Terms and Conditions of Use of Digitised Theses from Trinity College Library Dublin

Copyright statement

All material supplied by Trinity College Library is protected by copyright (under the Copyright and Related Rights Act, 2000 as amended) and other relevant Intellectual Property Rights. By accessing and using a Digitised Thesis from Trinity College Library you acknowledge that all Intellectual Property Rights in any Works supplied are the sole and exclusive property of the copyright and/or other IPR holder. Specific copyright holders may not be explicitly identified. Use of materials from other sources within a thesis should not be construed as a claim over them.

A non-exclusive, non-transferable licence is hereby granted to those using or reproducing, in whole or in part, the material for valid purposes, providing the copyright owners are acknowledged using the normal conventions. Where specific permission to use material is required, this is identified and such permission must be sought from the copyright holder or agency cited.

Liability statement

By using a Digitised Thesis, I accept that Trinity College Dublin bears no legal responsibility for the accuracy, legality or comprehensiveness of materials contained within the thesis, and that Trinity College Dublin accepts no liability for indirect, consequential, or incidental, damages or losses arising from use of the thesis for whatever reason. Information located in a thesis may be subject to specific use constraints, details of which may not be explicitly described. It is the responsibility of potential and actual users to be aware of such constraints and to abide by them. By making use of material from a digitised thesis, you accept these copyright and disclaimer provisions. Where it is brought to the attention of Trinity College Library that there may be a breach of copyright or other restraint, it is the policy to withdraw or take down access to a thesis while the issue is being resolved.

Access Agreement

By using a Digitised Thesis from Trinity College Library you are bound by the following Terms & Conditions. Please read them carefully.

I have read and I understand the following statement: All material supplied via a Digitised Thesis from Trinity College Library is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of a thesis is not permitted, except that material may be duplicated by you for your research use or for educational purposes in electronic or print form providing the copyright owners are acknowledged using the normal conventions. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

$\mathcal{L}_2$ Inference for Shape Parameter Estimation

by

Claudia L. Arellano Vidal, MSc

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 2014

To my parents, Patricio and Ximena

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Claudia L. Arellano Vidal

April 1, 2014

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Claudia L. Arellano Vidal

April 1, 2014

Abstract

In this thesis, we propose a method to robustly estimate the parameters that controls the mapping of a shape (model shape) onto another (target shape). The shapes of interest are contours in the 2D space, surfaces in the 3D space and point clouds (either in 2D and 3D spaces). We propose to model the shapes using Gaussian Mixture Models (GMMs) and estimate the transformation parameters by minimising a cost function based on the Euclidean ($\mathcal{L}_2$) distance between the target and model GMMs. This strategy allows us to avoid the need for the computation of one to one point correspondences that are required by state of the art approaches making them sensitive to both outliers and the choice of the starting guess in the algorithm used for optimisation.

Shapes are well represented by GMMs when careful consideration is given to the design of the covariance matrices. Compared to isotropic covariance matrices, we show how shape matching with $\mathcal{L}_2$ can be made more robust and accurate by using well chosen non isotropic ones. Our framework offers a novel extension to $\mathcal{L}_2$ based cost functions by allowing prior information about the parameters to be included. Our approach is therefore fully Bayesian.

This Bayesian-$\mathcal{L}_2$ framework is tested successfully for estimating the affine transformation between data sets, for fitting morphable models and fitting ellipses. Finally we show how to extend this framework to shapes defined in higher dimensional feature spaces in addition to the spatial domain.

Acknowledgments

This thesis would not have been possible without the help, support and patience of my principal supervisor Dr Rozenn Dahyot. She used to say that she was not my supervisor but just an advisor instead. But the truth is that she was my super-advisor. I really appreciate all the good advices, the encouragement she gave me along this process and above all her friendship. I would also like to express my gratitude to my co-supervisor, Professor Kurshid Ahmad for all his useful advice not only about the thesis but also about life in general. I would like to acknowledge to Trinity College Dublin and the Government of Chile for the financial support. Thanks to all GV2 group and in particular to Jonathan, Stefan, Mirko, Fintan and Ludovic for all their help. Special thanks to Ziggy for proofreading my thesis and most of my papers.

I had a wonderful time in Ireland while I was writing my doctoral thesis and I owe that to all the wonderful people I had around. I am very greatful for the company and friendship of Atul, Kerstin, Katja and Ziggy. Thank you guys for all the movie nights, the long chats, the laugh and the good times. Special thanks to Soraya, Eonmi, Christina and Venkat for being not only good friends but also my family while living in Dublin. My gratitude to my boyfriend Joe for all his support and patience.

Finally, I wish to thank my parents and my sister for their love and encouragement. Without them I would never have had so many opportunities.

CLAUDIA L. ARELLANO VIDAL

University of Dublin, Trinity College
April 2014

Contents

Abstract ix

Acknowledgments xi

List of Tables xvii

List of Figures xix

Chapter 1 Introduction 1

1.1 Overview and motivation 2

1.2 Current problems and scope of the present work 3

1.3 Summary of contributions 4

1.4 Thesis outline 5

1.5 List of publications 6

Chapter 2 Literature Review 8

2.1 Shape analysis 8

2.1.1 Shape description 8

2.1.2 Shape representation 10

2.2 Statistical inference theory 11

2.2.1 Statistical modelling 12

2.2.2 Likelihood based estimation 15

2.3 Shape inference applications 18

2.3.1 Registration & correspondences 18

2.3.2 3D face reconstruction 21

2.3.3 Morphable shape model 24

2.3.4 Ellipse fitting 25

2.4 Summary 27

I Method: Inference with Bayesian $\mathcal{L}_2$ **29**

Chapter 3 Inference with $\mathcal{L}_2$ and $\mathcal{L}_2$E **30**

3.1 Divergence between probability density functions 30

3.1.1 Closed form solution of $\mathcal{L}_2$ 31

3.1.2 Inference with $\mathcal{L}_2$ 32

3.1.3 $\mathcal{L}_2E$ approximation 32

3.2 $\mathcal{L}_2E$ parameter estimation 33

3.2.1 Sample size 33

3.2.2 Robustness to outliers 34

3.2.3 Estimation with fixed bandwidth 35

3.3 Point cloud registration with $\mathcal{L}_2E$ 36

3.4 Conclusion 40

Chapter 4 Bayesian $\mathcal{L}_2$ for Shape Inference **41**

4.1 Bayesian $\mathcal{L}_2$ 41

4.1.1 Application to affine registration 44

4.1.2 Application to morphable shape model fitting 44

4.1.3 Combining affine registration and morphable shape fitting 44

4.1.4 Application to ellipse fitting 45

4.2 Modelling curves and surfaces with GMMs 45

4.2.1 Using non-isotropic covariances 45

4.2.2 GMM from shapes in images 48

4.2.3 Role of h in Bayesian $\mathcal{L}_2$ shape inference 48

4.2.4 Complexity reduction for parsimonious representation 49

4.2.5 Optimising GMM from images 53

4.3 Extension to shape representation with GMM 55

4.4 Conclusion 55

II Experiments and Results **57**

Chapter 5 Affine Transformation Estimation with Bayesian-$\mathcal{L}_2$ **58**

5.1 Rigid parameter estimation 58

5.1.1 Using isotropic covariance 60

5.1.2 Using non-isotropic covariance 67

5.2 Scaling parameter estimation 70

5.2.1 Non informative prior 70

5.2.2 Gaussian Prior 72

5.3 3D reconstruction applications 72

5.3.1 Multi-view reconstruction . . . 74
5.3.2 Simultaneous localization and mapping . . . 75
5.4 Conclusion . . . 81

Chapter 6 Morphable Shape Fitting with Bayesian-$\mathcal{L}_2$ and Gaussian Prior **83**
6.1 2D morphable hand model fitting . . . 83
6.1.1 Role of the prior . . . 85
6.1.2 Using isotropic and non-isotropic covariance matrices . . . 86
6.2 3D morphable face model fitting . . . 88
6.2.1 Fitting 3D face model to synthetic data . . . 89
6.2.2 Fitting 3D face model to Kinect data . . . 91
6.3 Conclusion . . . 96

Chapter 7 Extensions of Bayesian-$\mathcal{L}_2$ framework **98**
7.1 Detecting multiple instances . . . 98
7.2 Application to ellipse detection . . . 101
7.2.1 Fitting one ellipse to noisy observations . . . 102
7.2.2 Detecting Multiples ellipses . . . 108
7.3 Conclusion . . . 113

Chapter 8 Conclusion **114**
8.1 Summary . . . 114
8.2 Limitations and future perspectives . . . 116

Appendix A Mathematical Expressions **118**
A.1 Closed form solution for $\mathcal{L}_2$ . . . 118
A.2 $\mathcal{L}_2$ and $\mathcal{L}_2$E using isotropic covariance matrices . . . 118

Appendix B Algorithms **120**
B.1 Mean Shift algorithm for rigid transformation . . . 120
B.2 Mean Shift algorithm for shape fitting . . . 122
B.3 Ellipse fitting . . . 124
B.4 Detecting ellipses in images . . . 124

Appendix C Additional Results **127**
C.1 Affine transformation . . . 127
C.1.1 Sensitivity to noise . . . 127
C.1.2 Robustness to outliers . . . 130
C.1.3 Scaling estimation . . . 130
C.2 Ellipse fitting . . . 140

C.3 Shape fitting 141

Bibliography **143**

List of Tables

3.1 Average estimates of the mean and standard deviation with their confidence intervals (100 runs) with varying proportions of outliers and sample size. 37

3.2 The global minimum depends on the selected fixed bandwidth σ. 37

5.1 Numerical assessment of the experiments displayed in Figure 5.1a. In this case we set both algorithms to use the same parameters. 60

5.2 Numerical assessment of the experiments displayed in Figure 5.1b. In this case we include the annealing strategy for improving convergence of the Mean Shift Algorithm. 61

5.3 Numerical assessment of the experiments displayed in Figure 5.1c. This case is similar to the previous one (case b) where the annealing strategy is included in the Mean Shift Algorithm. 61

5.4 Rate of convergence towards the ground truth when considering an error of $\pm 1^0$ 66

5.5 Run-time performance of the algorithm measured as the time it takes the algorithm to converge (in seconds). 67

5.6 Computational complexity of the algorithm. 67

5.7 A comparison of the convergence of isotropic versus non-isotropic modellings for drastically sub-sampled data sets. 70

5.8 The rate of convergence towards the ground truth when considering error of 5%. We evaluate the cases where: no changes are made in the data (original), data is perturbed with noise (levels: 0.01 and 0.03), data is perturbed with outliers, and occluded data. In all cases 100 experiments were run and the number of successes are reported in the table. 72

5.9 The rate of convergence towards the ground truth when considering an error of 10% and 5%. The bandwidth for this experiment was set to $h_{max} = 0.5$ and $h_{min} = 0.01$. 72

5.10 Paired difference test: The table shows the value for the mean ($\bar{d}$), standard deviation (σ_d), number of samples (n^o) and the t-value calculated as $t = \frac{\bar{d}-0}{\sigma_d/\sqrt{n}}$ 81

6.1 Comparison of the convergence rate between isotropic and non-isotropic modelling 87

6.2 Malahanobis distance $d_{i,j}$ between the estimated parameters of faces F1 to F6 (Figure 6.13). 95

B.1 Expressions for computing $L_{(j,s)}$ and $b_{(j)}$ 123

List of Figures

1.1 Computer generated image of a room full of chairs [1] 1

1.3 Example of the parameter estimation problems to be considered in this thesis. Figure a) shows two shapes (hand). The two shapes are aligned to each other and displayed in Figure b). Figure c) shows the fitting process performed where one of shape is deformed in order to represent the second shape as close as possible. 2

1.2 Examples of three shapes. Figure a) shows a parametric curve (ellipse). Figure b) shows a 2D contour model of a hand and Figure c) shows a surface representation of a human face (3D mesh). 2

1.4 Point to point correspondence between shapes 3

1.5 Thesis outline. 6

2.1 Structural shape representation. The shape is broken down in segments that are represented individually. In a) we reprinted a horse shape that has been represented using *curvature* and *orientation* reported in [2]. The structural description of a chromosome shape described in [2] is shown in b) and the primitives used are shown in c). 9

2.2 Global shape description. Figure in a) represents a shape contour that can be defined using the one dimensional function shown in b) $x(t) = \cos(2t)$. The representation of the character M in c) can be described using its edge image d). 10

2.3 Point Cloud description by sampling the edge image. 11

2.4 Point Cloud description by sampling the edge image. 11

2.5 3D mesh representation using different resolutions (increasing the number of faces) [3]. 11

2.6 Statistical inference using parametric modelling. In a) a set of observation is shown as blue dots. In b) the estimated Gaussian Mixture (contour) is plotted as a contour along with the observations. The density function is assumed to be a Gaussian Mixture with three components. The optimisation is performed using a EM algorithm. 13

2.7 This figure shows the two approaches in non-parametric modelling: Histograms and kernel density estimation. We model the density function given a set of $n = 30$ observations. In the top row we present the histogram of the data set when using a) 10 and b)20 bins. In the middle row the same data set is represented using a Gaussian kernel for each point in the data set and using a bandwidth of $h = 0.2$ and $h = 0.1$ (Figures c and d respectively). Finally the density estimation using the kernels in c) and d) are presented in e) and f) respectively. 14
2.8 Scheme of SOM [4] 17
2.9 Example of Self Organizing map for diferents iterations. The dots are the set of observations. The grid is a set of connected nodes that are automatically organised to represent the observations [4] 18
2.10 Point to point correspondence between data sets. [5] 19
2.11 The shape correspondence concept of the 3D Morphable Model: The figure illustrates how vertices are indexed in the parametrized domain. For instance, according to Levine et al. [6], the tip of the nose will be assigned in the same position (k) in the vector shape s_i in every scan, even though the coordinate x, y and z that represent its position will change among individuals. 24

3.1 Example: a) Normal distribution $g(\mathbf{x}|\Theta = (\mu = 2, \sigma = 2))$. Figure b) shows a set of observations sampled from g (black dots) and Figure c) shows the empirical distribution $\hat{f}$ associated with the observations. The empirical distribution is represented by a delta Dirac function centred in each observation. 33
3.2 Results obtained when running 100 experiments for samples sizes of $5, 10, 50, 100, 1000$ and 10000 reported on the abscissa. The figure shows the average error between the ground truth and the estimates with a 95% confidence interval for the mean in a) and standard deviation in b). In both cases the error decreased when increasing the number of samples n_f. 34
3.3 The figure shows a set of samples taken from the true density function $g(\mathbf{x}|\Theta = (\mu = 2, \sigma = 2))$, and a few outliers drawn from a different density function (in red dash line). 35
3.4 Average error (between ground truth and estimates) with 95% confidence intervals with sample size $n_f = 5$ (top row) and $n_f = 10000$ (bottom row). 36
3.5 Estimates (100 experiments) with mean $\hat{\mu}$ reported in absissa and $\hat{\sigma}$ on the y-axis (0% outliers). The ground truth $(\mu, \sigma) = (2, 2)$ is plotted in black, estimates with $n_f = 100$ (blue) and estimates with $n_f = 10000$ (red). 38

3.6 Estimates (100 experiments) for $n_f = 10000$ with mean $\hat{\mu}$ reported in abscissa and $\hat{\sigma}$ on the y-axis. The ground truth $(\mu, \sigma) = (2, 2)$ is plotted in black. The estimates without outliers are shown in red, the estimated with 20% of outliers are shown in purple and the green one correspond to the estimated using 50% of outliers. 38

3.7 Cost function ($\mathcal{L}_2 E(\mu)$) for a sample size $n_f = 5$ with two outliers. The red line corresponds to $\sigma = 1$, blue dashed ($\sigma = 1.5$), green dots ($\sigma = 3$) and black dash-dot ($\sigma = 5$). The figure shows the impact of the bandwidth on the estimation of μ using the $\mathcal{L}_2 E$ approximation. 39

4.1 Modelling when using isotropic covariance matrices. 46

4.2 Effects of choosing a) proportional weight for the Gaussians instead of b) uniform weights. The ridge in a) has a similar value along the shape while in b) the ridge changes value along the shape. 47

4.3 Scheme of the principal directions of the covariance matrix extracted from a given 3D mesh 48

4.4 Reduction of the number of Gaussians in the mixture to represent the shape. 50

4.5 Similarity between shapes (Euclidean distance) when modelling the GM using non-isotropic covariance matrices and different numbers of Gaussians in the mixture. 51

4.7 Example of a portion of the mesh of a face when described using a different number of vertices. As more vertices are used (left) a more accurate representation is produced. 51

4.6 Density functions computed from the hand model when using non-isotropic Gaussians (with $n = 71$, $n = 56$, $n = 39$ and $n = 31$ kernels from top to bottom respectively). 52

4.8 Example of the meshing process using SOM. Figure a) shows the original 3D point cloud (blue dots) and the starting position of the mesh (red). Figures b) and c) show the final solution for different numbers of vertices (100 and 400, respectively). 53

4.9 Example of reduction of data by sub-sampling. Figure a) shows the original point cloud extracted from the edge map of Figure 4.11 a). Figure b) shows the data when sub-sampling the image while in c) when sub-sampling the connected edges. 54

4.10 2D view of the density function computed for Figure 4.11a. In the first column we show the density functions when using all data (764 points) for bandwidths of 7, 4 and 1 respectively. The second column shows the density functions of the observations with a resolution of 70% and in the last column with resolution of 50%. The number of Gaussians used to define the density functions are 526 and 372 respectively. In the three cases, the rows shows the 2D views when different bandwidths are used. At the top $h = 7$, in the middle row $h = 4$ and in the bottom row $h = 1$. . 54

4.11 The set of observations are extracted from the image a). We use the edge map and the gradient of the image to compute the position and curvature of the points of interest. In b) we show the map of the normal vector associated to the edge map. In c) we show the computed observation $\{(u_i, \psi_i)\}_{i=1\cdots,n} \in \mathbb{R}^3$. 56

5.1 2D data sets: alignment obtained when testing the 2D data sets Fish, Contour and a Chinese Character: Reference point set (blue circles), observation (red asterisk) and estimated solution (green cross). The top row of the figure shows results obtained using Jian and Vermuri's algorithm while in the bottom row we show the convergence obtained using our proposed MS algorithm. 61

5.2 Examples of the experiment performed for an angle of rotation of $\phi_{GT} = 30^o$. Results are displayed for our proposed algorithm, KC registration and ICP registration. 63

5.3 Convergence rate $(\%)$ of the estimated solution (y-axis) when perturbing the data with different noise levels (x-axis) using our proposed algorithm (red line) and when using the Kernel Correlation algorithm (blue dot-dash line). In a) we show the results for a rotation angle of $\phi_{GT} = 20^o$ and in b) $\phi_{GT} = 30^o$ 64

5.4 Error mean with 95% of confidence of the estimated rotation when data is perturbed using different levels of noise (from 0 to 0.2). In a) we show the target angle or rotation is $\phi_{GT} = 20^o$ and in b) $\phi_{GT} = 30^o$ 64

5.5 Examples when both sets are rotated by 20^o. 65

5.6 Examples when both sets are rotated by 90^o. 65

5.7 The error obtained for 100 experiments when using our proposed algorithm (red line) and the KC algorithm (blue dots). The mean of the error for the 100 runs when using our proposed algorithm is 0.0186 (degrees) and the standard deviation is 0.1105. In the case of the KC algorithm the mean and standard deviation are 0.5024 and 0.4842 respectively. 66

5.8 Example of a set of observations a) and model of the hand b) used for estimating the rigid transformation when modelling GMM using non-isotropic covariance matrices. In c) we show a sub-sampled version of the hand model. 68

5.9 Cost function computed when both GMMs are modelled using isotropic covariance matrices (blue dash line) and when one of them is modelled using non-isotropic modelling (red line). 69

5.10 An example of results obtained when sub-sampling the data sets and when using isotropic (green dash line) and non-isotropic modelling (red line). The blue dots correspond to the full observation. 70

5.11 Examples of the data set used for testing 71

5.12 Examples of the results obtained when testing our algorithm under different conditions. 73

5.13 Experiment setup. The object to be scanned is placed as close as possible to the Kinect sensor (approximately 50 cm). The object (here a head) is rotated in order to capture different points of view. 74

5.14 Results obtained when aligning different views captured using the Kinect sensor. The noise is reduced as the number of views n merged increase. A laser scanner acquisition is also shown for visual comparison purposes . 75

5.15 Kinect scan alignment for a face. The top row shows a sequence of images captured from different points of view. Bottom row: Mesh created from a) a single acquisition, b) multiples views and c) face captured using a laser scanner. 75

5.16 a) the picture of the object (Duck) and d) laser scan respectively (ground truth). In b) the reconstruction with one acquisition using the Kinect sensor and e) its error w.r.t. the laser scan has an average of 2.25mm. When merging 15 views together c) the average error f) is reduced to 1.54mm. Details of the Gnome face the average error of the reconstructed shape when four images are merged is 1.7mm. 76

5.17 Two scans displayed from different viewpoints 77

5.18 A set of examples of the alignment between scans. (a) two consecutive scans (input data), (b) the two scans after removing the ground, (c) the two scans aligned according to the transformation parameters estimated with our proposed algorithm and (d) is the estimated transformation applied to the full scan. 78

5.19 A set of scans aligned together. The figure on the left shows a set of 6 consecutive scans. The middle figure shows the results after estimating the transformation parameters between the scans. The same result from another point of view is displayed in the right figure 79

5.20 The distance between the shape when aligned using the proposed method (blue line) and when aligned using the ground truth (red line). Results obtained when modelling the density function using isotropic bandwidth $h = 7, 14, 21$ and 35. 79

5.21 Details of the alignment between buildings. a) using our proposed algorithm for aligning the scans and b) using ground truth. 81

6.1 Overview of the shape estimation process using a shape model as prior information. The observation as a point cloud $\{u_i\}_{i=1,\cdots,n}$ in b) is obtained after preprocessing the image a) captured by a 2D camera sensor. The parameter estimation is then divided into two steps: c) shape alignment $\Theta = [\mathrm{s}, \mathrm{R}, \mathrm{t}]$ and d) shape fitting $\Theta = [\alpha]$. 84

6.2 Results obtained when using synthetic hands generated from the model a) and when adding random noise b). Figures c) and d) show the estimated shape when the prior information is not considered in the algorithm. In all figures: observations (blue dots), initial guess (green dash) and estimated solution (red line). Setting: $h_{max} = 50$, $h_{min} = 5$. 85

6.3 Hand model fitting to a point cloud obtained from 2D images. Column a) and b) show the colour image and edge map used for the experiments. In column c) the estimated hands (red dash) and in d) the solution when the prior is not used in the algorithm. For the two experiments we compute the Euclidean distance $\mathbf{d}$ between the model and the observations. When using the prior information the algorithm minimises the Euclidean distance better (results in c) than when the prior is not used d). 86

6.4 Examples of observations when randomly sub-sampling the original data set. 87

6.5 Examples of results obtained when using non-isotropic covariance matrices for the GMM that represent the hand model. Column a) shows the data used as observations (100 samples from the original data set). Column b) and c) show the results obtained (red line) superimposed with the observation used b). Results using the full data set are displayed in c). . . 88

