transform and efficient kernel density estimation," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 664–671 vol.1, Oct 2003.

[146] R. Hoffman and A. K. Jain, "Segmentation and classification of range images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp. 608–620, Sept 1987.

[147] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *SIGGRAPH Comput. Graph.*, vol. 26, pp. 71–78, July 1992.

[148] H. R. Byrd, C. J. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Mathematical Programming*, vol. 89, no. 1, pp. 149–185, 2000.

[149] R. Waltz, J. Morales, J. Nocedal, and D. Orban, "An interior algorithm for nonlinear optimization that combines line search and trust region steps," *Mathematical Programming*, vol. 107, no. 3, pp. 391–408, 2006.

[150] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 2262–2275, Dec 2010.

[151] J. Yang, H. Li, and Y. Jia, "Go-icp: Solving 3d registration efficiently and globally optimally," in *2013 IEEE International Conference on Computer Vision*, pp. 1457–1464, Dec 2013.

[152] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," *ACM Trans. Graph.*, vol. 23, pp. 399–405, Aug. 2004.

[153] N. Bonnotte, *Unidimensional and Evolution Methods for Optimal Transportation*. Theses, Université Paris Sud - Paris XI, Dec. 2013.