6.6 Examples of results obtained when reducing the number of Gaussians for the shape model (n_g). Top row shows results of the estimation when using non-isotropic covariance matrices. The bottom row shows results when using isotropic covariance matrices. In all examples, the red line corresponds to the estimate while the blue dots to the observations. 89
6.7 Fitting synthetic faces: target face (observations, top row), random starting guess for initialising the algorithm (middle row) and estimated face with our algorithm (bottom row). 90
6.8 Surface error computed between the target face and our reconstruction. . . 90
6.9 Preprocessing for generating the points cloud of the face (target): depth map (a) as captured by the RGB-D sensor, selection of the scene (b) in close range (between $0.5m$ and $1.2m$ from the sensor), extracted face and skin region (c). 91
6.10 Shape alignment: different views of the point clouds (model (grey) and observations (red)) after alignment (top row). Same point clouds displayed before alignment (bottom row). 92
6.11 Histogram of the errors between the observations to the average shape: before alignment (blue) and after (red). The sum of the absolute error is 6.3456×10^6 (after alignment) compared with 1.7054×10^7 (before alignment). 93
6.12 Average shape used as initial guess. 93
6.13 Estimated reconstructed faces (labelled F1 (top), to F6 (bottom)) from 3 viewpoints shown with the colour image captured with the Kinect (left). . 94
6.14 At the top, profile view of F5: laser scan (left), reconstruction (middle) and Kinect point cloud (right). At the bottom, frontal view of F5: reconstruction (left) and laser scan (right). 95
6.15 Histograms of the errors between the observations and the reconstructed face (red line) and the observations and the average shape of the model (blue dash) for F1 (left) and F2 (right). 95
6.16 Percentage of observations below a distance between 1 and 5mm (reported on the absissa) for F1 (left) and F2 (right), before fitting (blue dash) and after fitting (red line). 96
6.17 Values of the $J = 20$ coordinates of parameter α normalised with the eigenvalues computed for faces F1 to F6. 96

7.1 Example of detection of multiples instance (i.e coins). 98
7.2 Illustration of the iteration of the proposed method for detecting multiple instances of a shape. 100
7.3 Normalised probability of each point of being part of the detected shape. . 101

7.4 Example of the GMM of the ellipse computed using different bandwidths. 103

7.5 Figure shows three examples of observation used in the experiments. The ellipses were computed using the target ellipse for which the parameters are $\Theta = [\gamma, a, b, x_o, y_o] = [0, 3, 5, 0, 0]$ corrupted using Gaussian noise with standard deviation σ^2 equal to 0.1, 0.3 and 0.5 respectively 103

7.6 Example of computing the error rate. Figure a) shows in blue the true ellipse (observation) and in red dots the estimated one. The black region in figure b) represents the area of the symmetric difference in between the two ellipses. 104

7.7 Mean Error obtained for the data sets perturbed with 5 different standard deviation of noise (σ from 0.1 to 0.5). In a) we compare the proposed method (green star) versus a recent robust ellipse fitting algorithm reported by Yu et al. [7] (blue square). In b) our proposed method (green start) is compared with two standard ellipse fitting algorithms. The red triangle represents the algorithm which cost function minimises the algebraic distance while the blue dot is the well known Least Square method that is based on the minimisation of a geometric distance 105

7.8 Figure shows a detail of the error of each method including its 95% of confidence interval. 105

7.9 Results of the mean error compared with standard fitting ellipse methods for 5 level of noise. This experiment corresponds to the case where 10% of extra points randomly distributed are added to the observation. The error computed when using our proposed method (green start) is compared with two standard ellipse fitting algorithms. The red triangle represents the algorithm which cost function minimises the algebraic distance while the blue dot is the well known Least Square method that is based on the minimisation of a geometric distance [8, 9]. 106

7.11 Results compared two ellipses. For our method we run a trial of 50 examples obtaining a mean error of 0.0133 with a standard deviation of 0.0012. When perturbed using $\sigma = 0.1$. In all the figures the red dots correspond to the observation. In a) we show the results obtained using the direct fit algorithm, in b) Least Square method and in c) our proposed method. 106

7.10 Examples of the experiments performed. For all cases, the red dot is the observation. The estimated ellipse for the Least square method (Blue), Direct fit (black) and the proposed method (green). 107

7.12 This Figure shows an example of the data set used for testing the algorithm. This data set was published by [10] and it contains 6 sets of 50 images. Each set contains a different number of occluded ellipses from 4 to 24. In the first and third row the original images are displayed while in the second and forth the observations extracted are shown as a map of normal vector corresponding to the edge points in the original image. . . . 109
7.13 Example of the results obtained when applying our proposed algorithm for detecting multiples ellipses. On the top row the original images and at the bottom the detected ellipses (red) and the observations (blue). . . . 110
7.14 Example of the results obtained when applying our algorithm to data sets containing 4,8,12,16,20 and 24 occluded ellipses respectively. 111
7.15 Testing on synthetic images containing occluded images. In Figure a) we report the values obtained for the Recall while in b) the results for precision. Each set (from 4 to 24) was evaluated using 50 images. For comparison we report the results obtained using the approaches proposed by Chia et al.[10], Mai et al. [11], Kim et al [12] and the Hough transform based methods RHT and SHT proposed by [13] and [14] respectively. . . 112
7.16 Testing our algorithm on 50 synthetic images each containing four 4 ellipses occluded as overlap error is varied from 0.05 to 0.55. 112
7.17 Examples of detecting ellipses in RGB images. 112

C.1 Example 1 of results obtained for noise level 1 and angle of rotation 30^o . 127
C.2 Example 2 of results obtained for noise level 5 and angle of rotation 30^o . 128
C.3 Example 3 of results obtained for noise level 1 and angle of rotation 30^o . 128
C.4 Example 4 of results obtained for noise level 5 and angle of rotation 30^o . 129
C.5 Example 5 of results obtained for noise level 1 and angle of rotation 90^o . 129
C.6 Example 6 of results obtained when both sets are rotated in 45^o 130
C.7 Example 1 scaling 130
C.8 Example 2 scaling 131
C.9 Example 3 scaling 131
C.10 Example 4 scaling 131
C.11 Top: Results for s_1, estimated (blue line) and ground truth (red square). Bottom shows the results for s_2. The experiments where performed using the same data sets scaled using a normal distribution for both parameters. We run 100 experiments and all of them converge to the ground truth. . . 132
C.12 Top: Results for s_1, estimated (blue line) and ground truth(red square). Bottom shows the results for s_2. The experiments where performed using the same data set but adding noise. The scaling is performed using a normal distribution for both parameters. 133

C.13 Example scaling with outliers in the data set 134
C.14 Example 1 scaling with outliers in the data set 134
C.15 Example 2 scaling with occlusion in the data set 134
C.16 Example 3 scaling with occlusion in the data set 135
C.17 Top: Results for s_1, estimated (blue line) and ground truth(red square). Bottom shows the results for s_2. The experiments where performed using the occluded data set which was scaled using a normal distribution for both parameters. 136
C.18 Example 4 scaling with noisy data set 137
C.19 Example 5 scaling with noisy data set 137
C.20 Example 6 scaling with noisy data set 137
C.21 Example 7 scaling with noisy data set (S_1) 138
C.23 Example 1 Occlusion (25%) and Outliers (20%) Examples (using prior) . 138
C.24 Example 2 Occlusion (25%) and Outliers (20%) Examples (using prior) . 138
C.22 Example 8 scaling with noisy data set (S_2) 139
C.25 Example 3 Occlusion (25%) and Outliers (20%) Results for S_1 and S_2 (with prior) 139
C.26 Results of the parameters estimated using our proposed method. 140
C.27 Estimated shapes (in red solid line) using the isotropic shape model (a) and the non-isotropic shape model (b). The observations are shown as blue dots. 141
C.28 The green line correspond to the non-isotropic model and the blue dots when using the isotropic model. The abscissa corresponds to number of kernels used in the model. 141

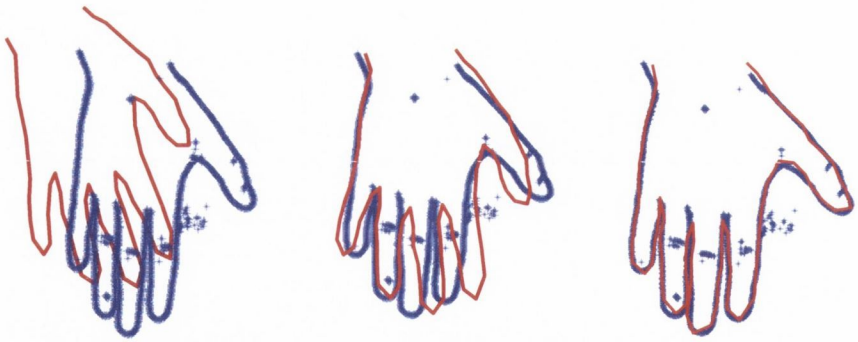
Chapter 1

Introduction

Human vision is a powerful perceptual system that allows us to understand the world. This process of understanding is usually based on the classification and identification of objects we can see. One of the attributes that most distinguishes an object from its surroundings is its shape. It is well known and commonly accepted that the mechanism underlying the human perception of shapes is innate. This innate human ability is so powerful that objects that has been rotated, scaled or even occluded can be still be recognised. A well known example is shown in Figure 1.1 where a room full of chairs is displayed. There is no difficulty for humans to recognise the chair partially hidden behind the desk or the small rotated version hanging from the ceiling.

Figure 1.1: Computer generated image of a room full of chairs [1]

In order to replicate the ability to see from a computer's point of view it is necessary to define the concept of shape from a mathematical and digital perspective. Once the description of a reference shape is provided, any new shape can be identified by comparison. This comparison process should take into account all the transformations or deformations that the new shape may be subject to. Those transformations can be mathematically modelled by a set of parameters. In this thesis we address the problem of



a) b) c)

Figure 1.3: Example of the parameter estimation problems to be considered in this thesis. Figure a) shows two shapes (hand). The two shapes are aligned to each other and displayed in Figure b). Figure c) shows the fitting process performed where one of shape is deformed in order to represent the second shape as close as possible.

estimating those transformation parameters between shapes in order to achieve methods for detecting and reconstructing them. In this thesis the focus will be on shapes that can be represented either using a parametric expression or by a set of points (i.e point cloud, meshes etc.). Examples of shapes are shown in Figure 1.2.

a) Ellipse b) Hand c) Face

Figure 1.2: Examples of three shapes. Figure a) shows a parametric curve (ellipse). Figure b) shows a 2D contour model of a hand and Figure c) shows a surface representation of a human face (3D mesh).

1.1 Overview and motivation

Estimating the parameters of a shape is a problem that arises in different fields in computer vision. The problem is usually classified according to the parameters to be estimated, the knowledge or information about the shape and the kind of observation that is being dealt with. Two general categories related to shape parameter estimation found in the literature and that we will study are *shape alignment* and *shape fitting*.

Shape Alignment: The alignment process consists of estimating the affine transformation (rotation, translation and scaling) between the two shapes (cf. Figure 1.3b). This

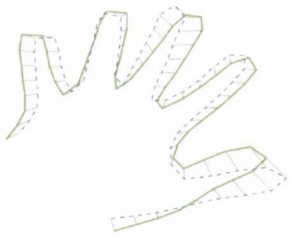


Figure 1.4: Point to point correspondence between shapes

process is commonly referred to as shape registration. Registration of shapes usually involves a second process where point correspondence between the two shapes needs to be computed (cf. Figure 1.4). The correspondence between shapes is not a trivial problem and usually introduces error when estimating parameters, in particular when the two shapes are not defined using the same description.

Shape Fitting: A statistical shape model is deformed in order to fit a set of observations (cf. Figure 1.3c). Many state of the art methods solving this problem require the model and the observation to be aligned. Additionally, the correspondences between the two shapes need to be established. The need for correspondences imply that the fitting algorithm depends directly on the previous step: registration. Any error in the correspondences is directly reflected in the parameters estimated during the fitting process. This makes fitting algorithms very sensitive, especially to outliers and noise in the data.

1.2 Current problems and scope of the present work

Shape fitting algorithms still depend on the accuracy of the correspondences. Indeed, errors in the correspondence in between the model and the observations mislead the fitting process. This implies the need for a registration algorithm that achieves better results. Furthermore, the results need to be accurate even when outliers, noise and occlusions are presents.

More recent registration algorithms based on the idea of distance between density functions offer more promising alternatives for robust registration. However, little effort has been done yet in terms of modelling density functions tailored for representing shapes. Moreover, robust methods are often more computationally intensive. Therefore particular attention is also required for proposing density functions representing well and efficiently the shapes to limit computations.

The goal is then to achieve a method for estimating shape parameters that could meet the following criteria:

- Robust to outliers
- Achieve an accurate estimation of the parameters

- Avoid point-to-point correspondence in order to be able to compare shapes that have been sampled at different rates or acquired using different sensors.

- Have a computationally efficient algorithm

The question here is how to balance all those requirements by modelling suitable cost functions to optimise. We study the robust inference for shape parameters estimation. We explore, in particular, methods based on the $\mathcal{L}_2$ distance. We focus on the modelling of the shape as a density function. Moreover, we use Gaussian Mixture Models and analyse the role of the covariance matrices for achieving a robust, accurate and efficient estimation.

1.3 Summary of contributions

We propose in this thesis to model shapes as probability density functions and to use the concept of divergence as a measure of similarity between them. In particular, we consider two Gaussian Mixture Models f and g defined for two shapes. The transformation parameters between these shapes can then be estimated using the Euclidean distance between probability density functions also known as $\mathcal{L}_2$. The key contributions reported in this thesis are:

1. We show that the robustness while estimating parameters using the $\mathcal{L}_2$ metric depends directly on the modelling of the density functions (f and g). Furthermore, the covariance associated to the density function modelled as GMM affects the convergence of the optimisation algorithm and the accuracy of the estimation. This leads to the need for modelling better suited GMM when representing shapes and estimating their parameters (cf. Chapter 3).

2. We propose to define GMM for representing shapes that use non-isotropic covariance matrices based on its geometry. We demonstrate that this modelling improves the estimation results and it helps in achieving a more efficient optimisation algorithm for estimating parameters (cf. Chapter 4).

3. We propose a Bayesian framework based on the $\mathcal{L}_2$ metric. This framework allow us to include prior information about the parameters to estimate (cf. Chapter 4).

4. We explore the Bayesian-$\mathcal{L}_2$ framework proposed and tackle challenging problems in computer vision such as the affine transformation (cf. Chapter 5), shape fitting (cf. Chapter 6) and parametric curve estimation (cf. Chapter 7). The contribution of these three Chapters is summarised as follows:

(a) In Chapter 5 we propose a method for estimating the affine transformation between data sets. A dedicated Mean Shift algorithm is implemented for solving the optimisation problem when the bandwidths of the GMM are assumed isotropic. When non-isotropic bandwidth are used a Newton algorithm is proposed.

(b) In Chapter 6 a method for estimating the parameters of the morphable model that best fits a set of observations is proposed. A dedicated Mean Shift algorithm is used for solving the optimisation. The main advantage of this method is that no correspondence is needed between data sets as in most shape fitting algorithms.

(c) Chapter 7 presents a method for detecting multiples instance of an ellipse. The parameters of the ellipse are estimated using the Bayesian framework proposed previously.

1.4 Thesis outline

The work carried out in this thesis is structured in seven chapters (cf. Figure 1.5). Chapter 2 summarises the state of the art in shape parameter estimation. It contains a brief overview of shape representation and statistical inference. Additionally, the most relevant algorithms for shape registration and fitting are reported along with a discussion about the problems and challenges still remaining in the field. The following five chapters contain the contribution of this thesis (Chapter 3 to Chapter 7). They are classified in two parts. Part I includes Chapters 3 and 4. Here, we propose a new framework for estimating parameters. We first introduce the $\mathcal{L}_2$ metric and discuss its advantages for robust estimation. The $\mathcal{L}_2$ metric is a measure of similarity between density functions. We show that its robustness depends on the modelling of these density functions (Chapter 3). In Chapter 4 we propose to include prior information when using the $\mathcal{L}_2$ metric. We define a Bayesian framework where the data term is based on the $\mathcal{L}_2$ distance. We propose ways of modelling shapes as density functions and study the role of the covariance matrices in the robustness and accuracy of the estimation process. In Part II of this thesis we report the experiments and results obtained when using our proposed method to solve challenging computer vision problems. We classify the experiments in three Chapters depending on the shape parameters to be estimated. In Chapter 5 we address the affine transformation between data sets. In Chapter 6 we analyse the performance of the proposed Bayesian-$\mathcal{L}_2$ framework for fitting morphable models. We assume in this chapter that a statistical shape model of the shape of interest is available. The usual limitations given by correspondence are overcome making the algorithm suitable even for data sets sub-sampled at different rates or with significant occlusion. In Chapter 7 we explore the

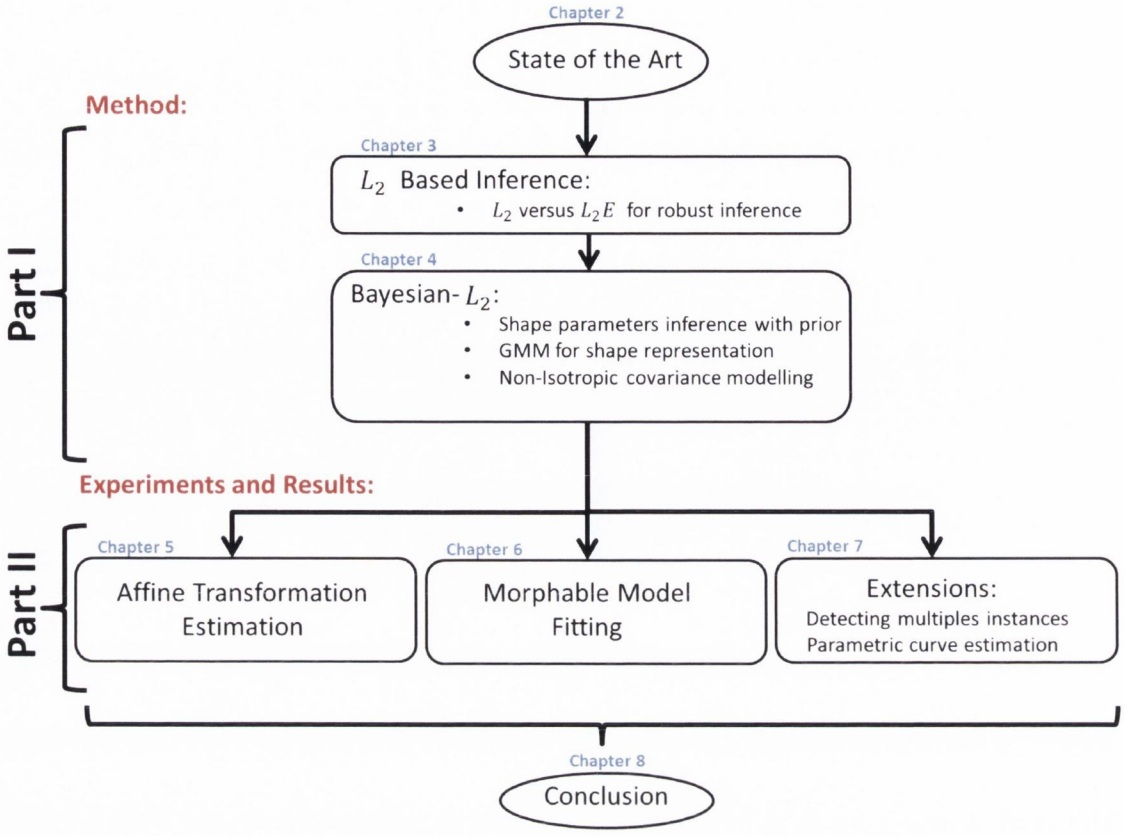


Figure 1.5: Thesis outline.

advantages of modelling shapes as density functions and its suitability for including extra information about the shape. We explore the use of multidimensional density functions that model the contour of the shape as well as its curvature. We test this method when detecting curves (ellipses). In addition, we propose a method for estimating multiple instances of a shape from a class of interest. Finally the conclusion of the work carried out in this thesis is summarised in Chapter 8. All the details of the algorithms proposed and the mathematical expressions used in this thesis are reported in the Appendix A and B respectively. Additional results of the experiments performed can be found in the Appendix C.

1.5 List of publications

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

1. J. Ruttle, C. Arellano and R. Dahyot, Robust Shape from Depth Images with GR2T. in Press Pattern Recognition Letters. January 2014.

2. C. Arellano and R. Dahyot, Robust Bayesian Fitting of 3D Morphable Model, Conference on Visual Media Production (CVMP), London, November 2013

3. C. Arellano and R. Dahyot, Shape Model Fitting Algorithm without Point Correspondence, 20th European Signal Processing Conference (Eusipco), Bucharest Romania, August 2012.

4. C. Arellano and R. Dahyot. Mean Shift Algorithm for Robust Rigid Registration Between Gaussian Mixtures Models, 20th European Signal Processing Conference (Eusipco), Bucharest Romania, August 2012.

5. J. Ruttle, C. Arellano and R. Dahyot. Extrinsic Camera Parameters Estimation for Shape-from-Depths, 20th European Signal Processing Conference (Eusipco), Bucharest Romania, August 2012

6. C. Arellano and R. Dahyot. Shape Model Fitting Using non-Isotropic GMM , 23nd IET Irish Signals and Systems Conference, Maynooth Ireland, June 2012

7. C. Arellano and Rozenn Dahyot, Stereo Images for 3D Face Applications: A Literature Review. International Machine Vision and Image Processing conference (IMVIP). ISBN 1-4438-2962-5, Limerick Ireland, Sept. 2010.

Chapter 2

Literature Review

In this chapter we present an overview of shape analysis and its uses in computer vision applications. We first define different approaches for representing shapes in a digital context (cf. Section 2.1). Following this, in Section 2.2, we present a brief review of the basic concepts of statistical inference and density function modelling. Statistical inference plays a key role in shape parameter estimation. In particular for applications such as shape detection, classification and recognition. These applications and the most relevant algorithms related to these problems are reviewed in Section 2.3.

2.1 Shape analysis

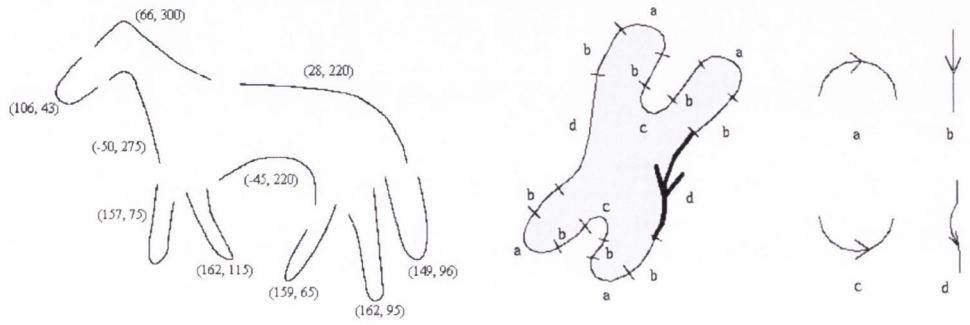
Any shape can be mathematically defined as the trajectory of a point in movement $x(t)$. with t the time and x the function describing the trajectory. Furthermore, in a computer vision context, it can also be defined as all the geometrical information that remains when location, scale and rotational effects are filtered out of an object [15]. Methods for describing and representing shapes will be discussed in the following sections. Those methods can be categorized depending on from where the structures that describe the shape are extracted. Two groups are recognized: *contour-based* methods and *region-based* methods. A full review of those methods can be found in [16]. In this thesis we focus only on contour based methods.

2.1.1 Shape description

Contour-based methods for describing shapes are commonly classified as structural (discrete) or global (continuous) approaches.

Structural Approaches: The structural approach breaks down the shape into boundary segments called primitives. Several primitives have been reported in literature such as

polygonal approximation, curvature decomposition and curve fitting [16]. Two examples are shown in Figure 2.1. In (a) the horse shape have been decomposed using the *smooth curvature* technique [2]. The information contained in each primitive corresponds to the maximum curvature and orientation of the segment to be considered. In (b) the primitives are described using a *Syntactic Analysis*. This approach is inspired by the composition of language where sentences are built from phrases, phrases from words and words from alphabets. Following this idea the shape is represented by a set of predefined primitives as shown in Figure 2.1c.

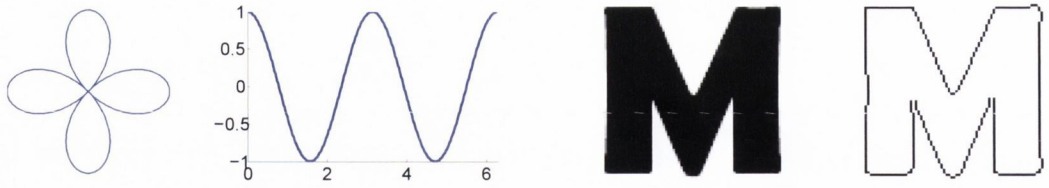


a) b) c)

Figure 2.1: Structural shape representation. The shape is broken down in segments that are represented individually. In a) we reprinted a horse shape that has been represented using *curvature* and *orientation* reported in [2]. The structural description of a chromosome shape described in [2] is shown in b) and the primitives used are shown in c).

The main problem with structural methods is the generation of primitives. There is no formal definition about the number of primitives required or the kind of primitives necessary to properly represent the shape. A second drawback relies on its nature as a local representation of the shape. The lack of a global definition of the shape or a topological integration between those segments affects the proper description of the shape. The capture of a global shape feature is equally important for a proper representation.

Global approaches: Global approaches do not break the contour shape into subparts. Instead, the whole boundary is used to describe the shape. For instance, a mathematical description of the shape as the function $x(t)$ in 2.2 is considered as a global method. Unfortunately, such description is usually not available for most of the shapes. However, it inspires the concept of *shape signature*, as a function that can be derived from the shape boundary points. Some of the exiting shape signatures include information such as centroid profile, complex coordinates, centroid distance, tangent angle and curvature among others. Shape descriptors are also common global methods for shape representation. The simplest descriptors known as shape factors are: circularity (perimeter^2/area), eccentricity, major axis orientation and bending energy. However, they can usually only describe



a) b) c) d)

Figure 2.2: Global shape description. Figure in a) represents a shape contour that can be defined using the one dimensional function shown in b) $x(t) = \cos(2t)$. The representation of the character M in c) can be described using its edge image d).

the general characteristics of the shape. For a more detailed description it is necessary to include more distinctive shape features such as shape area, edges, corners or ridges (cf. Figure 2.2 c and d). Shape context, for instance, is an example of a feature descriptor. It is based on the selection of a set of points taken from the contour of the shape. The key idea is to use the distribution of these points over relative positions as robust, compact, and highly discriminative descriptor. Shape context has been used as a method for estimating corespondence between shapes and for shape matching applications [17]. Descriptors are not only defined in the shape space but also in different domains such the Fourier descriptors [18, 19, 20] or wavelets descriptors [21, 22].

2.1.2 Shape representation

To evaluate and manipulate shapes in a digital context it is necessary to look not only at how accurate the shape is described but also how this description is represented.

String representation: Structural approaches for shape description are usually represented by encoding the information into a string $S = s_1, s_2, s_3, ..., s_n$. Each element s_i corresponds to specific information (primitives) from the shape. For instance, the shape of the chromosome shown in Figure 2.1 b) can be represented as a grammatical string using the predefined letter of each segment as: $S = dbabcbabdbabcbab$.

Point Cloud and Mesh Representation: Common shape representation for global methods are point clouds and meshes. The point cloud corresponds to a set of points, usually specified by a 3-tuple, [x, y, z] defined on an orthogonal coordinate system. In the case of 2D the points are usually extracted by sampling the edges of an image (cf. Figure 2.3). Meshes on the other hand, correspond to a structured representation of the point cloud. It is defined by a topology given by fitting polygons (i.e triangles) to the points. The mesh is then described as a set of vertex, edge, normal and faces (cf. Figures 2.4 and 2.5).

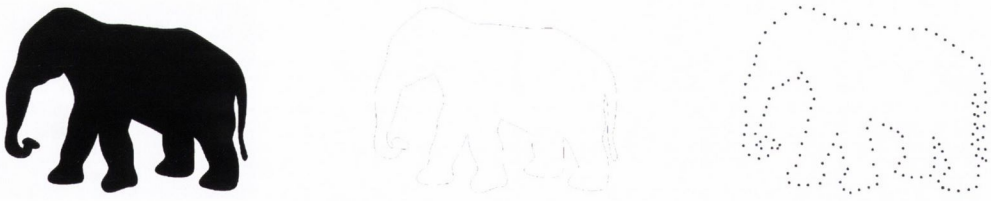


Figure 2.3: Point Cloud description by sampling the edge image.

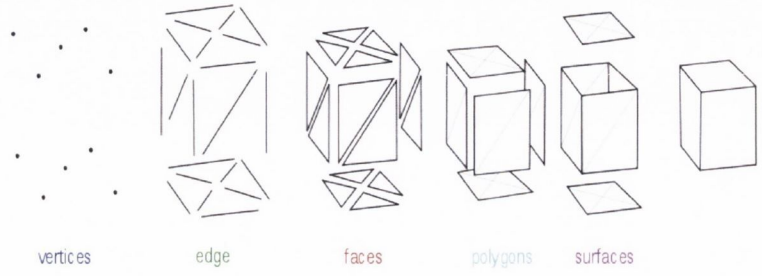


Figure 2.4: Point Cloud description by sampling the edge image.

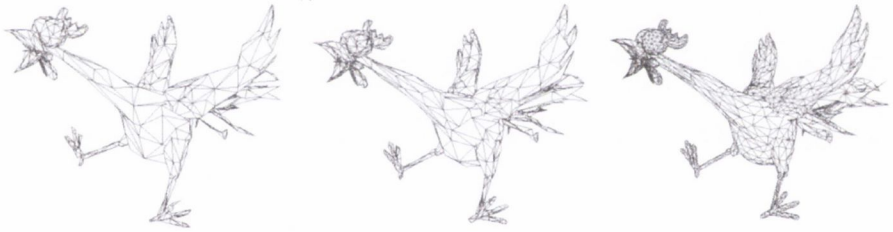


Figure 2.5: 3D mesh representation using different resolutions (increasing the number of faces) [3].

2.2 Statistical inference theory

Probability theory plays an important role in the solution of pattern recognition and computer vision problems. It provides a framework for inferring information from a set of observations collected for a specific variable. The statistical inference relies on the modelling of a probability density function that encodes the properties of the variable we are trying to analyse. The parameters of the density function are inferred according to the observations collected. Since this is a statistical modelling of the variable of interest the error and uncertainty are explicitly acknowledged [23, 24]. We present in Section 2.2.1 two approaches for modelling statistical inference problems. In addition, we review the well known EM algorithm and its extension as a self-organising map algorithm (cf. Section 2.2.2).

2.2.1 Statistical modelling

The modelling of an inference problem can be categorized in two groups: Parametric and Non-parametric modelling.

Parametric modelling: In parametric modelling, the distribution of the density function is assumed to be known and governed by a set of parameters (such as mean and variance for instance). As a result the problem is to infer the values of those parameters that define the probability density function. In other words, given a set of n observation $\{\mathbf{x}_k\}_{k=1,\cdots,n}$ of the random variable $\mathbf{x}$ and assuming its underlying probability density function as $p(x|\Theta)$. The problem is then to infer the parameters Θ that best represent $p(x|\Theta)$ given the observations. A simple example that illustrates the concept of inference using parametric models is shown in Figure 2.6.

When prior information about the parameters to estimate are available, the problem can be solved in a Bayesian framework as follows:

$$p(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n) \propto L(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)\, p(\Theta) \tag{2.1}$$

Where,

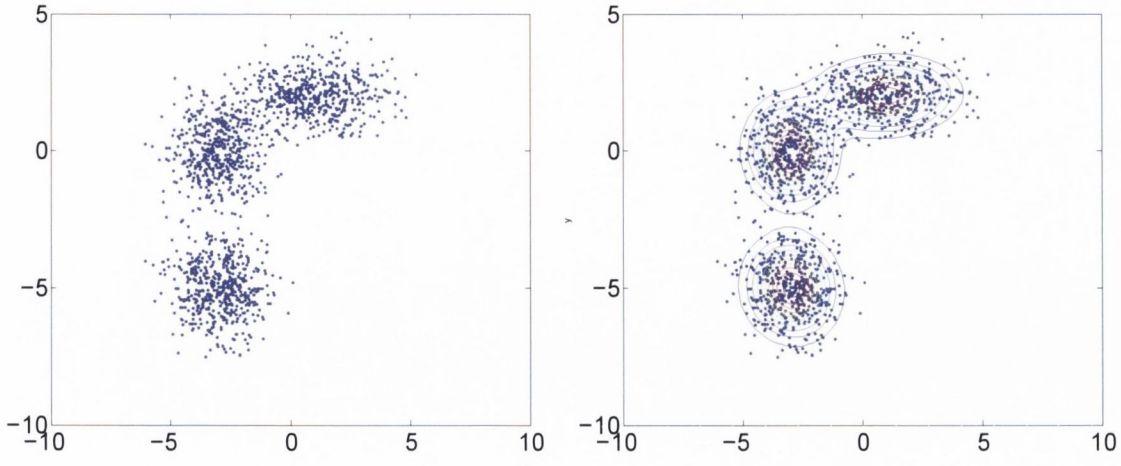
- $p(\Theta)$ is the prior distribution of the parameter Θ. It quantifies the uncertainty about Θ before taking the data into account. The priors can be categorized according to how informative they are. Informative priors take into account previous information related to the variable Θ. Weakly informative priors on the other hand, are commonly used as regularization terms. It only helps in the stabilization of the optimization algorithms and prevent solutions that would contradict past knowledge.

- $L(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)$ is the likelihood function of Θ and it is expressed as a function of the observations. When all the observations are considered independent to each other, the likelihood can be computed as follows:

$$L(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n) = p(\mathbf{x}_1|\Theta)p(\mathbf{x}_2|\Theta)....p(\mathbf{x}_n|\Theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\Theta) \tag{2.2}$$

- $p(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)$ is the posterior distribution that expresses uncertainty about the parameter Θ after taking into account the prior $p(\Theta)$ and the likelihood function $L(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)$. The parameters estimation problem can then be solved by maximising this posterior probability $p(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)$.

$$\hat{\Theta} = \arg\max_{\Theta} L(\Theta|\mathbf{x}_1, \mathbf{x}_2, ..\mathbf{x}_n)\, p(\Theta) \tag{2.3}$$

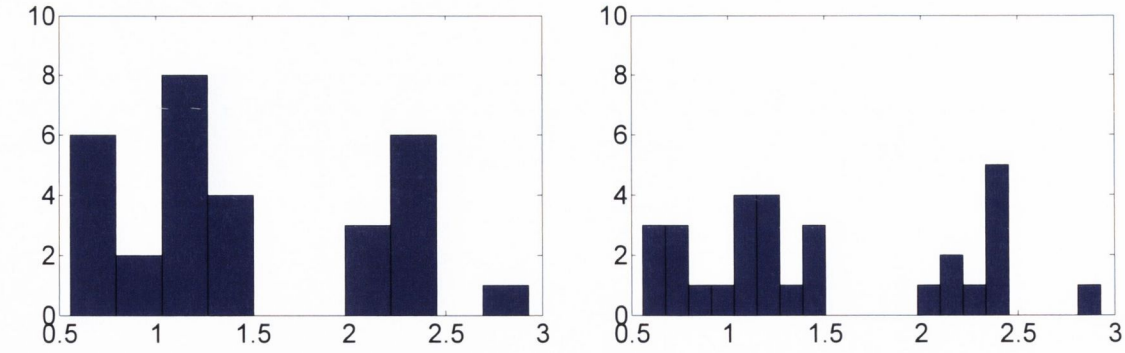


a) Observation $\mathcal{X} = \{\mathbf{x}_k\}_{k=1,\cdots,n}$ b) Inference (contour) $p(x|\widehat{\Theta})$

Figure 2.6: Statistical inference using parametric modelling. In a) a set of observation is shown as blue dots. In b) the estimated Gaussian Mixture (contour) is plotted as a contour along with the observations. The density function is assumed to be a Gaussian Mixture with three components. The optimisation is performed using a EM algorithm.

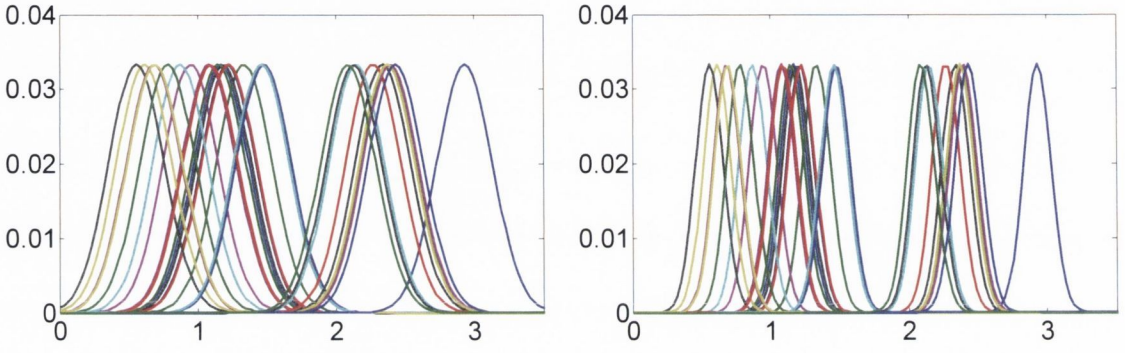
When there is no prior information available, the estimation of parameters can still be performed using only the likelihood function. Those methods are commonly known as EM-*like* algorithms (cf. Section 2.2.2). The success in the estimation relies on the assumption that the number of observations is big enough. This ensures the convergence towards the true density function. Unfortunately, this assumption is not always true for real applications leading towards an erroneous solution. Furthermore, there are two additional difficulties when inferring parameters from parametric models. The first one is related to the assumption needed over the underlying probability density function. The election of the type of density function to use biases the solution of the problem. The second issue is the lack of resistance to outliers. As it is shown in Equation 2.2 the likelihood is usually computed as the product of the probability of each observation given the parameters Θ. This product is very sensitive to outliers and it affects the robustness of the inference.

Non-parametric modelling: Non-parametric models for statistical inference are more flexible than parametric modelling. They make fewer assumptions about the form of the underlying probability density function. The parameters contained in these models control the complexity of the distribution rather than its form. Non-parametric techniques for density estimation are usually histograms and kernel functions. Histograms are the simplest method for density estimation. It is a partition of the range of possible values of the variable in intervals "bins". The probability density function is estimated according to the number of samples falling in each interval (cf. Figure 2.7 top row).
In a Kernel Density Estimation (KDE) the density function is given by the contribution of



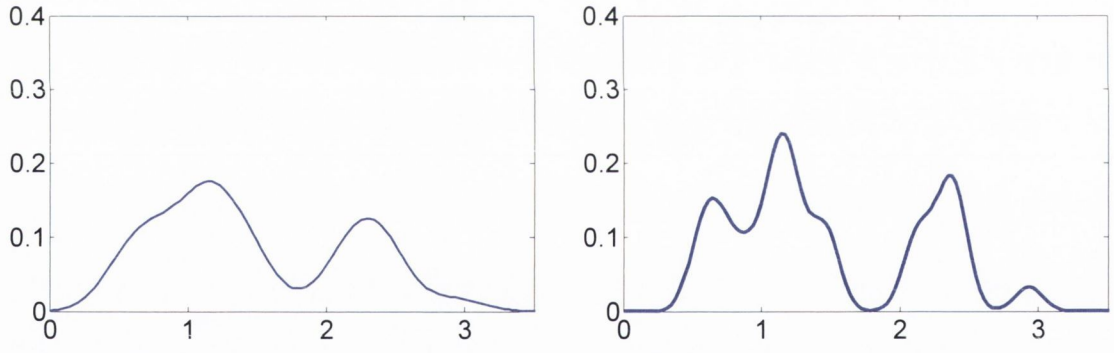
a) Histogram with 10 bins

b) Histogram with 20 bins



c) Gaussian Kernel, $h = 0.2$

d) Gaussian Kernel, $h = 0.1$



e) Kernel Density Estimation, $h = 0.2$

f) Kernel Density Estimation, $h = 0.1$

Figure 2.7: This figure shows the two approaches in non-parametric modelling: Histograms and kernel density estimation. We model the density function given a set of $n = 30$ observations. In the top row we present the histogram of the data set when using a) 10 and b)20 bins. In the middle row the same data set is represented using a Gaussian kernel for each point in the data set and using a bandwidth of $h = 0.2$ and $h = 0.1$ (Figures c and d respectively). Finally the density estimation using the kernels in c) and d) are presented in e) and f) respectively.

a set of kernel functions centred over each sample. The density function is then computed as the sum of the contribution of all the kernels as follows:

$$\widehat{p}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h^D}\mathcal{K}\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right)$$

Where h is the bandwidth of the kernel function and D the dimension where the density function is defined. The kernel function $\mathcal{K}$ can be any function $f(\mathbf{x})$ that meets the following three criteria:

$$\text{(a) } \int f(\mathbf{x})d\mathbf{x} = 1 \qquad \text{(b) } \int \mathbf{x}f(\mathbf{x})d\mathbf{x} = 0 \qquad \text{(c) } \int \mathbf{x}^2 f(\mathbf{x})d\mathbf{x} < \infty$$

The election of the right bandwidth for modelling the kernels is very important in non-parametric modelling since it directly affects the resulting density function. An example of the impact of the bandwidth selection is shown in Figure 2.7(bottom). Several methods have been proposed in literature for estimating the bandwidth [25, 26, 27]. The classical methods are based on the minimization of a similarity criterion between the estimated density function $\widehat{p}(\mathbf{x})$ and its true but unknown density function $p(\mathbf{x})$. However, a robust estimation is still difficult to achieve. Some well known distance criterion proposed in literature are:

$\int \lvert\widehat{p}(\mathbf{x}) - p(\mathbf{x})\rvert d\mathbf{x}$	Integrated absolute error
$\int \widehat{p}(\mathbf{x}) \log \widehat{p}(\mathbf{x})/p(\mathbf{x})d\mathbf{x}$	Kullback-Lieber distance
$\int [\widehat{p}(\mathbf{x})^{\frac{1}{2}} - p(\mathbf{x})^{\frac{1}{2}}]^2 d\mathbf{x}$	Hellinger Distance
$\int [\widehat{p}(\mathbf{x}) - p(\mathbf{x})]^2 d\mathbf{x}$	ISE: Integrated square error

2.2.2 Likelihood based estimation

In this section the Expectation Maximisation algorithm (EM) and the probabilistic self-organising map (SOM) are reviewed. The EM algorithm is the base of several methods for shape registration and it is the inspiration for the probabilistic self-organising map that will be used in a section of this thesis.

Expectation maximisation algorithm

The EM algorithm is an efficient and iterative procedure to compute the Maximun Likelihood estimate in presence of missing or hidden data. It estimates the parameters of the model for which the observation are the most likely. This algorithm consists of two processes, the E-step (expectation) and the M-step (Maximisation).
Given a random vector $\mathbf{x}$ as the observations, the goal is to define an iterative algorithm for finding the parameters Θ of the density function that best represent the observation. We want to compute an update estimation such that after the n^{th} iteration we have:

$$L(\Theta_{n-1}) > L(\Theta_n) \tag{2.4}$$

With $L(\Theta) = \ln P(\mathbf{x}|\Theta)$ the log likelihood function. Since $\ln(\mathbf{x})$ is a strictly increasing function, the value of Θ that maximise $P(\mathbf{x}|\Theta)$ also maximise $L(\Theta)$. Maximising $L(\Theta)$

is equivalent to maximise the difference:

$$L(\Theta) - L(\Theta_n) = \ln p(\mathbf{x}|\Theta) - \ln p(\mathbf{x}|\Theta_n) \tag{2.5}$$

In order to consider unobserved data, a hidden vector Z is introduced. Now we write,

$$p(\mathbf{x}|\Theta) = \sum_z p(\mathbf{x}|\mathbf{z}, \Theta)p(\mathbf{z}|\Theta) \tag{2.6}$$

Using Jensen inequality $\ln \sum_i \lambda_i \mathbf{x}_i >= \sum_i \lambda_i \ln \mathbf{x}_i$ and letting the constant λ_i be of the form $p(\mathbf{z}|\mathbf{x}, \Theta_n)$ we can rewrite equation 2.5 as follows:

$$L(\Theta) - L(\Theta_n) = \sum_z p(\mathbf{z}|\mathbf{x}, \Theta_n) \ln \frac{p(\mathbf{x}, \mathbf{z}, \Theta)p(\mathbf{z}|\Theta)}{p(\mathbf{z}|\mathbf{x}, \Theta_n)p(\mathbf{x}|\Theta_n)} \tag{2.7}$$

$$L(\Theta) = L(\Theta_n) + \sum_z p(\mathbf{z}|\mathbf{x}, \Theta_n) \ln \frac{p(\mathbf{x}, \mathbf{z}, \Theta)p(\mathbf{z}|\Theta)}{p(\mathbf{z}|\mathbf{x}, \Theta_n)p(\mathbf{x}|\Theta_n)} \tag{2.8}$$

Defining $l(\Theta|\Theta_n) = L(\Theta_n) + \Delta(\Theta|\Theta_n)$, we can rewrite as follows:

$$L(\theta) > l(\Theta|\Theta_n) \tag{2.9}$$

Maximising $l(\Theta|\Theta_n)$ is then a way of optimising the log likelihood function since $l(\Theta|\Theta_n)$ is bounded above by the likelihood function.

Some of the advantages of the EM algorithm are its conceptual simplicity, easy of implement, and the fact that each iteration improves Θ. However, it can require many iterations to converge towards the solution, and higher dimensionality can dramatically slow down the process. Furthermore, the EM algorithm works better when the fraction of missing information is small and the dimensionality of the data is not too large.

Self organising maps (SOM)

Self Organising Maps were first introduced by Kohonnen et al [28] and it is a data analysis method that combines vector quantisation with topology preservation. Given an input vector $\{\mathbf{x}_k\}_{n=1,\cdots,N}$ and a set of nodes $\{\mathbf{u}_s\}_{s=1,\cdots,S}$ in a latent space with some specific topology associated (i.e a grid defined by a neighborhood). Each node is defined with a position on the latent space g_s and a centre in the input vector space $\mathbf{u}_s$. We associate a winning node (r) for each point in the input vector $\mathbf{x}$ and update the position of the node $\mathbf{u}_r$ in the vector space according to the following expression:

$$\mathbf{u}_r = \frac{\sum_{n=1}^{N} h_{s_n r}\mathbf{x}_n}{\sum_{n=1}^{N} h_{s_n r}} \tag{2.10}$$

Where $h_{s_n r}$ is a neighbourhood function evaluated at r and the winning neuron s_n for that point in the input vector $(\mathbf{x}_n)$.

$$h_{rs} = exp(-\lambda||g_r - g_s||^2) \tag{2.11}$$

Figure 2.8 shows a scheme of the input vector (black dots), the nodes (green dots) and the topology associated (red lines).

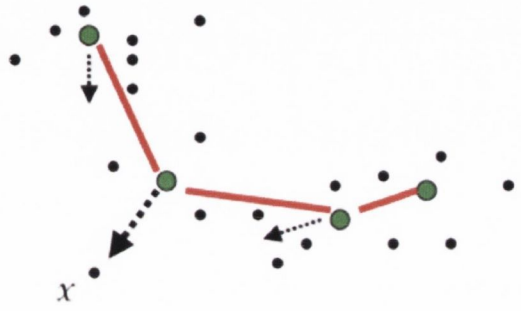


Figure 2.8: Scheme of SOM [4]

One of the limitations of the standard SOM algorithm is the method used for the assignation of the winning neuron. It is mainly based on the Euclidean distance. This metric is not always appropriate. Especially when the input data is not expressed as real-valued vector. This introduces the question of which measure of dissimilarity would be most appropriate and whether the centre of the neuron should be the same type of object than the input data. A second limitation is that this algorithm cannot be expressed as a cost function so it has no guarantee of convergence. In order to address those limitations Verbeek et al. [4] introduced the Self-Organizing mixture model which defines the standard SOM algorithm as a EM optimization problem. This is done by adding constrain to the EM algorithm according to a SOM strategy in order to enforce topology. The expression for $p(\mathbf{z}|\mathbf{x}, \Theta_n)$ in equation 2.8 can be constrained by using the normalised neighbourhood function as follows:

$$p(\mathbf{z}|\mathbf{x}, \Theta_n) \propto h_r(s) = exp(-\lambda||g_r - g_s||^2) \tag{2.12}$$

With,

$$\sum_s h_r(s) = 1$$

An example of the performance of the algorithm is presented in Figure 2.9 where a set of nodes arranged as a curricular network are organized according to the input data (black dots).

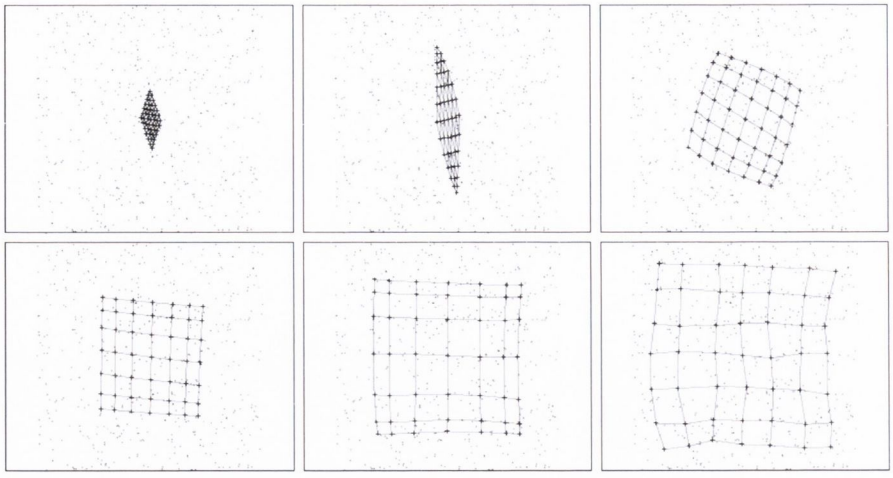


Figure 2.9: Example of Self Organizing map for diferents iterations. The dots are the set of observations. The grid is a set of connected nodes that are automatically organised to represent the observations [4]

2.3 Shape inference applications

In this section we focus on specific applications for shape parameters estimation such as point set registration, morphable model fitting and ellipse fitting.

2.3.1 Registration & correspondences

Point set registration is the task of assigning correspondences between two sets of points and to recover the transformation that maps both sets of points (cf. Figure 2.10). Registration is the base for several applications [29, 30, 31, 32, 33, 34, 35]. There is a vast literature dedicated to algorithms that solve the registration problem. However, we will focus here on the most relevant algorithms. They can be categorised as ICP-based algorithms [36, 37] and probabilistic modelling for parameters estimation [38, 39, 40, 41, 42, 5].

Iterative Closest Point (ICP)

One popular method to perform registration between two point sets is the Iterative Closest Point (ICP) algorithm [36, 37]. The ICP algorithm was introduced by Besl et al. [36] and Zhang [37]. It is based on a point-to-point correspondence between the two data sets performed using the nearest neighbour criteria. Once the correspondences have been found the transformation is calculated. These two steps (correspondence-transformation) are iterated until the convergence criterion is reached. It is one of the most used algorithms for registration due to its simplicity and its low computational complexity. Many

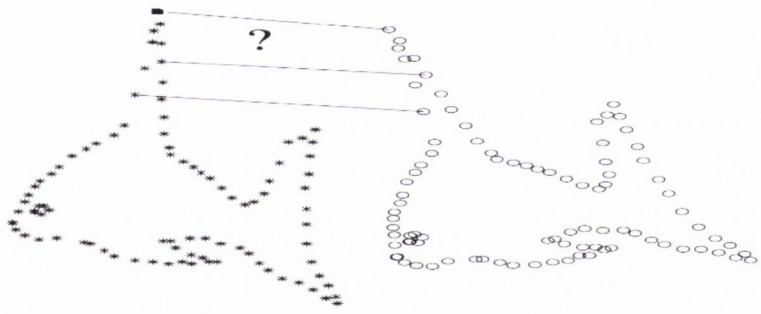


Figure 2.10: Point to point correspondence between data sets. [5]

improvements have been made to the basic ICP algorithm [43, 44, 45, 46, 47]. However, they are still sensitive to outliers and initialisation. ICP requires the initial position of the two point sets to be adequately close. This is usually achieved by matching manually labelled points in both sets [48, 49]. An alternative to improve the initial guess of the ICP algorithm is by using RANSAC [50]. RANSAC (Random Sample Consensus) is an iterative algorithm inroduced by Fischler et al. [51]. It is used to estimate parameters of a mathematical model from a set of observed data which contains outliers. The algorithm consists in selecting iteratively a random subset of the original data to estimate the parameters. The iterations are performed until the parameters estimated well explain the observation [52]. The main disadvantage of this algorithm is there is no upper bound on the number of iterations needed to compute the parameters. Moreover, RANSAC is not always able to find the optimal solution even for moderately contaminated data sets and it usually performs badly when the number of inliers in the data is less than 50%.

Probabilistic methods

In order to overcome the limitation of ICP based algorithms several probabilistic methods have been reported. We categorise those algorithms in two groups: Methods based on EM-like algorithms and those based on matching probability density functions.

EM-like algorithms: The registration problem can be redefined as a density estimation problem. Gold et al. [53] proposed a Robust Point Matching algorithm (RPM) where one of the data sets is used as a sample of the density function modelled with the reference data set [54, 40, 41]. This modelling comes from the assumption that samples are uniformly distributed around points in the reference dataset. The problem can be solved as a two-step optimisation (correspondence-transformation) but in an EM-like fashion where the centroids of the density function are estimated with the transformation parameters. Adding an extra kernel to model the outliers compensates the sensitivity to outliers of

the EM algorithm. Several algorithms follow the strategy of modelling the registration explicitly as a Maximum Likelihood problem. Some of them are Luo et al. [55], Myronenko et al. [5] and Ganger et al. [42]. McNeil et al. [56] converted a Procrustes analysis into a maximum likelihood framework using the EM algorithm. The idea is to improve the robustness of Procrustes to noise and clutter. The algorithm is used for shape matching where the background is modelled using a Gaussian kernel in order to reduce its sensitivity to outliers. Most of those methods use for simplicity isotropic covariance matrices for modelling the GMMs. An improvement for those models was recently introduced by Horaud et al. [57]. They use non-isotropic covariance matrices for modelling the GMM and solve the estimation problem using an Expectation conditional Maximisation algorithm. Luo et al.[58] proposed a unified framework to tackle the alignment and correspondence problem simultaneouslly. The idea is to constrain the recovery of pose parameters using relational constraints provided by the structural arrangement of the points. Two density functions are defined modelling the errors of the correspondence and the alignment respectively. Using a EM algorithm both density functions are linked using interleaved iterative steps. The cross-entropy is used as a utility measure between the two probability distributions.

EM-based algorithms have shown to be more robust to outliers than those based on ICP. However, the main drawback of those algorithms is that they rely on the assumption that a dense set of observations is available. This implies a many-to-one correspondence, which is not true when the number of observations is roughly the same (or less) than the number of points in the reference set.

Matching probability density functions: In this case, two probability density functions are modelled; one for each data set. The estimation is then solved by minimising a similarity metric between those density functions [59, 60]. Methods using this strategy differ to each other depending on the modelling of the density functions and the metric used for the estimation. Glaunes et al.[61], for instance, propose the case where two discrete distributions are considered (weighted sums of Dirac measures). It is shown to behave properly in limit of continuous distributions on sub-manifolds. As a consequence, the algorithm may be applied to various matching problems, such as curve or surface matching (via a sub-sampling), or mixings of landmark and curve data. Jian et al.[39] propose the use of GMMs to model both data sets and to estimate parameters by minimising the $\mathcal{L}_2$ distance between them. A similar approach is used by Wang et al.[62] for aligning multiples shapes. The $\mathcal{L}_2$ distance is a divergence metric between probability density functions. Other divergence metrics that have been used include the Cauchy-Schwartz divergence [63], Shanon Divergence [64], Bregman divergence [65] and Havrda-Charvt divergence [66].

Tsin et al. [38], on the other hand, used the cross-correlation between the two density functions as a measure of similarity. However, it can only solve the rigid registration between shapes. Jian et al.[39] show that using the cross-correlation for solving the rigid transformation is equivalent to use the $\mathcal{L}_2$ distance. Furthermore, the $\mathcal{L}_2$ distance has the advantage of having a closed form solution when density functions are modelled as Gaussian Mixtures. In practice, those methods use isotropic covariance when modelling the GMMs. Furthermore, they usually solve the problem by approximating one of the density functions by its empirical distribution. In this case the problem can be interpreted as maximising the expected likelihood kernel since it is the expectation of one distribution with respect to another [67] (We will discuss the implications of this approximation in more detail in Chapter 3). An isotropic modelling for the GMM is also proposed by Roy et al. [68]. They define an EM algorithm that is implemented for solving the non-rigid transformation between shapes and uses the $\mathcal{L}_2$ distance as part of the optimisation. However, the likelihood is modelled assuming one data set as samples of the density function modelled using the second data set (as in most EM-like algorithms).

2.3.2 3D face reconstruction

Face reconstruction is a very attractive research area as it forms the basis of a wide range of applications such as face animation [69, 48], human-computer interfaces, face recognition [6, 70] and medical applications. Methods for reconstructing 3D faces can be categorised depending on the nature of the input data and their applications. The most accurate system to capture a 3D face is the laser scanner, which records a dense 3D map. Unfortunately its high cost and the need for cooperation from the individual during the acquisition process has limited its usage [71]. Some cheaper alternatives rely on the inference of the 3D face geometry from RGB images, where multiple cues from images or multiples views of the scene are used for inferring the 3D shape. A faster 3D capture system can be achieved by using structured light. However, this technique is less accurate than the 3D scanner and it requires the use of multiples images or post-processing techniques to improve its performance [72, 73, 74]. We review as follows reconstruction methods from RGB and RGB-D images.

3D face reconstruction RGB images

Inferring 3D face information from 2D images is a cheap, fast and non-invasive alternative. Approaches that attempt to solve the 3D face reconstruction problem from 2D images can be classified in two groups: *Analysis by synthesis* and *Shape from X*. Analysis by synthesis techniques are based on parametric statistical modelling. They aim at reconstructing 3D faces by finding the parameters of a generative face model that synthe-

size the closest face to the input data [75].

Shape from X on the other hand, is a group of techniques that use some specific cue (X) to infer the 3D shape. Examples of cues commonly used in 3D object reconstruction that have been applied to face reconstruction are: Motion, Contour, Shading, Stereo and Silhouette among others [76, 77, 78, 79]. Using an array of video cameras, Bradley et al. [80] succeeds, however, in capturing 3D textured faces by combining multi-view stereo with tracking techniques.

Analysis by synthesis on the other hand has proven to achieve better results due to the use of prior information. It uses a morphable model of the face built over a set of observations taken from a training database [75, 81, 82, 83]. The problem can then be defined in a Bayesian framework where the parameters of the model that best fit the observations are estimated by maximising its posterior probability given the input data.

Most optimisation algorithms used for fitting morphable models to RGB images are based either on Stochastic Newton Optimization (SNO) [75], Linear Shape and Texture Fitting (LiST) [70] or Inverse Compositional Image Analysis (ICIA) [84, 85]. The evaluation of these methods depends on their application. For instance, SNO is reported to be more accurate but it lacks efficiency compared with ICIA [6]. Improvements to these algorithms have been achieved by including additional information to the cost function. This can be done by including multiple features from a single image [86] or using multiple images [87, 88, 89, 90]. This multiple feature/image strategy provides a fitting algorithm more robust to local minima, perhaps due to the smoothness of the overall cost function achieved by the extra information used. However, correspondences in between the input images and the model are assumed known in order to match the extra features. In practice, this correspondence is difficult to achieve and mismatches during the starting of the fitting process affect the robustness and accuracy of the fitting algorithm.

3D face reconstruction from depth map

A popular alternative for capturing a 3D map is by using an RGB-D sensor. Two technologies of RBG-D sensors co-exist: time of flight cameras and structured light cameras. Those sensors are usually cheaper than the 3D laser scanner. Both provide a noisy recording of the 3D environment, however they have become a hot topic of research in recent years as an alternative for 3D shape reconstruction. n particular, the availability of commercial devices such as the Microsoft Kinect has triggered the development of many algorithms for improving its performance. Its usage has been extended for many new applications other than just video games, for which the Kinect was originally designed.

Scanning 3D faces with an RGB-D sensor does not provide accurate results. The recorded data is very noisy and the depth map contains holes or missing data due the limitations of the sensor. Two approaches have been recently explored in the literature

for dealing with those problems. The first one relies on the use of multiple acquisitions and combines them in order to obtain a high-resolution 3D shape. The second uses a 3D face model that is fitted to the captured data. Results achieved when fitting shape models performed better since they are capable of inferring details of the face such as the mouth or eyes. These details are usually lost when inferring a 3D shape by averaging all the registered depth scans.

The performance of most fitting algorithms depends on the initial correspondences between the observed data and the shape model. For instance, the representation of a particular feature observed (i.e tip of the nose) must be in correspondence with the vertex representing the same feature in the shape model. As a consequence most fitting algorithms depend on feature selection or landmarking. This is required for initialisation, limiting their automation and with that, their applications. 3D reconstruction using structured light or Time-of-Flight cameras has been a hot research topic in the past few years [91, 92, 93, 94, 95, 96]. If the quality of the recorded depth map is noisy and suffers from missing data, it is still an attractive alternative for 3D reconstruction. Using a Time-of-Flight camera, Cui et al. [97] propose to merge a set of depth maps to infer a less noisy 3D shape. The key problem to solve is the alignment or registration of all the captured depth scans. Similarly, using a turning table and a cheap Kinect camera, Ruttle et al. infer a 3D shape of an object by merging several point clouds recorded from different viewpoints around that object [98]. The registration is performed by maximising a cross-correlation between two pdfs. The accuracy achieved is shown to be similar to the 3D surface recorded with an expensive Laser scanner [99].

Newcombe et al. [100] propose the KinectFusion system for merging sequential depth scans recorded with a Kinect camera and inferring a 3D mesh of a scene in real time. The noise is reduced by filtering each scan before registration. Focusing on faces, Hernandez et al. [101] extend the KinectFusion approach to infer laser scan quality 3D faces. However, small details are lost due to spatiotemporal smoothing used in the process. Weise et al. [48] use kinect depth images to infer the facial expressions and dynamics of an actor to animate an avatar in real time. Their approach uses a user-specific expression model that is fitted to the observed scans with the Iterative Closest Point (ICP) algorithm. Similarly using a Kinect camera, Zollhofer et al. [69] propose an automatic method for 3D face reconstruction using Morphable Models. Their algorithm relies on feature landmarks that can be detected from the face (eyes, nose and chin).

Schneider et al. [102] proposed two algorithms based on the ICP framework for registering laser scans of human heads. Holes may occur in the resulting mesh but these can be filled in using a prior model for heads. Using a morphable model also helps for efficiently resampling the inferred mesh.

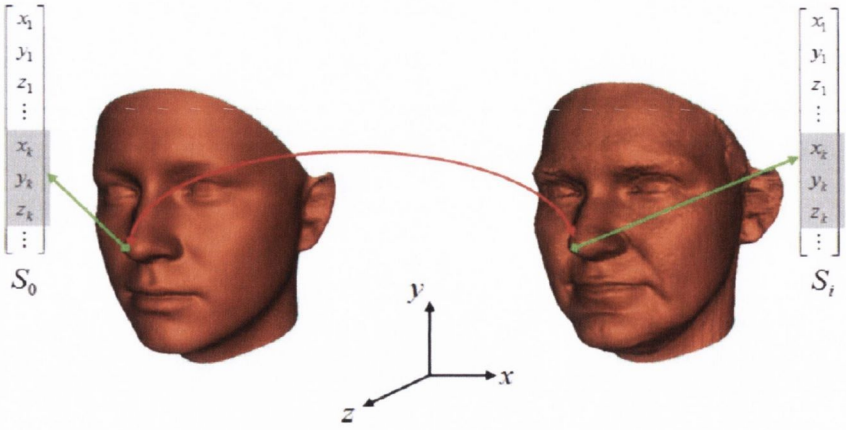


Figure 2.11: The shape correspondence concept of the 3D Morphable Model: The figure illustrates how vertices are indexed in the parametrized domain. For instance, according to Levine et al. [6], the tip of the nose will be assigned in the same position (k) in the vector shape s_i in every scan, even though the coordinate x, y and z that represent its position will change among individuals.

2.3.3 Morphable shape model

In 1995 Cootes et al.[103] proposed to build a shape model that can deform in order to fit a data set or image. They argued that a model should only be able to deform in ways characteristic of the class of objects it represents. The model is created by learning patterns of variability from a training set of correctly annotated images. These models can be used for an image search in an iterative refinement algorithm analogous to that employed by Active Contour Models (Snakes). Since then, several improvements have been made for building those models [104, 105, 106, 107] and many applications have been reported [108, 109, 110]. We review as follows the 3D model of a human face since it will be used in the experimental part of this thesis.

The 3D Morphable Model is a generative model for 3D face reconstruction introduced in 1999 by Vetter and Blanz [75]. This model contains a low dimensional parametrization of face shape and texture. It is obtained by statistically capturing shape and texture variability in a set of training faces. Recent contributions addressing the building process of a Morphable Model have been made by Patel et al. [81], Paysan et al [82] and Basso el at [83]. All of them describe the process in 4 steps: data acquisition, correspondence, shape alignment and finally the statistical modelling.

3D face data acquisition of the training set usually involves 3D laser scan or a structured light system. The laser scan is more accurate but its long acquisition time is very critical for capturing natural-looking faces. Once all individuals are scanned, the data set is parametrized in a common domain to establish correspondence and alignment along the faces. This ensures that all points such as eyes or mouth corners for instance share the same position in the parameterized domain (cf. Figure 2.11). Correspondence can be

achieved, for example, by using automatic techniques such as the Iterative Closest Point algorithm (ICP)[36] or manually by labelling landmarks among the data set. The correspondence and alignment steps effectively help in creating the shape (s_i) and texture (t_i) vector for each individual i:

$$s_i = [x_{1,i}, y_{1,i}, z_{1,i}, x_{2,i}, y_{2,i}, z_{2,i}, ...x_{n,i}, y_{n,i}, z_{n,i}]$$
$$t_i = [r_{1,i}, g_{1,i}, b_{1,i}, r_{2,i}, g_{2,i}, b_{2,i}, ...r_{n,i}, g_{n,i}, b_{n,i}]$$

Using Principal Component Analysis, we can define any new shape as a function of the mean shape $\overline{S}$ and a linear combination of q orthonormal bases:

$$s_{new} = \overline{S} + \sum_{i=1}^{q} \alpha_i S_i + \varepsilon$$

Similarly, using the average texture $\overline{T}$ and taking the m orthonormal bases we can define any new texture vector as follows:

$$t_{new} = \overline{T} + \sum_{i=1}^{m} \beta_i T_i + \varepsilon$$

In both cases ε represent the error coming from the eigenvectors that are not used in the linear representation. Assuming a Gaussian distribution for the error, we can estimate the probability function of shape (α) and texture (β) as follows:

$$p(\alpha) \propto e^{-\frac{1}{2}\sum_{i=1}^{q} \frac{\alpha_i^2}{\sigma_{s,i}^2}}$$

$$p(\beta) \propto e^{-\frac{1}{2}\sum_{i=1}^{m} \frac{\beta_i^2}{\sigma_{t,i}^2}}$$

Those probability functions $p(\alpha)$ and $p(\beta)$ represent the prior information for shape and texture respectively. In this thesis, we use the Gaussian model for the shape developed by Paysan et al [82] which is publicly available for research.

2.3.4 Ellipse fitting

Fitting ellipse to data is a challenging problem that arises in several fields. Some examples of applications are segmentation of cells [111] study of galaxies [112], medical diagnostics [113], camera calibration and face detection among others [114, 115]. As many applications as there are of fitting ellipses there are also a great number of algorithms proposing solutions to this problem [116]. They are commonly classified in three categories: Least Square based methods, Hough Transform based methods and the most recent approach known as edge contour following methods.

Least Square based methods are usually classified in two categories according to the cost function to optimise. Those categories are: (1) methods minimising a *geometric* error, and (2) methods minimising an *algebraic* error [117, 118, 119, 120]. The minimisation of the geometric error corresponds to the Maximum Likelihood (ML) estimation for the ellipse parameters. ML methods are regarded as the most accurate methods for ellipse fitting and various computational schemes have been proposed [121]. However, those methods have several drawbacks. They need several iterations for solving the non-linear optimization problem and they are very sensitive to noise. Therefore, the convergence of those methods is not guaranteed. Moreover, the convergence usually depends on the accuracy of the initialisation.

Algebraic methods on the other hand are easier to implement and computationally efficient. However, the main problem of those methods is that they do no guarantee the result will be an ellipse. A normalization process is required in order to enforce the solution [9, 122]. For instance, Szpak et al. [123] propose a penalty function that guaranteed an ellipse when using the Sampson distance. However, the result is then biased by the normalization scheme chosen. Algebraic methods are less robust with respect to ML methods when the data is coming from a small segment of an ellipse. Furthermore, accuracy of the results of those methods still depends on the initialisation as in ML algorithms. A method for finding a reasonable starting guess for ML algorithm is proposed in [124].

The Hough Transform is a well known approach to detect ellipses [125, 126, 127]. It is based on a voting system where each edge pixel of an image is considered. This voting procedure is carried out in a parameter space, from which candidate ellipses are obtained as local maxima in a accumulator space that is explicitly constructed based on the parameters of the ellipse. Several algorithms have been proposed to improve performance of the HT method [128, 14, 129]. Some approaches explore the inclusion of additional information such as the directional property of the pixels [130, 131]. Unfortunately, those methods are easily affected by possible noise in the image. A different strategy that improves the computational complexity of the HT is to sub-sample the data set. For instance, Kiryati et al. [132] used the Probabilistic Hough Transform (PHT) where just a portion of the edge pixel of an image is used. Xu et al.[133] on the other hand, proposed the Randomised Hough Transform (RHT) which used randomly chosen n-tuples of data points. This method was originally designed for detecting circles but it was extended to ellipses by McLaughlin et al.[13]. The Randomized Hough Transform serves as a powerful variant of the standard Hough Transform that exploits the geometric properties of ellipses in order to speed up the detection process [134, 135]. Despite its simplicity and efficiency, the RHT performs poorly if the target ellipses overlap (or mutually-occlude) with each other.

A general advantage of Hough Transform based methods is that they do not require connectivity in between consecutive edge pixels. This makes these algorithms useful when the observation is a sparse set of points. The main drawback is that they are very sensitive to the choice of the quantisation of the parametric space. Incorrect quantisation leads to the detection of false ellipse or missing the true ellipse. Furthermore, the performance of the HT algorithms deteriorate when the number of ellipses in the image increase.

Edge following methods exploit the connectivity between edge pixels to detect ellipses [12, 11, 111, 10, 136]. The main strategy of these methods is to detect arcs and then group them in order to detect the ellipses. Kim et al. [12] and Mai et al.[11] connect pixels by line fragments from where the arcs are computed. The grouping of arcs becomes a critical process since errors in that stage would be propagated to the ellipse detection step. Chia et al.[10] introduce some improvements by introducing a self-correction stage where the grouping process is corrected using a feedback loop. In other words, low confidence ellipses are replaced by a set of better hypothetical ellipses obtained by combining arcs from different groups. Those methods are considered the most successful in detecting multiples of ellipses in digital images. However, they only work when there is connectivity along the edge pixels. If the observation is a sparse set of data points, this methodology can not be implemented.

2.4 Summary

In this chapter we have reviewed the theoretical base that will be used in this thesis. We first presented an overview of shape analysis and statistical inference. The application of parameter estimation techniques for solving computer vision problems was illustrated in three challenging problems: registration, shape fitting for object reconstruction (a human face in particular) and curve fitting (ellipse). The literature suggests pending problems to solve in order to achieve a method for estimating parameters that can be accurate, robust, easy to implement and hopefully computationally efficient. Fitting algorithms for instance, depend on the correspondence between the model and the observation. This correspondence, usually achieved through registration algorithms, is still difficult to obtain. Registration algorithms have been improved thanks to the use of probabilistic methods. However, those methods are limited by the modelling of the density function and therefore, the modelling of the cost function to implement. For instance, EM-like algorithms are based on a likelihood function that is very sensitive to outliers. Many robust techniques based on statistical inference have been applied, for instance, matching density function for parameter estimation. However, their implementation is generic and does not include any particular information related to the shape for which the parameters are

being estimated. In the remainder of this thesis we aim to model a method for shape parameter estimation based on robust statistical techniques to include more information related to the shape when modelling the cost function.

Part I

Method: Inference with Bayesian $\mathcal{L}_2$

Chapter 3

Inference with $\mathcal{L}_2$ and $\mathcal{L}_2$E

The $\mathcal{L}_2$ distance between probability density functions has been recently used as a metric for shape registration (cf. Section 2.3.1). However, in practice, $\mathcal{L}_2$ is often approximated by another metric called $\mathcal{L}_2E$. In this chapter, we analyse the implications of using the $\mathcal{L}_2E$ distance for parameter estimation and its performance in terms of accuracy and robustness. In Section 3.1 we present a brief review of the concept of divergence between probability density function and the definition of the $\mathcal{L}_2$ distance. In Section 3.2 we introduce $\mathcal{L}_2E$ and analyse the assumptions made when using this metric for parameter estimation. We discuss, in particular, the work proposed by Jian and Vermuri [39] where the $\mathcal{L}_2$ distance is used as a cost function for point cloud registration (cf. Section 3.3). A set of experiments are performed in order to illustrate the performance of $\mathcal{L}_2E$ and how it relates to robust estimation. The results obtained motivate us to explore the use of the full $\mathcal{L}_2$ distance between density functions as a general framework for shape parameters estimation.

3.1 Divergence between probability density functions

The divergence between probability density functions is a measure of similarity. A popular family of divergence is the power density divergence. Given two densities functions $f(\mathbf{x})$ and $g(\mathbf{x})$, the power density divergence can be expressed as follows [137, 138, 139]:

$$d_\alpha(g, f) = \int \left\{ \frac{1}{\alpha} g^{1+\alpha}(\mathbf{x}) - \frac{1+\alpha}{\alpha} g(\mathbf{x}) f^\alpha(\mathbf{x}) + f^{1+\alpha}(\mathbf{x}) \right\} d\mathbf{x} \tag{3.1}$$

For $\alpha > 0$. When $\alpha = 0$ this expression becomes the Kullback-Leibler divergence [140]:

$$d_0(g, f) = \lim_{\alpha \to 0} d_\alpha(g, f) = \int g(\mathbf{x}) \log \left\{ \frac{g(\mathbf{x})}{f(\mathbf{x})} \right\} d\mathbf{x} \tag{3.2}$$

When $\alpha = 1$, this divergence becomes the Euclidean distance between the two probability functions, $f(\mathbf{x})$ and $g(\mathbf{x})$:

$$d_1(g, f) = \int \left\{ g^2(\mathbf{x}) - 2\, g(\mathbf{x})\, f(\mathbf{x}) + f^2(\mathbf{x}) \right\}\, d\mathbf{x} \tag{3.3}$$

The Euclidean distance d_1 between density functions is commonly known in statistical inference as $\mathcal{L}_2$ distance. This metric has been used as a quality of fit criteria for statistical inference when using non-parametric models and also as distance criterion for bandwidth selection (integrated square error [141, 27]). It has been shown to be a robust approach compared to other power divergences ($\alpha \neq 1$ in Equation 3.1) [60, 137, 139]. Furthermore, Jian et al.[39] have demonstrated the superiority of using $\mathcal{L}_2$ with respect to the log-likelihood (LME) for estimating transformation parameters. In this thesis, we will use $\mathcal{L}_2$ for shape registration and fitting. In the next paragraphs additional arguments in favour of $\mathcal{L}_2$ are presented.

3.1.1 Closed form solution of $\mathcal{L}_2$

Let us consider two probability density functions f and g that are Gaussian mixtures:

$$g(\mathbf{x}) = \sum_{i=1}^{n_g} w_i^g\, \mathcal{N}(\mathbf{x}; \mu_i^g, \Sigma_i^g)$$

and

$$f(\mathbf{x}) = \sum_{i=1}^{n_f} w_i^f\, \mathcal{N}(\mathbf{x}; \mu_i^f, \Sigma_i^f)$$

where $\mathcal{N}(\mathbf{x}; \mu_i^*, \Sigma_i^*)$ is the normal probability density function of mean μ_i^* and covariance Σ_i^*, and $0 \leq w_i^* \leq 1$ is its weight in the mixture (with $* = f, g$) such that $\sum_{i=1}^{n_*} w_i^* = 1$. For $\mathbf{x} \in \mathbb{R}$, $\mathcal{L}_2$ can then be computed explicitly using the following result:

$$\int \mathcal{N}(\mathbf{x}; \mu_1, \sigma_1^2)\, \mathcal{N}(\mathbf{x}; \mu_2, \sigma_2^2)\, d\mathbf{x} = \mathcal{N}(\mu_1 - \mu_2; 0, \sigma_1^2 + \sigma_2^2) \tag{3.4}$$

For more details and the formula for higher dimensions of $\mathbf{x}$, see Appendix A.1. The $\mathcal{L}_2$ has a closed form solution for Gaussian mixtures. Gaussian Mixtures Models (GMM) are a family of density functions that can approximate any probability density functions well when one is not limited by the number of components (n_g and n_f). However, when using GMM for representing shapes a parsimonious description should also be considered. In the following Chapter (cf. Chapter 4) a method for a parsimonius representation of shapes using GMM is discussed. The bandwiths are modelled as non-isotropic in order to reduce the number of components needed in the Gaussian Mixture.

3.1.2 Inference with $\mathcal{L}_2$

Consider that g depends on a parameter Θ such that $g(\mathbf{x}|\Theta)$ is a family of probability density functions (model), and f is a target distribution. $\mathcal{L}_2$ can be used for inferring Θ such that $g(\mathbf{x}|\Theta)$ is close to $f(\mathbf{x})$ as follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{L}_2(\Theta) = \|f - g\|^2 \right\} \tag{3.5}$$

This approach can be used for finding parameters of a GMM (model g) fitting a distribution (target f) that can be estimated ($\hat{f}$) from a set of observations [141]. Choosing $\hat{f}$ as a GMM means that $\mathcal{L}_2(\Theta)$ is available in closed form.

This global modelling for estimating parameters has two main advantages. First, it avoids the need for correspondence between the target distribution and the observations. Secondly, a global modelling improves its robustness to missing data and noise. The disadvantage is that the model can be pulled away from an optimal fit by outliers and noise. However, we propose in Chapter 4 a method that ameliorate the effects of noise and outliers by manipulating the bandwiths of the GMM representing the shapes.

3.1.3 $\mathcal{L}_2E$ approximation

Let us consider a set of observations $\{\mathbf{x}_i\}_{i=1,\cdots,n_f}$. The empirical probability density function provides a simple estimate $\hat{f}$ for the target f:

$$\hat{f}(\mathbf{x}) = \frac{1}{n_f} \sum_{i=1}^{n_f} \delta(\mathbf{x} - \mathbf{x}_i) \tag{3.6}$$

where δ corresponds to the delta Dirac function (a.k.a. normal distribution with zero valued variance). The parameters Θ can then be estimated by minimising the $\mathcal{L}_2$ as follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{L}_2(\Theta) = \int \{ g^2(\mathbf{x}|\Theta) - 2\, g(\mathbf{x}|\Theta)\, \hat{f}(\mathbf{x}) + \hat{f}^2(\mathbf{x}) \}\, d\mathbf{x} \right\} \tag{3.7}$$

The last term in the integral does not depend on the latent variable Θ so it can be discarded from the cost function leading to the $\mathcal{L}_2E$ approximation [141]:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{L}_2E(\Theta) = \int g^2(\mathbf{x}|\Theta) d\mathbf{x} - 2 \int g(\mathbf{x}|\Theta)\, \hat{f}(\mathbf{x})\, d\mathbf{x} \right\} \tag{3.8}$$

The second term can be interpreted as the correlation between the two density functions. Since one density function is the empirical distribution, this cross correlation term can be

computed as the expectation of $g(\mathbf{x}|\Theta)$, given the observations. We denote this term as:

$$\mathcal{C}E(\Theta) = \int g(\mathbf{x}|\boldsymbol{\Theta})\ \hat{f}(\mathbf{x})\ d\mathbf{x} = \frac{1}{n_f}\sum_{i=1}^{n_f} g(\mathbf{x}_i|\Theta) \tag{3.9}$$

$\mathcal{C}E$ is only valid as long as the empirical density function $\hat{f}$ is a good approximation of f. Next, we explore experimentally the performance of $\mathcal{L}_2E$ for parameter estimation.

3.2 $\mathcal{L}_2E$ parameter estimation

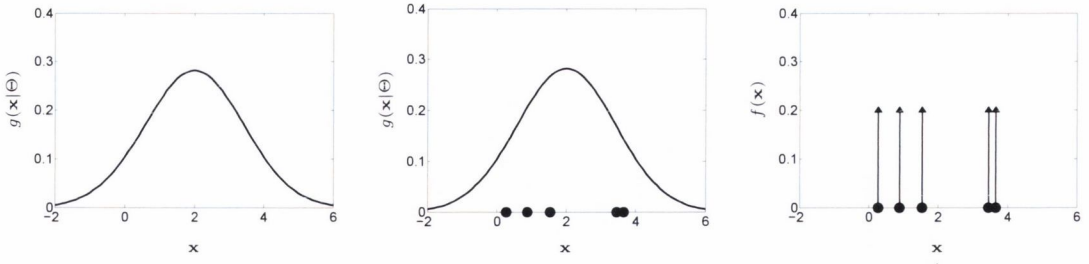
The empirical distribution is a good approximation of the true density function only if it meets the following criteria [141]:

- The number of observations n_f is large.
- The samples (observations) have been randomly collected from $f(\mathbf{x})$.

Both requirements may not be met in practice. Furthermore, the samples cannot be considered as randomly collected from $f(\mathbf{x})$ when occlusion or outliers are present.

3.2.1 Sample size

In this experiment, we evaluate how well the empirical distribution approximates the true density function for different sample sizes. Lets assume we have a model $g(\mathbf{x}|\Theta = (\mu, \sigma))$ defined as a normal distribution (cf. Figure 3.1 a). A set of observations are generated from $g(\mathbf{x}|\Theta = (\mu = 2, \sigma = 2))$ and used to estimate $\hat{f}$ (cf. Figures 3.1 b and c).

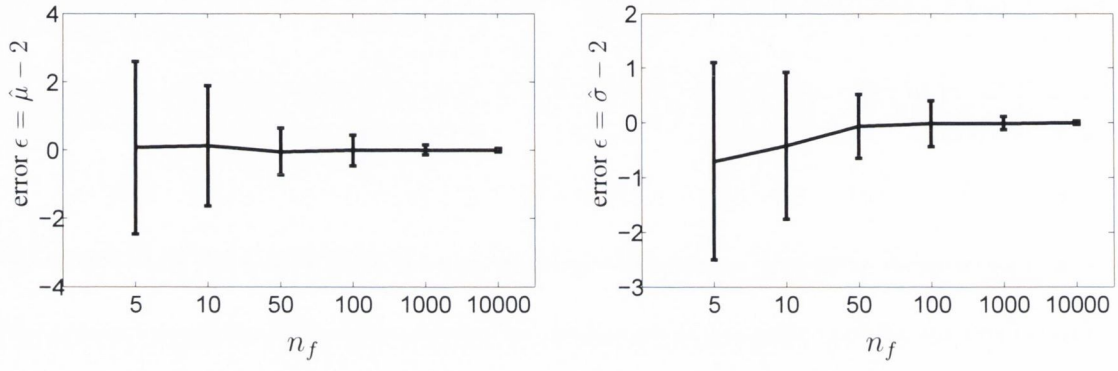


a) $g(\mathbf{x}|\Theta)$ b) Observations (black dots) c) $\hat{f}(\mathbf{x})$

Figure 3.1: Example: a) Normal distribution $g(\mathbf{x}|\Theta = (\mu = 2, \sigma = 2))$. Figure b) shows a set of observations sampled from g (black dots) and Figure c) shows the empirical distribution $\hat{f}$ associated with the observations. The empirical distribution is represented by a delta Dirac function centred in each observation.

Using $\mathcal{L}_2E$ we estimate the parameters μ and σ. We run 100 experiments for each sample size of $n_f = 5, 10, 50, 100, 1000$ and 10000, respectively. The error between the

ground truth ($\mu = 2, \sigma = 2$) and their estimated values are reported in Figure 3.2 with respect to the sample size n_f. The 95% confidence interval is also reported (computed with the 100 runs). The estimated values ($\hat{\mu}$ and $\hat{\sigma}$) better converge to the ground truth as the sample size used for the estimation increases. Furthermore, the confidence interval gets narrower when increasing the sample size, indicating that the estimates are all closer to each other in all runs. We have verified that as more observations become available estimates become more accurate and reliable.



a) Average error $\hat{\mu} - 2$ w.r.t. n_f b) Average error $\hat{\sigma} - 2$ w.r.t. n_f

Figure 3.2: Results obtained when running 100 experiments for samples sizes of $5, 10, 50, 100, 1000$ and 10000 reported on the abscissa. The figure shows the average error between the ground truth and the estimates with a 95% confidence interval for the mean in a) and standard deviation in b). In both cases the error decreased when increasing the number of samples n_f.

3.2.2 Robustness to outliers

In a second experiment outliers have been added to evaluate robustness of $\mathcal{L}_2E$ for parameters estimation (cf. Figure 3.3). We used the same data sets generated in the previous experiment but added extra samples (outliers) generated from a different normal distribution ($\mathcal{N}(\mathbf{x}; \mu = 7, \sigma = 3)$).

We ran 100 experiments for each sample size ($5, 10, 50, 100, 1000$ and 10000) using 5 levels of outliers percentages (5%, 10%, 20%, 30% and 50%). The mean and standard deviation obtained for the estimated $\hat{\mu}$ and $\hat{\sigma}$ are reported in Table 3.1. In Figure 3.4, we plot the average error obtained in the estimation of μ and σ when sample size are $n = 5$ and $n = 10000$. The confidence in the estimation is higher when increasing the sample size. However, the resistance to outliers deteriorates steadily as the proportion of outliers increases when both the mean and standard deviation are estimated with $\mathcal{L}_2E$. The average error is over 5% in both cases (μ and σ) when the percentage of outliers is larger than 10%.

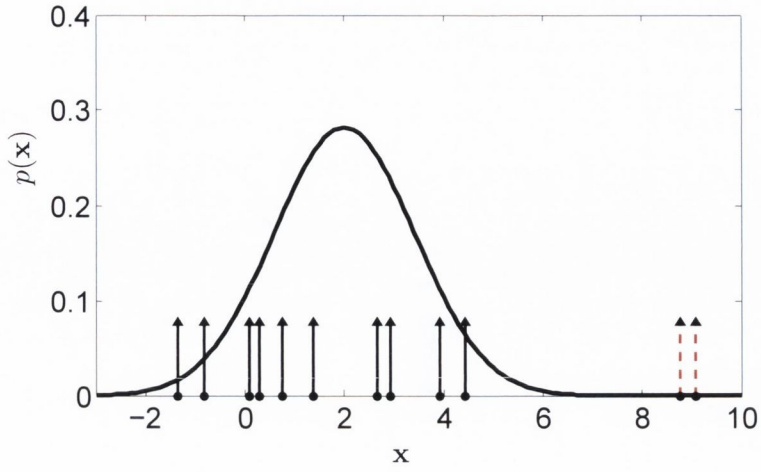


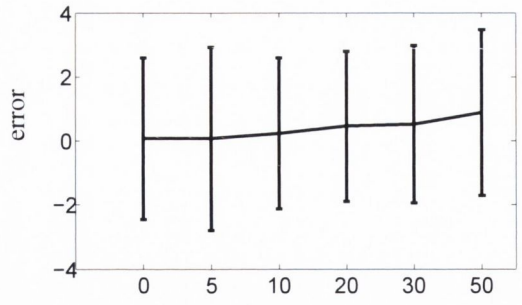
Figure 3.3: The figure shows a set of samples taken from the true density function $g(\mathbf{x}|\Theta = (\mu = 2, \sigma = 2))$, and a few outliers drawn from a different density function (in red dash line).

Figure 3.5 shows a plot of the estimated solutions for different sample sizes without outliers. When the sample size increases, the standard error associated with the estimates decreased. However, the estimates get further away to the ground truth as more outliers are present (cf. Figure 3.6).

3.2.3 Estimation with fixed bandwidth

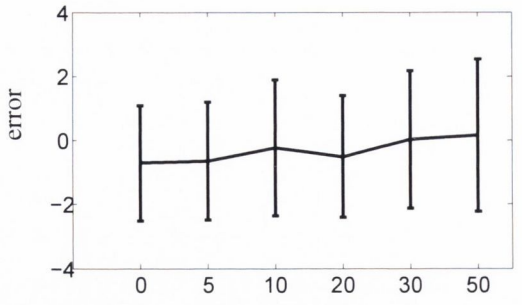
As we have shown, $\mathcal{L}_2E$ is sensitive to outliers for estimating μ and σ simultaneously. However, an arbitrary selection for the bandwidth can also affect the estimated solution for μ. This is illustrated with the following example: let us consider that we collected $n = 5$ samples from the density function $g(\mathbf{x}|\Theta)$ and added two outliers. The cost function to optimise given by $\mathcal{L}_2E$ for different choices of bandwidth is displayed in Figure 3.7. For small bandwidths ($\sigma = 1$ (red line) and $\sigma = 2$ (blue dashed line)) the cost function shows multiple local minima. The estimated solution will then depend on the starting position of the optimisation algorithm. On the contrary, if the bandwidth is large ($\sigma = 5$) the cost function becomes smooth. In this case, it has a single global solution but could be shifted because of outliers. Therefore, this solution may not truly represent the ground truth.

In Table 3.2 we reported the value corresponding to the minimum of the cost function for each fixed bandwidth (σ) along with its respective estimated value for $\hat{\mu}$. Note also that due to the approximation of $\mathcal{L}_2$ by $\mathcal{L}_2E$, the value of the cost function $\mathcal{L}_2E$ is no longer positive (as one would expect when using the Euclidean distance $\mathcal{L}_2$).



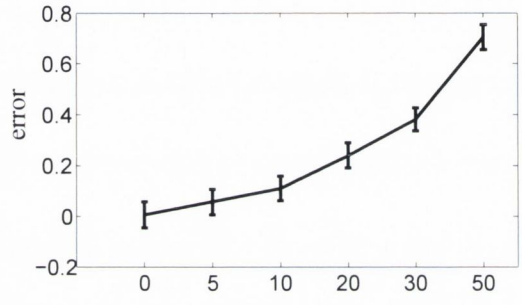
% of outliers

c) Average error $\hat{\mu} - 2$ w.r.t. % outliers ($n_f = 5$)



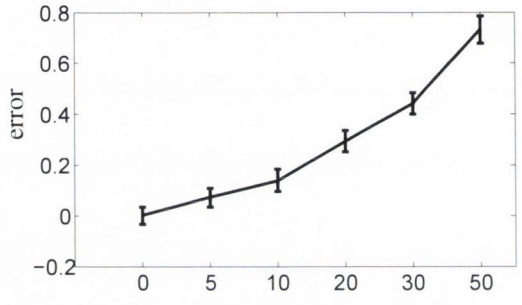
% of outliers

d) Average error for $\hat{\sigma}$ w.r.t. % outliers ($n_f = 5$)



% of outliers

c) Average error $\hat{\mu} - 2$ w.r.t. % outliers ($n_f = 10000$)



% of outliers

d) Average error for $\hat{\sigma}$ w.r.t. % outliers ($n_f = 10000$)

Figure 3.4: Average error (between ground truth and estimates) with 95% confidence intervals with sample size $n_f = 5$ (top row) and $n_f = 10000$ (bottom row).

3.3 Point cloud registration with $\mathcal{L}_2E$

Jian and Vermuri [39] have recently proposed the use of the $\mathcal{L}_2$ distance between probability density functions for point cloud registration. The rigid parameters Θ to be estimated are rotation and translation between shapes. The scaling between shapes is not discussed. The main goal of Jian's algorithm is to find correspondences between data sets. They represent the two discrete point sets by continuous density functions (Gaussian Mixtures). One point set is used to model $g(\mathbf{x}|\Theta)$ and the other models the target distribution $f(\mathbf{x})$. These density functions are explicitly modelled using the following settings:

1. The number of Gaussian components is the number of points in the point sets and all components are weighted equally.

2. For each component, the mean vector is given by the spatial location of each point.

Ground truth: $\mu = 2$ $\sigma = 2$ Outliers: $\mu = 7$ $\sigma = 3$

Outlier	$n = 5$	$n = 10$	$n = 50$	$n = 100$	$n = 1000$	$n = 10000$
0%	$\hat{\mu}$: 2.06 ± 1.26	2.13 ± 0.87	1.95 ± 0.34	1.99 ± 0.22	2.01 ± 0.07	2.00 ± 0.02
	$\hat{\sigma}$: 1.29 ± 0.90	1.58 ± 0.67	1.93 ± 0.28	1.98 ± 0.21	1.99 ± 0.06	2.00 ± 0.01
5%	$\hat{\mu}$: 2.06 ± 1.53	2.11 ± 0.86	2.08 ± 0.29	2.00 ± 0.23	2.04 ± 0.07	2.05 ± 0.02
	$\hat{\sigma}$: 1.35 ± 0.92	1.73 ± 0.68	2.07 ± 0.28	2.04 ± 0.18	2.06 ± 0.06	2.07 ± 0.01
10%	$\hat{\mu}$: 2.23 ± 1.18	2.26 ± 0.88	2.14 ± 0.35	2.13 ± 0.22	2.11 ± 0.07	2.10 ± 0.02
	$\hat{\sigma}$: 1.76 ± 1.05	1.86 ± 0.68	2.09 ± 0.26	2.11 ± 0.19	2.14 ± 0.05	2.13 ± 0.02
20%	$\hat{\mu}$: 2.45 ± 1.16	2.33 ± 0.98	2.22 ± 0.35	2.24 ± 0.22	2.23 ± 0.07	2.23 ± 0.02
	$\hat{\sigma}$: 1.48 ± 0.95	1.97 ± 0.65	2.23 ± 0.31	2.29 ± 0.20	2.28 ± 0.06	2.29 ± 0.02
30%	$\hat{\mu}$: 2.52 ± 1.22	2.47 ± 0.91	2.41 ± 0.34	2.37 ± 0.24	2.38 ± 0.08	2.38 ± 0.02
	$\hat{\sigma}$: 2.01 ± 1.07	2.27 ± 0.75	2.15 ± 0.35	2.37 ± 0.24	2.48 ± 0.07	2.44 ± 0.02
50%	$\hat{\mu}$: 2.88 ± 1.29	2.71 ± 0.85	2.65 ± 0.33	2.68 ± 0.24	2.70 ± 0.07	2.70 ± 0.02
	$\hat{\sigma}$: 2.15 ± 1.19	2.40 ± 0.85	2.68 ± 0.32	2.69 ± 0.25	2.72 ± 0.06	2.73 ± 0.02

Table 3.1: Average estimates of the mean and standard deviation with their confidence intervals (100 runs) with varying proportions of outliers and sample size.

Fixed Bandwidth (σ)	Cost Function ($\mathcal{L}_2 E$)	Estimated ($\hat{\mu}$)
1.0	0.0072	3
1.5	0.0312	1
3.0	-0.0984	2 (correct solution)
5.0	-0.3693	3

Table 3.2: The global minimum depends on the selected fixed bandwidth σ.

3. All components share the same spherical covariance matrix (identical fixed bandwidth).

Settings in (1) and (2) can be understood as modelling the density functions using Kernel Density Estimates (cf. section 2.2.1). The complexity of computing the cost function is proportional to the number of kernels used ($\mathcal{O}(\max\{n_f^2, n_f n_g, n_g^2\})$). This representation of the density function affects the computational *efficiency* of the algorithm when working with large data sets.

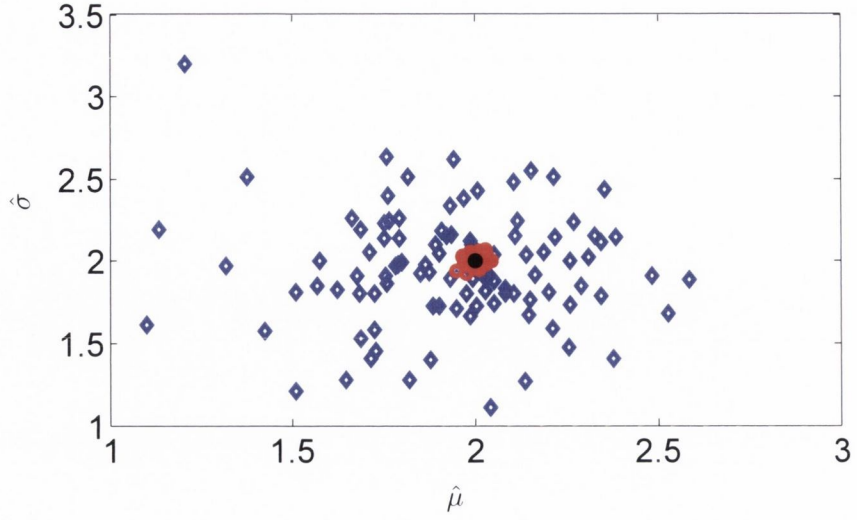


Figure 3.5: Estimates (100 experiments) with mean $\hat{\mu}$ reported in absissa and $\hat{\sigma}$ on the y-axis (0% outliers). The ground truth $(\mu, \sigma) = (2, 2)$ is plotted in black, estimates with $n_f = 100$ (blue) and estimates with $n_f = 10000$ (red).

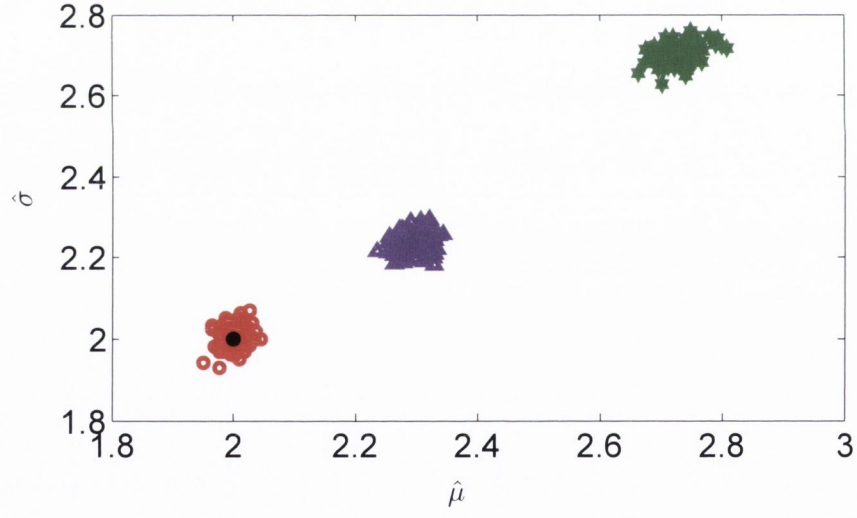


Figure 3.6: Estimates (100 experiments) for $n_f = 10000$ with mean $\hat{\mu}$ reported in abscissa and $\hat{\sigma}$ on the y-axis. The ground truth $(\mu, \sigma) = (2, 2)$ is plotted in black. The estimates without outliers are shown in red, the estimated with 20% of outliers are shown in purple and the green one correspond to the estimated using 50% of outliers.

The selection of the bandwidths are critical as they control how well the kernel Density Estimate (KDE) represents the true density functions in particular when observations are sparse. Moreover, the bandwidths give the user the ability to tailor the density functions such that they accurately represent the shapes of interest, as described by the data sets. Setting (3), mentioned above, considers the same spherical covariance matrix for all kernels. In this case the cost function (for the rigid transformation) is in fact equiva-

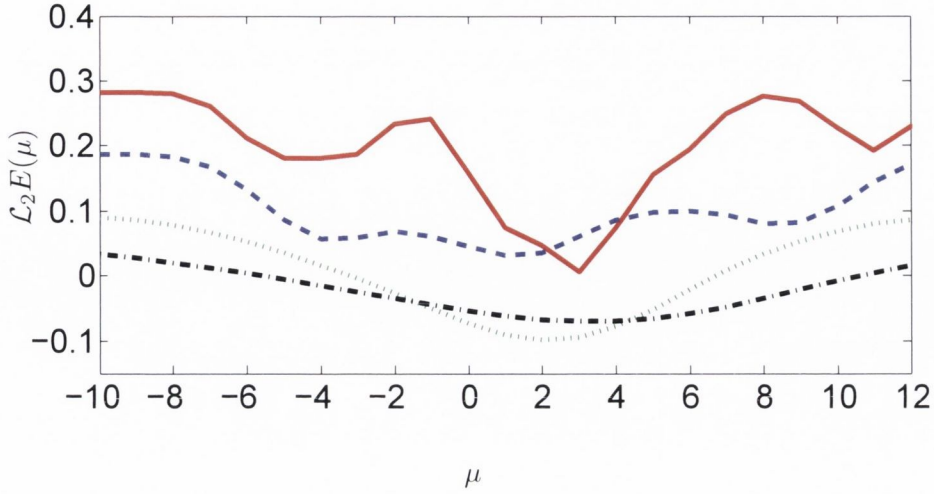


Figure 3.7: Cost function ($\mathcal{L}_2E(\mu)$) for a sample size $n_f = 5$ with two outliers. The red line corresponds to $\sigma = 1$, blue dashed ($\sigma = 1.5$), green dots ($\sigma = 3$) and black dash-dot ($\sigma = 5$). The figure shows the impact of the bandwidth on the estimation of μ using the $\mathcal{L}_2E$ approximation.

lent to $\mathcal{L}_2E$ (i.e. using the empirical probability density estimate to approximate f) (cf. Appendix A.2).

In general, the use of $\mathcal{L}_2E$ for parameters estimation presents the following issues:

- **Trade-off between accuracy and efficiency:** $\mathcal{L}_2E$ is a good approximation to $\mathcal{L}_2$ only if the number of samples is large enough. However, a large sample size implies a reduction of computational efficiency when estimating the parameters. This trade-off between *accuracy* and *efficiency* limits the applicability of the algorithm.

- **Bandwidth selection and robustness**: we have previously shown that robustness of $\mathcal{L}_2E$ deteriorates when estimating the bandwidth. The robustness of $\mathcal{L}_2E$ is only achieved when the bandwidth is fixed. However, the selection of the bandwidth influences the results of the estimation. Furthermore, the key idea of modelling data sets as density functions is to approximate a shape that is inherently a continuous function. From this point of view, the bandwidth associated with each Gaussian should play a better role in modelling the shape to fill in the gap between the observations.

The assumptions made in the modelling of the density functions favour the pair of point sets of similar sample rates. The $\mathcal{L}_2E$ registration approach can perform poorly in cases where the two point sets to be registered are captured from different sensors at different resolutions, missing data (e.g. holes) or outliers (e.g. occlusion) (cf. Section 5.1).

3.4 Conclusion

In this chapter, the concept of power divergence between probability density functions and its use as a cost function for shape parameter estimation was introduced. We have presented the $\mathcal{L}_2$ metric and its approximation $\mathcal{L}_2E$. The $\mathcal{L}_2$ distance has been used for the registration of shapes in the form of point clouds [39]. The density functions are modelled as KDEs. This representation affects efficiency of the estimation algorithm. Furthermore, the selection of the bandwidth affects the representation of the shape in the density function. In the case of rigid registration, the cost function is equivalent to $\mathcal{L}_2E$. The approximation made in $\mathcal{L}_2E$ relies heavily on hypotheses that may not be true in practice affecting accuracy, robustness and efficiency of this method.

As an alternative, to overcome these limitations with $\mathcal{L}_2E$, we propose to use $\mathcal{L}_2$ for the more general problem of shape fitting. Using $\mathcal{L}_2$ implies modelling Gaussian Mixture models well suited for shape representation. Moreover, we extend $\mathcal{L}_2$ to include prior information about the latent variable of interest making the whole framework Bayesian (cf. Chapter 4). The idea is to be able to deal with a vast number of problems such as shape registration, fitting and detection. This thesis explores the role of covariances (or bandwidths) in the modelling of the density functions used in $\mathcal{L}_2$ as it is thought to play an important role not only in the robustness of the optimisation algorithm but also in the efficiency of the algorithm used for optimisation. We will show in the following chapters that GMMs have the potential for encoding all the information needed for representing shapes efficiently and accurately leading to an effective Bayesian $\mathcal{L}_2$-based framework for their inference.

Chapter 4

Bayesian $\mathcal{L}_2$ for Shape Inference

In this chapter we propose a Bayesian framework for estimating shape parameters. It is based on the $\mathcal{L}_2$ distance between density functions. Hence, the observations and the class of shape of interest are modelled as density functions. In Section 4.1 we present the general framework and the shape estimation problems to solve in this thesis. We propose to model the density function of the shapes as Gaussian Mixtures (GMM) and discuss the role of the covariance matrices for a proper and efficient representation of shapes (cf. Section 4.2). Finally, in Section 4.3, we augment the dimension of the GMM in order to include additional information about the shape of interest. We show an example where the curvature of the shape is considered.

4.1 Bayesian $\mathcal{L}_2$

Let us consider two probability density functions $f(\mathbf{x})$ (the target function) and $g(\mathbf{x}|\Theta)$ (the model) defined for a random variable $\mathbf{x} \in \mathbb{R}^D$. The probability density function g is indexed by a parameter Θ that has a prior distribution $p_\Theta(\Theta)$. The metric $\mathcal{L}_2$ or Euclidean distance $\mathbf{d}$ (Equation 4.1) computed between the two density functions f and g are close to zero when g is similar to f, and on the contrary far from 0 if g is very dissimilar to f.

$$\mathbf{d}(\Theta) = \left(\int_{\mathbb{R}^D} \left\{ g^2(\mathbf{x}|\Theta) - 2\, g(\mathbf{x}|\Theta)\, f(\mathbf{x}) + f^2(\mathbf{x}) \right\} d\mathbf{x} \right)^{\frac{1}{2}} \tag{4.1}$$

Θ can be estimated by minimising $\mathbf{d}$ (or $\mathcal{L}_2$) and this can be translated into the following equation:

$$\mathbf{d}(\Theta) = \epsilon \sim p_\epsilon(\epsilon) \tag{4.2}$$

where the random quantity $\mathbf{d}(\Theta)$ is interpreted as the error between two density functions f and g. The error has a probability density function p_ϵ that is chosen such that:

$$\hat{\Theta} = \arg\max_{\Theta} p_\epsilon(\mathbf{d}(\Theta)) = \arg\min_{\Theta} \mathbf{d}(\Theta) \tag{4.3}$$

The inference of Θ using Equation 4.3 does not take into account any prior p_Θ about the latent quantity Θ that may be available. In order to have an inferential framework that is fully Bayesian, we introduce an auxiliary random variable λ and augment Equation 4.2 as follows:

$$\lambda + \mathbf{d}(\Theta) = \epsilon \sim p_\epsilon(\epsilon) \tag{4.4}$$

Equation 4.4 is equivalent to Equation 4.2 when $\lambda = 0$. In other words, problem 4.2 is a particular case of the problem 4.4. This reformulation allows us to identify the conditional probability density function of λ given Θ as $p_{\lambda|\Theta}(\lambda|\Theta) = p_\epsilon(\lambda + \mathbf{d}(\Theta))$ verifying:

$$\int_{\mathbb{R}} p_{\lambda|\Theta}(\lambda|\Theta)\, d\lambda = \int_{\mathbb{R}} p_\epsilon(\lambda + \mathbf{d}(\Theta))\, d\lambda = 1 \tag{4.5}$$

and this result is independent of the choice of the probability density function p_ϵ. The probability of the random variable Θ given λ can then be expressed using Bayes law as follows:

$$p_{\Theta|\lambda}(\Theta|\lambda) = \frac{p_\epsilon(\lambda + \mathbf{d}(\Theta))\, p_\Theta(\Theta)}{\int p_\epsilon(\lambda + \mathbf{d}(\Theta))\, p_\Theta(\Theta)\, d\Theta} \tag{4.6}$$

Since we are interested in estimating Θ when $\lambda = 0$, we propose to estimate Θ by maximising $p_{\Theta|\lambda}(\Theta|\lambda = 0)$:

$$\hat{\Theta} = \arg\max_{\Theta} p_{\Theta|\lambda}(\Theta|\lambda = 0) = \arg\max_{\Theta} \left\{ \underbrace{p_\epsilon(\mathbf{d}(\Theta))}_{\text{"Likelihood"}} \underbrace{p_\Theta(\Theta)}_{\text{Prior}} \right\} \tag{4.7}$$

This expression allows us to include prior information when estimating parameters using the Euclidean distance $\mathbf{d}$ between probability functions. The density function $p_\epsilon(\mathbf{d}(\Theta))$ can be interpreted as a sort of likelihood where the target function f has been taken into account and matched against the model g using the divergence $\mathbf{d}$.

In this thesis, we are going to show how this Bayesian framework can be successfully used for inference of shapes under the following assumptions:

- Both $f(\mathbf{x})$ and $g(\mathbf{x}|\Theta)$ are modelled using Gaussian Mixtures. As shown in Chapter 3, an explicit expression of $\mathcal{L}_2(\Theta)$ (or $\mathbf{d}(\Theta)$) is available when using Gaussian mixtures. We present in Section 4.2 how shapes can be well represented using Gaussian mixture models.

- For simplicity, we have used the following Gaussian prior for Θ in all our applica-

tions:

$$p_\Theta(\Theta) \propto \exp\left(-\sum_{j=1}^{J} \frac{(\theta_j - \mu_j^\theta)^2}{2\sigma_j^2}\right) \tag{4.8}$$

with $\Theta = [\theta_1, \theta_2 .... \theta_J]$. Using very large variances allows us to diminish the influence of the prior when needed and leads back to inference without a prior (cf. Equation 4.3).

- For simplicity and to be compliant with condition 4.3, we chose $p_\epsilon(\epsilon)$ also as a Gaussian distribution:

$$p_\epsilon(\epsilon) \propto \exp\left(-\frac{\epsilon^2}{2\tau^2}\right) \tag{4.9}$$

The bandwidth τ represents the tolerance given by the user for the model g to deviate from the target f and it can be used as a regulation term. It also allows for the compensation of the influence of the prior in the cost function. Choosing both p_ϵ and p_Θ as Gaussian distributions has the advantage of simplifying the cost function and using the log transformation, the estimation 4.7 is performed by:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{C}(\Theta) = \frac{\mathcal{L}_2(\Theta)}{\tau^2} + \sum_{j=1}^{J} \frac{(\theta_j - \mu_j^\theta)^2}{\sigma_j^2} \right\} \tag{4.10}$$

with $\mathcal{L}_2(\Theta) = (\mathbf{d}(\Theta))^2$.

To compute $\mathcal{L}_2(\Theta)$, we propose to use a GMM g defined as:

$$g(\mathbf{x}|\Theta) = \sum_{i=1}^{n_g} w_i^g \, \mathcal{N}(x; \mu_i^g(\Theta), \Sigma_i^g) \tag{4.11}$$

and the target GMM f as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{n_f} w_i^f \, \mathcal{N}(x; \mu_i^f, \Sigma_i^f) \tag{4.12}$$

As a consequence, the term $\mathcal{L}_2$ in the objective function in Equation 4.10 depends on the parameters of these mixtures:

$$\mathcal{L}_2(\Theta) = \mathcal{L}_2\left(\{(\mu_i^f, \Sigma_i^f, w_i^f)\}_{i=1,\cdots,n_f}, \{(\mu_i^g(\Theta), \Sigma_i^g, w_i^g)\}_{i=1,\cdots,n_g}\right) \tag{4.13}$$

Note how only the means of the model depend on Θ and not its covariances and weights. This simplifies the algorithms used to find the optimum, even more so when $\mu_i^g(\Theta)$ is linear. We will discuss in more detail the choice of the parameters in the GMMs f and g in Section 4.2. For clarity, we next present various functions $\mu_i^g(\Theta)$ used in this thesis.

4.1.1 Application to affine registration

The affine transformation between shapes is defined by three basic transformations: rotation, translation and scaling. In the case of 2D shapes for instance, the latent variable to estimate can be defined by the following parameters:

$$\Theta = [s_x, s_y, \phi, t_x, t_y]$$

Where s_x and s_y are the scaling parameters in both direction (x and y), ϕ is the angle of rotation between shapes and $\mathbf{t} = [t_x, t_y]$ is the translation vector. The relation $\mu_i(\Theta)$ is then written as:

$$\mu_i^g(\Theta) = \begin{pmatrix} s_x \cos\phi & -\sin\phi \\ \sin\phi & s_y \cos\phi \end{pmatrix} \mu_i^0 + \mathbf{t} \tag{4.14}$$

with a given μ_i^0. $\mu_i^g(\Theta)$ is not linear in Θ.

4.1.2 Application to morphable shape model fitting

Shape models are obtained by capturing the shape statistical variability from a set of training examples (cf. Section 2.3.3). We use morphable models computed with Principal Component Analysis (PCA). It provides parameters $(\overline{v}_i, T_{1i}, \cdots, T_{qi})$ such that:

$$\mu_i^g(\Theta) = \overline{v}_i + \sum_{j=1}^{q} \alpha_j \, T_{ji} \tag{4.15}$$

The latent variable to estimate Θ corresponds to

$$\Theta = [\alpha_1, \alpha_2, \alpha_3, ...\alpha_q]$$

and $\mu_i^g(\Theta)$ is linear w.r.t. Θ.

4.1.3 Combining affine registration and morphable shape fitting

Both models (Sections 4.1.2 and 4.1.1) can be combined to simultaneously register and fit a target shape using:

$$\mu_i^g(\Theta) = \begin{pmatrix} s_x \cos\phi & -\sin\phi \\ \sin\phi & s_y \cos\phi \end{pmatrix} \left(\overline{v}_i + \sum_{j=1}^{q} \alpha_j \, T_{ji} \right) + \mathbf{t} \tag{4.16}$$

with

$$\Theta = [s_x, s_y, \phi, t_x, t_y, \alpha_1, \alpha_2, \alpha_3, ...\alpha_q]$$

$\iota_q]$

and for given morphable model parameters $(\overline{v}_i, T_{1i}, \cdots, T_{qi})$.

and for given morphable model parameters $(\overline{v}_i, T_{1i}, \cdots, T_{qi})$.

4.1.4 Application to ellipse fitting

A point on an ellipse (i.e. at a given $t_i \in [0; 2\pi]$) can be created with the following equation:

$$\mu_i^g(\Theta) = \begin{pmatrix} a\cos t_i \\ b\sin t_i \end{pmatrix} \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} + \begin{pmatrix} x_o \\ y_o \end{pmatrix} \tag{4.17}$$

The latent variables to estimate are the parameters that define the curve

$$\Theta = [\gamma, a, b, x_o, y_o].$$

where (x_o, y_o) is the centre of the ellipse, a and b the semi-minor and semi-major lengths respectively and γ the angle of rotation of the ellipse with respect to the horizontal axis. $\mu_i^g(\Theta)$ is nonlinear w.r.t. Θ.

4.2 Modelling curves and surfaces with GMMs

Curves and surfaces may be described by continuous parametric functions, $t \in \mathbb{R} \rightarrow \mathbf{x}(t) \in \mathbb{R}^2$ and $(t_1, t_2) \in \mathbb{R}^2 \rightarrow \mathbf{x}(t_1, t_2) \in \mathbb{R}^3$, and they can be easily discretised to be fitted with a GMM. More often, shapes are directly available in discrete forms such as: a set of points (i.e. vertices), a set of points with their normal vectors (i.e. vertices and normals), a discrete set of connected points (i.e. vertices and edges in 2D, or vertices and faces in 3D). Note that we use the same terminology (vertex, edge, normal, face) as defined in the PLY computer file format (Polygon File Format) for meshes.

We denote $\{u_i\}_{i=1,\cdots,n}$ the vertices available to describe a discretised curve or a surface. If only the vertices are known (i.e. point cloud), then a GMM with isotropic bandwidth is fitted with:

$$\mu_i = u_i \quad , \Sigma_i = h\mathrm{I} \quad \text{and } w_i = \frac{1}{n} \tag{4.18}$$

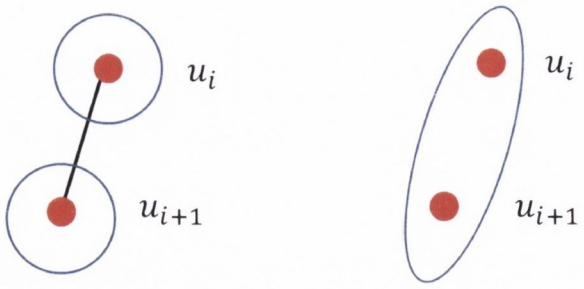
where I is the identity matrix and h a scalar or bandwidth that can be tuned by the user. This representation is valid in $\mathbb{R}^2$ and $\mathbb{R}^3$ (i.e. for curves and surfaces).

4.2.1 Using non-isotropic covariances

When more information is available, non-isotropic covariance matrices can be proposed (cf. Figure 4.1):

- For curves in $\mathbb{R}^2$, if edges are also given between vertices, for instance if u_i is connected to u_{i+1}, then the mean is chosen as their barycentre:

$$\mu_i = \frac{u_i + u_{i+1}}{2}$$



a) Isotropic kernel b) Non-Isotropic kernel

Figure 4.1: Modelling when using isotropic covariance matrices.

The covariance matrix which is by definition semi-positive, is written in the form $\Sigma_i = \mathrm{Q}\Lambda\mathrm{Q}^T$. The rotation matrix Q is defined as:

$$\mathrm{Q} = [\vec{n}_{1i}|\vec{n}_{2i}]$$

where $\vec{n}_{1i} = \frac{u_{i+1}-u_i}{\|u_{i+1}-u_i\|}$ is the tangent vector and $\vec{n}_{2i}$ is the normal vector defined such that $\|\vec{n}_{2i}\| = 1$ and $\vec{n}_{2i}^T\vec{n}_{1i} = 0$.

The matrix Λ is the diagonal matrix defined as:

$$\Lambda = \begin{pmatrix} \|u_{i+1} - u_i\|h_t & 0 \\ 0 & h \end{pmatrix}$$

The bandwidth h_t controls the fuzziness in the tangent direction, whereas h controls the fuzziness in the normal direction to the curve. The weights are chosen as a function of the covariance matrices in order to achieve a uniform value along the ridge of the surface defined by the GMM (cf. Figure 4.2):

$$w_i \propto \sqrt{|\Sigma_i|}$$

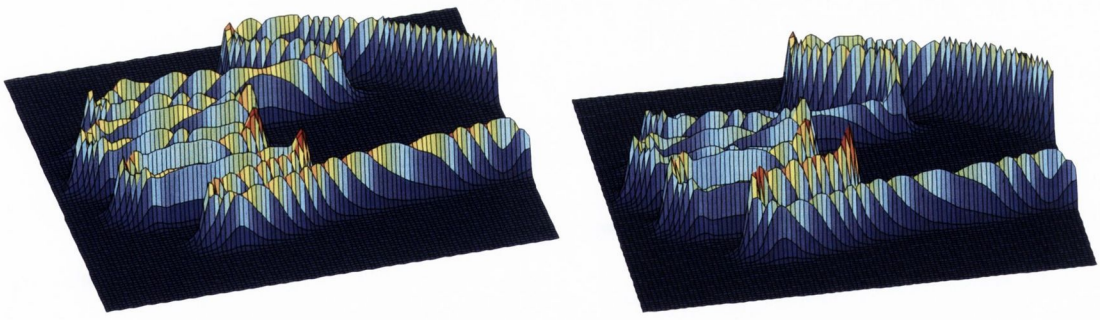
such that $\sum_i w_i = 1$.

- For surfaces in $\mathbb{R}^3$, if faces are given such that for instance (u_i, u_{i+1}, u_{i+2}) defines a triangular face, then the mean is defined as the barycentre:

$$\mu_i = \frac{u_i + u_{i+1} + u_{i+2}}{3}$$

The covariance matrix $\Sigma_i = \mathrm{Q}\Lambda\mathrm{Q}^T$ has a rotation matrix Q defined as:

$$\mathrm{Q} = [\vec{n}_{1i}|\vec{n}_{2i}|\vec{n}_{3i}]$$



a) Proportional weight b) Uniform weight

Figure 4.2: Effects of choosing a) proportional weight for the Gaussians instead of b) uniform weights. The ridge in a) has a similar value along the shape while in b) the ridge changes value along the shape.

where $\vec{n}_{1i}$ and $\vec{n}_{2i}$ are the tangent unit vectors while $\vec{n}_{3i}$ is the unit normal vector to the triangular face (cf. Figure 4.3). These are computed as the eigenvectors of the positive definite matrix (i.e. PCA is applied to the three vertices):

$$(u_i - \mu_i | u_{i+1} - \mu_i | u_{i+2} - \mu_i) \, (u_i - \mu_i | u_{i+1} - \mu_i | u_{i+2} - \mu_i)^T$$

The normal vector $\vec{n}_{3i}$ is the eigenvector associated with the zero eigenvalue. The eigenvalues h_{t1} and h_{t2} associated with $\vec{n}_{1i}$ and $\vec{n}_{2i}$ are used to define the diagonal matrix Λ:

$$\Lambda = \begin{pmatrix} h_{t1} & 0 & 0 \\ 0 & h_{t2} & 0 \\ 0 & 0 & h \end{pmatrix}$$

The bandwidths h_{t1} and h_{t2} control the fuzziness in the tangent direction. h controls the fuzziness in the normal direction to the surface and is set by the user. The weights are set to:

$$w_i \propto \sqrt{|\Sigma_i|}$$

subject to $\sum_i w_i = 1$.

As can be noticed, curves and surfaces are treated in a similar fashion and we refer to the following expression for this method:

$$(\mu_i, \Sigma_i, w_i) \tag{4.19}$$

where μ_i, Σ_i and w_i are computed as indicated in this section.

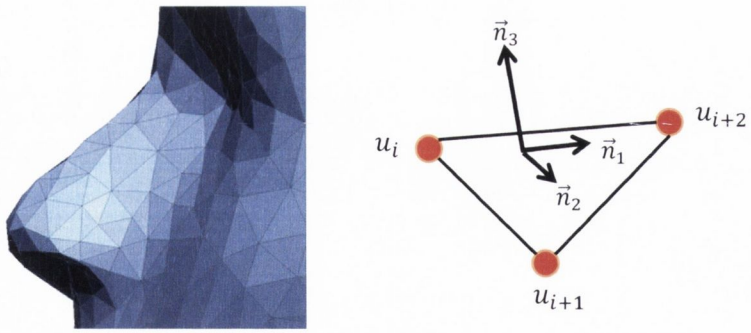


Figure 4.3: Scheme of the principal directions of the covariance matrix extracted from a given 3D mesh

4.2.2 GMM from shapes in images

Curves are not always directly available as a set of vertices in our applications. The curve of interest may come from the edges extracted from an image $I(\mathbf{x})$. The vertices $\{u_i\}$ correspond then to the edge pixel locations as found by an edge detector (e.g. Canny [142]). In addition, the gradient of the image $\nabla I = (I_x, I_y)^T$ is also available and can be used to augment the vertices with their corresponding normals $\vec{n}_{2i} = \frac{\nabla I(x_i,y_i)}{\|\nabla I(x_i,y_i)\|}$. We also define the tangent vector at location $u_i = (x_i, y_i)$:

$$\vec{n}_{1i} = \frac{1}{\|\nabla I(x_i, y_i)\|} \begin{pmatrix} I_y(x_i, y_i) \\ -I_x(x_i, y_i) \end{pmatrix} \tag{4.20}$$

The mean μ_i can be chosen as the vertices u_i and the covariance $\Sigma_i = \mathrm{Q}\Lambda\mathrm{Q}^T$ is computed with $\mathrm{Q} = [\vec{n}_{1i}|\vec{n}_{2i}]$ and the diagonal matrix Λ is:

$$\Lambda = \begin{pmatrix} h_t & 0 \\ 0 & h \end{pmatrix}$$

where h_t can be chosen proportional to the width of the pixel. h controls the fuzziness in the normal direction to the curve and is set by the user. The weights are computed as $w_i \propto \sqrt{|\Sigma_i|}$ subject to $\sum_i w_i = 1$.

4.2.3 Role of h in Bayesian $\mathcal{L}_2$ shape inference

The only free parameter to set when computing $\mathcal{L}_2$ is the bandwidth h. This parameter plays two important roles in the proposed estimation framework. First, it affects the description of the shape. It controls the fuzziness in the normal direction to the curve/surface (cf. Figure 4.6). Secondly, it affects the convexity of the cost function. Moreover, the cost function in Equation 4.10 can be expressed as a function of h as

follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{C}(\Theta) = \frac{\mathcal{L}_2(\Theta, h)}{\tau^2} + \sum_{j=1}^{J} \frac{(\theta_j - \mu_j^{\theta})^2}{\sigma_j^2} \right\} \tag{4.21}$$

The optimisation is performed using Gradient Ascent (GA) algorithms depending on the choice of the initial guess $\Theta^{(0)}$, the orthogonal bandwidth h and the bandwidth τ associated to $\mathcal{L}_2$.

$$\hat{\Theta} \leftarrow \text{GA}\left(\mathcal{C}(\Theta), \Theta^{(0)}, h, \tau\right)$$

We have shown in Section 3.2.3 that the larger the value for h the smoother the cost function. Therefore, to make our approach not sensitive to the initial guess $\Theta^{(0)}$, we will use a simulated annealing framework. We use the orthogonal bandwidth h as a temperature (decreased with a geometric rate) to solve the optimisation of the cost function $\mathcal{C}(\Theta)$ defined in Equation (4.21).

The use of simulated annealing helps in converging to the global solution. However, this is not guaranteed since it depends on how the rate of the bandwith is decreased. The use of this scheme helps in achieving a better solution but involves a reduction in efficiency of the algorithm. However, we will show in Chapter 5 that when using non-isotropic modelling for the GMM the cost function becomes more robust and the convergence to the global solution is often achieved without the need of an annealing strategy.

The bandwidth τ, on the other hand, controls the influence of the prior in the cost function. A strong influence of the prior could lead the convergence of the optimisation algorithm towards the mean values $\mu_j \ \forall j$. On the other hand, if the influence of the prior is too weak the results may not truly represent the class of shape we are aiming at estimating. We found experimentally that the bandwidth τ can be set proportional to $\mathcal{L}_2(\Theta, h)$ as follows:

$$\tau = \gamma \sqrt{\frac{\mathcal{L}_2(\Theta, h)}{J}} \tag{4.22}$$

We use $\gamma = 0.5$ for all the experiments performed in this thesis. The bandwidth τ is updated along with the bandwidth h according to the simulated annealing strategy. The optimisation scheme is summarised in Algorithm 1.

4.2.4 Complexity reduction for parsimonious representation

Sub-sampling the number of vertices before modelling a GMM is a useful approach to reduce the number of Gaussians and the computation complexity involved when minimising $\mathcal{L}_2$. This can be done in a naive way by randomly selecting a subset of the vertices but the resulting GMM may no longer represent the shape properly. We propose here smart ways to downsample GMMs such that curves and surfaces remain well encoded.

Algorithm 1 Annealing framework: the final estimate $\hat{\Theta}$ does not depend on the choice of $\Theta^{(0)}$

Input: h_{max}, h_{min},β and $\Theta^{(0)}$
  Init $h = h_{max}$ and $\hat{\Theta} = \Theta^{(0)}$
  repeat
    $\tau \leftarrow \gamma\sqrt{\frac{\mathcal{L}_2(\hat{\Theta},h)}{J}}$
    $\hat{\Theta} \leftarrow \text{GA}\left(\mathcal{C}(\Theta), \hat{\Theta}, h, \tau\right)$
    $h \leftarrow \beta \times h$
  until $h = h_{min}$
Output: $\hat{\Theta}$

2D Representation

In the case of shape described using connected points in 2D, the selection of the number of Gaussians to use can be easily done by the user. The only caution is to group vertices with a similar behaviour (in the case where shapes are deforming for instance) or represent a similar description of the shape. The goal is to reduce redundancy in the description of the shape. An example of using a hand is shown in Figure 4.4).

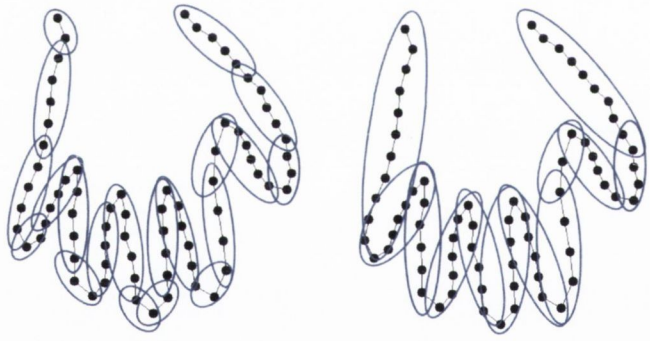


Figure 4.4: Reduction of the number of Gaussians in the mixture to represent the shape.

The effect on the density function when reducing the number of Gaussians is shown in Figure 4.6. In the top row as reference we compute the density functions using non-isotropic covariance matrices and consider all the vertices of the model (72 vertices, 71 Gaussians). The density function is computed for different bandwidths $h = 30, 20, 10$ and 5. From the second row to the bottom, we show the density functions computed when using 56, 39 and 31 Gaussians respectively (for bandwidths of $30, 20, 10$ and 5). All density functions where modelled using the same hand shape. In order to evaluate how similar the shapes are we compute the Euclidean distance between them. We use as reference the shape modelled using all vertices (second row in Figure 4.6). Results in Figure 4.5 show the distance between the shapes when the GMM is computed using 56 (blue line), 38 (green dash-dot) and 31 (black dash) Gaussians. For comparison we

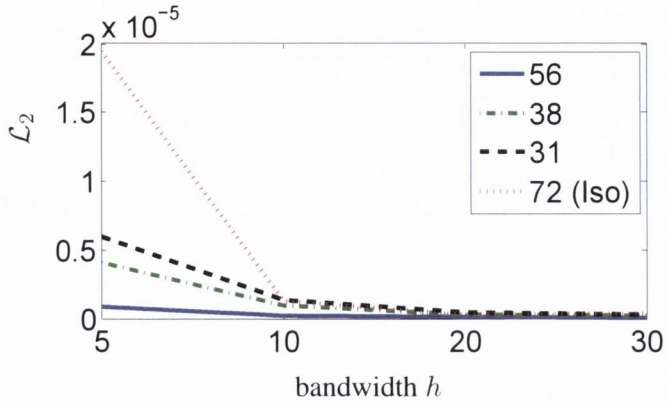


Figure 4.5: Similarity between shapes (Euclidean distance) when modelling the GM using non-isotropic covariance matrices and different numbers of Gaussians in the mixture.

include the case where all vertices are used but the GMM is computed using isometric covariance matrices (red dots). For large bandwidths the difference between the shapes is not significant. However, for smaller bandwidths even the shape modelled using 31 vertices is closer to the reference shape than when it is modelled with all the vertex (72) using isotropic covariance matrices.

3D representation

In the case of 3D data, the selection of vertices for reducing the number of Gaussians may not be trivial. There are a vast number of meshing algorithms in the literature as well as software specialising in data processing that can solve the problem more efficiently (cf. Figure 4.7). However, the main disadvantage of these software programs is that the reduction of the number of vertices is limited. Holes may appear in the mesh when the reduction of a vertex is too extreme. As an alternative, we propose to use Self-Organising Maps (SOM) as a meshing algorithms to reduce the number of Gaussians in the mixture.

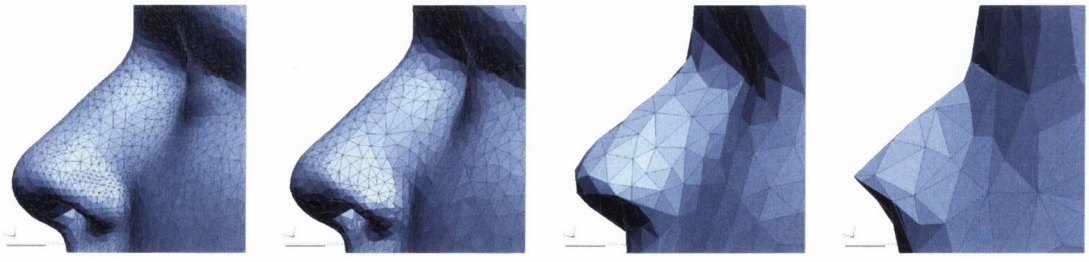
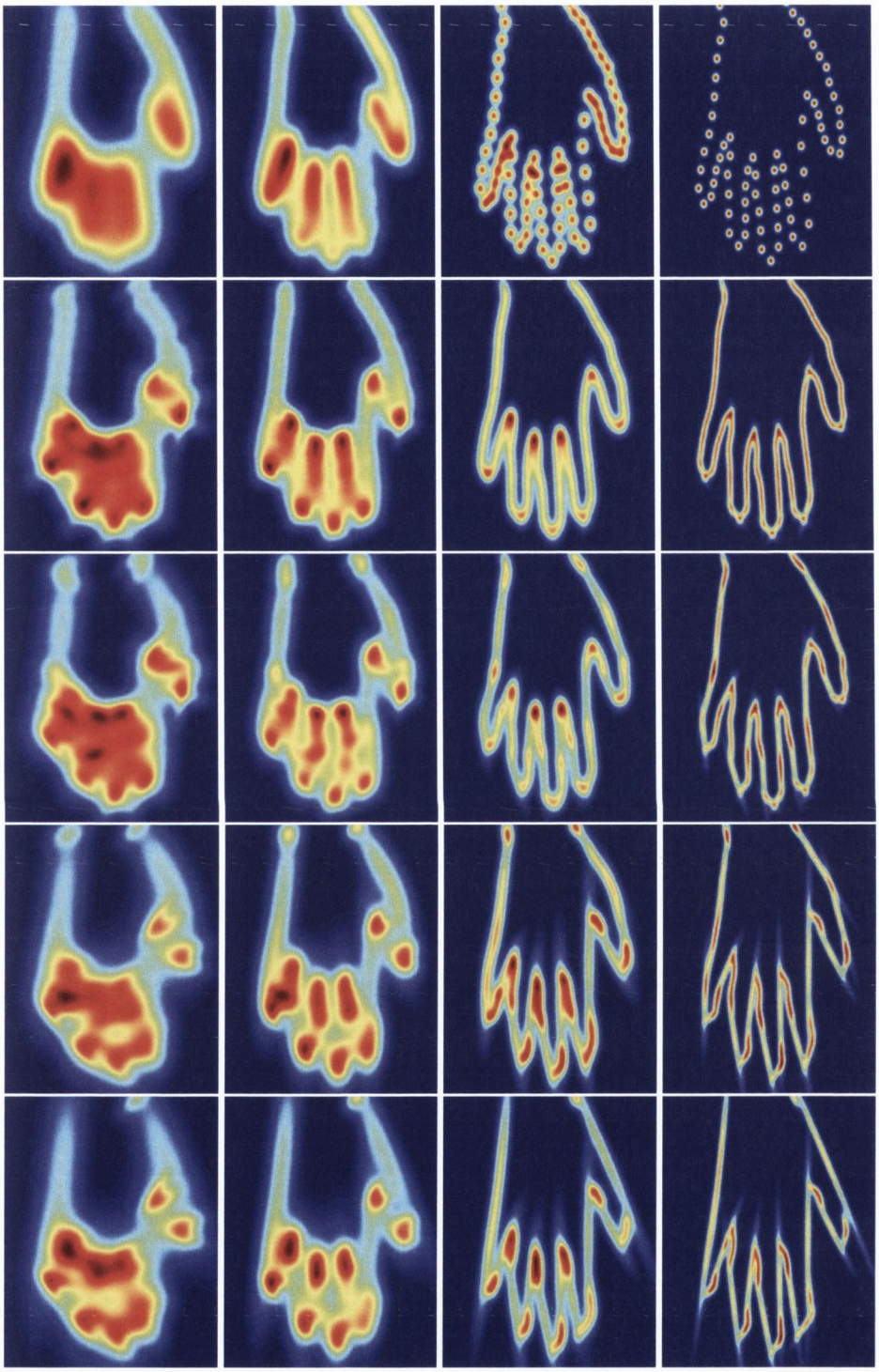


Figure 4.7: Example of a portion of the mesh of a face when described using a different number of vertices. As more vertices are used (left) a more accurate representation is produced.

SOM is a type of artificial neural network that was originally used for clustering and

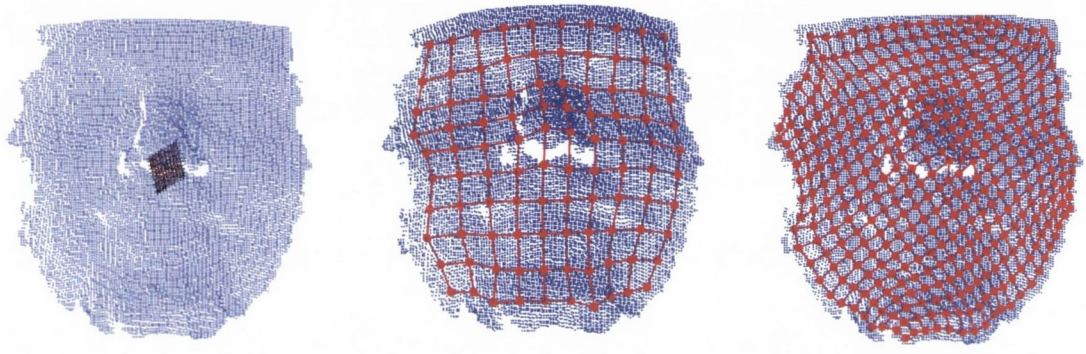


a) $h = 30$ b) $h = 20$ c) $h = 10$ d) $h = 5$

Figure 4.6: Density functions computed from the hand model when using non-isotropic Gaussians (with $n = 71$, $n = 56$, $n = 39$ and $n = 31$ kernels from top to bottom respectively).

classification of data (cf. Section 2.2.2). However, we take advantage of the topology involved in its definition and apply it for meshing purposes. We base our algorithm on the statistical self-organising method proposed by Verbeek [4]. It estimates the modes of a Gaussian mixture while preserving the spatial relationship between them. The resulting modes are used as vertices from where the final Gaussian Mixture is computed using non-isotropic covariance matrices as presented previously.

An example of the results given by the algorithm is shown in Figure 4.8. An additional advantage of this method is that the original representation of the shape does not need to already be a mesh. Any set of non-structured data points can be used as well (any point cloud).



a) b) c)

Figure 4.8: Example of the meshing process using SOM. Figure a) shows the original 3D point cloud (blue dots) and the starting position of the mesh (red). Figures b) and c) show the final solution for different numbers of vertices (100 and 400, respectively).

4.2.5 Optimising GMM from images

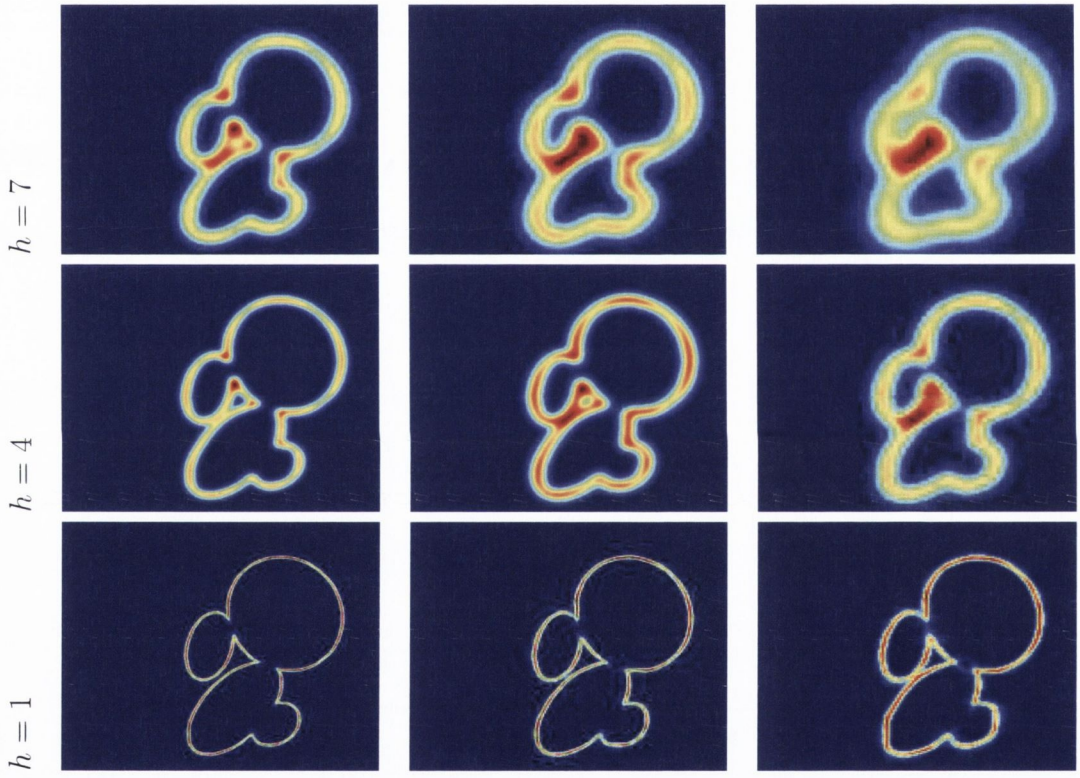
In this case, we do not have any information about the shape contained in the image. Hence, the reduction in the number of Gaussians to use in the GMM is performed by sub-sampling the data using a uniform rate. Two strategies can be implemented:

1. Sub-sampling the image: This can be done either by directly reducing the resolution of the image or by sampling it at a uniform sampling rate. An example of the reduction in the data points is illustrated in Figure 4.9b. The effect in the density function modelled from the sub-sampled data is illustrated in Figure 4.10.

2. Sub-sampling the edge map: A smarter alternative is to sub-sample the edge map. In this case we select the connected edges automatically using a standard image processing algorithm in matlab (bwboundaries). The data can then be sub-sampled

a) No sampling $n = 746$ b) Sampling the image $n = 123$ c) Sampling the edge $n = 115$

Figure 4.9: Example of reduction of data by sub-sampling. Figure a) shows the original point cloud extracted from the edge map of Figure 4.11 a). Figure b) shows the data when sub-sampling the image while in c) when sub-sampling the connected edges.



a) $n = 746, 100\%$ b) $n = 526, 70\%$ c) $n = 376, 50\%$

Figure 4.10: 2D view of the density function computed for Figure 4.11a. In the first column we show the density functions when using all data (764 points) for bandwidths of 7, 4 and 1 respectively. The second column shows the density functions of the observations with a resolution of 70% and in the last column with resolution of 50%. The number of Gaussians used to define the density functions are 526 and 372 respectively. In the three cases, the rows shows the 2D views when different bandwidths are used. At the top $h = 7$, in the middle row $h = 4$ and in the bottom row $h = 1$.

using a uniform rate along the connected edges in the edge map. An example of the resulting sub-sampling when applied to Figure 4.9b is illustrated in Figure 4.9c.

4.3 Extension to shape representation with GMM

In this section we show how to consider additional information about the shape of interest by augmenting the dimensionality of the density functions (f and g). We illustrate the case where the vertices and the curvature of the shape are considered. In the case of images, for instance, the means of the GMM are the vertices that correspond to the edge pixel locations and their curvatures $\mu_i = \{(u_i, \psi_i)\}_{i=1\cdots,n}$. The curvature for each vertex can be computed using the gradient of the image as follows:

$$\psi_i = \arctan\left(\frac{I_y(x_i, y_i)}{I_x(x_i, y_i)}\right) \tag{4.23}$$

The covariance matrix $\Sigma_i = \mathrm{Q}\Lambda\mathrm{Q}^T$ is similar to the one defined in Section 4.2.2 but augmented in one dimension such as Q:

$$\mathrm{Q} = \begin{pmatrix} \vec{n}_{1i}|\vec{n}_{2i} & \mathrm{B}^{\mathrm{T}} \\ \mathrm{B} & 1 \end{pmatrix}$$

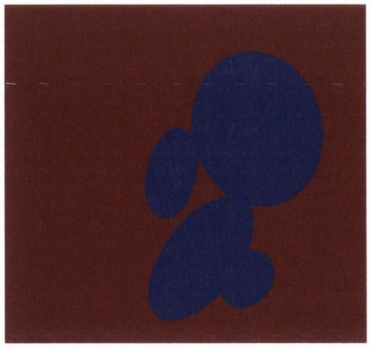
with $\mathrm{B} = [0, 0]$. The diagonal matrix of bandwidths Λ is defined as follows:

$$\Lambda = \begin{pmatrix} h_t & 0 & 0 \\ 0 & h & 0 \\ 0 & 0 & h_\psi \end{pmatrix}$$

where h_ψ is the bandwidth associated with the curvature of the shape that has been included as an extra dimension. An example of the data extracted from an image is shown in Figure 4.11. We have shown here that the domain of the density functions f and g is not limited to the spatial domain. It can be augmented by including extra information about the shape of interest. Increasing the dimension of the GMM can increase the amount of information about the shape to encode. However, it does not affect the complexity of the cost function. The computational complexity depends on the number of Gaussians used and not on their dimensionality. We have illustrated a simple example using curvature as additional information. However, it can be extended to any other information about the shape such as colour, illumination, gradient, or motion among others.

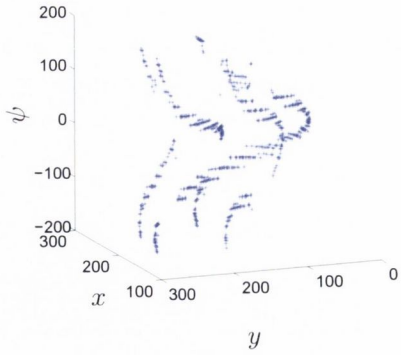
4.4 Conclusion

We have presented a Bayesian framework for shape parameter estimation. The data term in the Bayesian expression is considered as a sort of likelihood defined using the $\mathcal{L}_2$ distance between two density functions f and g. Where f and g are modelled using GMMs representing the target shape (observation) and the model shape respectively. We



a) Original image

b) Normal map



c) Observation

Figure 4.11: The set of observations are extracted from the image a). We use the edge map and the gradient of the image to compute the position and curvature of the points of interest. In b) we show the map of the normal vector associated to the edge map. In c) we show the computed observation $\{(u_i, \psi_i)\}_{i=1\cdots,n} \in \mathbb{R}^3$.

have also shown in this Chapter alternative modellings of the GMMs according to the information available about the shape (curve, surface and images). When some structure is provided (e.g connected vertices) non-isometric covariance matrices can be used for modelling the GMM. The non-isometric modelling achieves a more representative description of the shape (as density function) and allows us to reduce the number of Gaussians in the mixture without compromising the representation of the shape. We have briefly introduced the shape estimation problems to be explored in the remaining Chapters of this thesis. The experiments performed and results obtained when applying the proposed Bayesian $\mathcal{L}_2$ framework are presented in the following Chapters (Part II).

Part II

Experiments and Results

Chapter 5

Affine Transformation Estimation with Bayesian-$\mathcal{L}_2$

In this chapter we address the estimation of the affine transformation between point clouds (rotation, translation and scaling). We tackle the problem by solving the rigid transformation separately (cf. Section 5.1) and the scaling (cf. Section 5.2). We show in Section 5.1.1 that when modelling the GMMs with isotropic covariance matrices and assuming a Gaussian prior with large bandwidth, the cost function is comparable to the one used by Jain and Vermury [39] and to the Kernel Correlation registration [38]. However, in this case, a dedicated Mean Shift algorithm is implemented using an annealing strategy. The resulting algorithm is shown to improve the estimation results when comparing with state of the art algorithms [39, 38, 5]. Furthermore, in Section 5.1.2, we demonstrate that modelling non-isotropic covariance matrices in the GMM improves the transformation estimation. Moreover, it shows better performance when estimating transformation between data sets sampled at different rates.

Finally, in Section 5.3, we analyse two applications of the proposed Mean Shift algorithm. The first application, corresponds to a method for reconstructing objects from multi-view images captured using a depth sensor (Microsoft Kinect). The second application shows the performance of the algorithm when aligning data obtained from a Simultaneous Localization and Mapping system (SLAM). The results obtained in both applications show the robustness of the algorithm when aiming to recover the transformation between data sets.

5.1 Rigid parameter estimation

In this section we aim to solve the rigid transformation problem between point sets. We consider the cost function given by the Bayesian-$\mathcal{L}_2$ framework previously proposed (cf.

Equation 4.21) and rewrite it as follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{C}(\Theta) = \frac{\mathcal{L}_2(\Theta, h)}{\tau^2} + \sum_{j=1}^{J} \frac{(\theta_j - \mu_j)^2}{\sigma_j^2} \right\}$$

The latent variable to estimate corresponds to the parameters defining the rotation and translation between the point sets $\Theta = [\phi, t_x, t_y]$. We made the assumption that the prior the latent variable is modelled using a Gaussian distribution with a large bandwidth. In other words, we assume all the possible solutions have almost the same probability of occurrence. Under this assumption, the estimation problem is equivalent to minimising the $\mathcal{L}_2(\Theta)$ distance between those two density functions as follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{L}_2(\Theta, h) = \int \{ g^2(\mathbf{x}|\Theta) - 2\, g(\mathbf{x}|\Theta)\, f(\mathbf{x}) + f^2(\mathbf{x}) \}\, d\mathbf{x} \right\} \tag{5.1}$$

The terms $\int \{ g^2(\mathbf{x}|\Theta) \}\, d\mathbf{x}$ and $\int \{ f^2(\mathbf{x}) \}\, d\mathbf{x}$ do not affect the optimisation (the former is constant due the rigid transformation and the latter does not depend on Θ). Hence, the problem is equivalent to:

$$\hat{\Theta} = \arg\max_{\Theta} \left\{ \int \{ g(\mathbf{x}|\Theta)\, f(\mathbf{x}) \}\, d\mathbf{x} \right\} \tag{5.2}$$

In Section 5.1.1 we analyse the case where all Gaussians in the mixture are modelled using isotropic covariance matrices (cf. Equation 4.18). The cost function in 5.2 is similar to the one used for Jian and Vermuri [39] and Tsin et al. [38]. However, in both cases they model one density function $g(\mathbf{x}|\Theta)$ and assume the other data set corresponds to a set of samples of the same density function. In other words, they approximate the second sets by its empirical distribution. In fact, they solve the $\mathcal{L}_2$E distance instead of $\mathcal{L}_2$ (cf. Section 3.1.3). However, when all Gaussians have the same bandwidth (isotropic covariance) to model the $\mathcal{L}_2$ using a bandwidth h is equivalent to model $\mathcal{L}_2$E with a bandwidth $h\sqrt{2}$ (cf. Appendix A.2). Moreover, the bandwidth plays an important role not only in the description of the shape but also in the optimisation algorithm defined for solving the estimation problem (cf. Section 3.2.3). Hence, we propose here a dedicated Mean Shift algorithm for solving the rigid transformation. We included an annealing strategy that helps the convergence of the algorithm to the global solution. Algorithms 2 and 3 summarised the method used. For a more detailed explanation please refer to Appendix B.1.

In Section 5.1.2, on the other hand, additional information related to the structure of the points cloud is assumed. In this case, non-isotropic modelling is considered for one of the point sets. We evaluate the effects in the estimation results achieved when comparing with the case where isotropic covariance matrices are used.

Algorithm 2 Estimation of Θ.

Input: $t = 0, \Theta^{(0)}, e, M$
  repeat
    $\widehat{\delta\Theta} = \arg\max_{\delta\Theta} \mathcal{C}(\delta\Theta, \Theta^{(t)})$ (algorithm 3)
    $\Theta^{(t+1)} = \Theta^{(t)} + \widehat{\delta\Theta}$
    $t \leftarrow t + 1$
  until $\|\Theta^{(t+1)} - \Theta^{(t)}\| \leq e$ or $t > M$
Output: $\hat{\Theta} = \Theta^{(t)}$

Algorithm 3 Estimation of $\delta\Theta$.

Input: $s = 0, \Theta^{(t)}, \delta\Theta^{(0)} = 0, e, N$
  repeat
    $\delta\Theta^{(s+1)} = \mathrm{A}(\delta\Theta^{(s)}, \Theta^{(t)})\, \mathbf{b}(\delta\Theta^{(s)}, \Theta^{(t)})$
    $s \leftarrow s + 1$
  until $\|\delta\Theta^{(s+1)} - \delta\Theta^{(s)}\| \leq e$ or $s > N$
Output: $\widehat{\delta\Theta} = \delta\Theta^{(s)}$

Case a)

Methods	Distance d	Error Er	Bandwith h
Jian [39]	$9.73 \cdot 10^{-10} \approx 0$	$2.28 \cdot 10^{-5}$	Fixed
Proposed	$3.2850 \cdot 10^{-7} \approx 0$	$3.285 \cdot 10^{-4}$	Fixed

Table 5.1: Numerical assessment of the experiments displayed in Figure 5.1a. In this case we set both algorithms to use the same parameters.

5.1.1 Using isotropic covariance

Figure 5.1 show examples of the performance of our proposed algorithm compared to Jian's. We first compare in Figure 5.1a the performance of our MS algorithm when using the same settings used in Jian and Vermuri's algorithm (equal bandwidths for all gaussians in the two density functions). We compute the error between the ground truth and the estimated parameters. Results are approximately zero in both cases. For comparison we also compute the Euclidean distance d between the two density functions, which is close to zero in both cases since the sets are perfectly aligned (see Table 5.1). When the data sets (observation and reference sets) are not the same set of points (which is the standard scenario in real applications) Jian's algorithm is more likely to get stuck in local solutions. However, our algorithm converges to the global solution thanks to the annealing strategy implemented. Examples are shown in Figure 5.1b and c. For each data set we have chosen a non-uniform sub-sample. The settings used and results are reported in Tables 5.2 and 5.3.

As follows we test the performance of the algorithm under different scenarios (noise and outliers) and evaluate the efficiency of the algorithm when compared with kernel

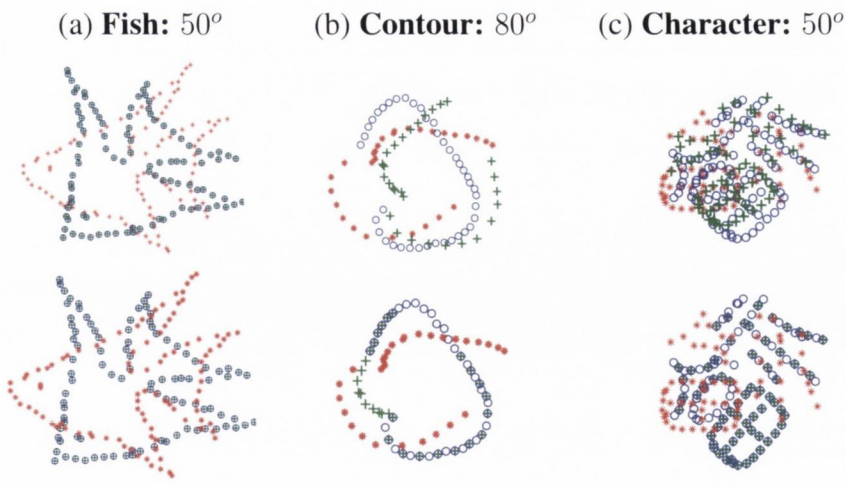


Figure 5.1: 2D data sets: alignment obtained when testing the 2D data sets Fish, Contour and a Chinese Character: Reference point set (blue circles), observation (red asterisk) and estimated solution (green cross). The top row of the figure shows results obtained using Jian and Vermuri's algorithm while in the bottom row we show the convergence obtained using our proposed MS algorithm.

Case b)

Methods	Distance d	Error Er	Bandwidth h
Jian [39]	0.3698	2.75	Fixed
Proposed	0.1629	$1.3291 \cdot 10^{-4} \approx 0$	Annealing ($h_{max} = 2, h_{min} = 0.01$)

Table 5.2: Numerical assessment of the experiments displayed in Figure 5.1b. In this case we include the annealing strategy for improving convergence of the Mean Shift Algorithm.

Case c)

Methods	Distance d	Error Er	Bandwidth h
Jian [39]	0.074	0.0454	Fixed
Proposed	0.02	$2.3799 \cdot 10^{-5} \approx 0$	Annealing ($h_{max} = 2, h_{min} = 0.01$)

Table 5.3: Numerical assessment of the experiments displayed in Figure 5.1c. This case is similar to the previous one (case b) where the annealing strategy is included in the Mean Shift Algorithm.

Correlation and the ICP algorithms [38, 36, 143].

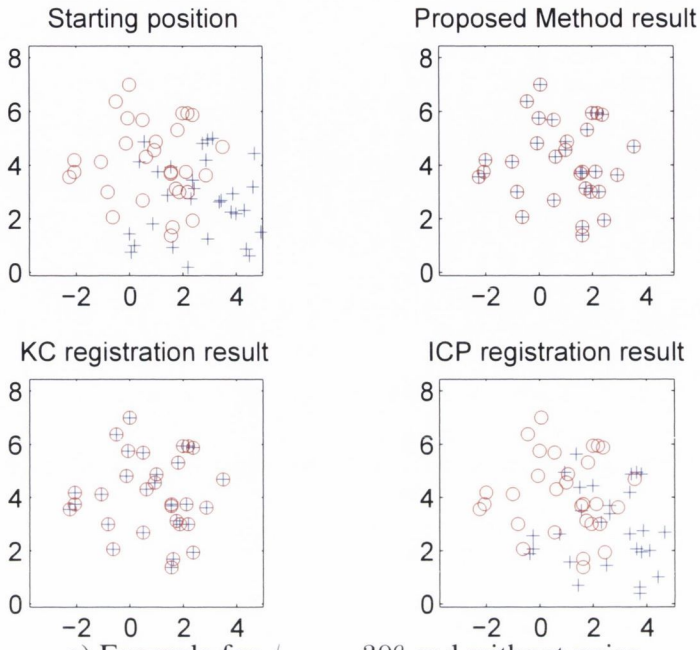
Sensitivity to noise: We generate 100 random 2D data sets containing 30 points each (cf. Figure 5.2a top left (red circles). For each data set, a second one is created by rotating it by an angle of ϕ_{GT} and perturbing it with Gaussian noise (cf. Figure 5.2a) top left (Blue crosses). We repeat the experiment using 5 different levels of Gaussian noise (from 0 to 0.2). Examples of the results obtained when estimating the rigid transformation is shown at the top right of Figure 5.2a and b. For comparison we also show the results obtained using the Kernel Correlation registration and the ICP registration (bottom left ad right respectively [38, 36]).

We evaluate the rate of convergence of the 100 data sets generated at the different noise levels. We consider that the algorithm has converged to the right solution when the error between the estimated $\hat{\phi}$ angle and the ground truth (ϕ_{GT}) is $\pm 1^o$. Results show that our algorithm has a better convergence rate than KC (cf. Figure 5.3).

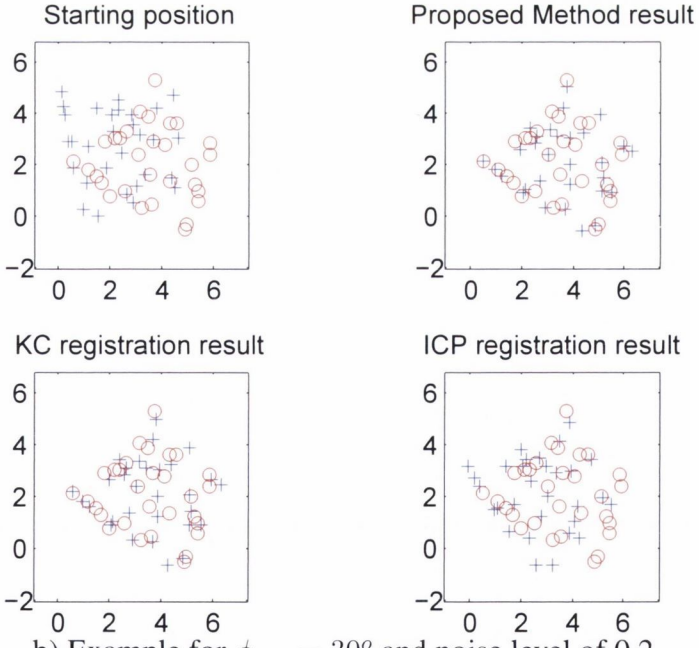
The mean of the error ($\epsilon = ||\hat{\phi} - \phi_{GT}||$) between our estimated $\hat{\phi}$ and the ground truth ϕ_{GT} is reported in Figure 5.4 with its 95% confidence interval. For the KC algorithm we use the setting proposed by the authors ($h = 2$) [38]. The bandwidths used in our proposed algorithm are $h_{max} = 3$ and $h_{min} = 0.2$. Our algorithm outperform KC registrations for all levels of noise. Additional results can be found in the Appendix C.1.1.

Robustness to outliers: Here we use the road data from [38] and generate the reference data set by adding 20% of outliers uniformly distributed. The second data set is also corrupted with 20% of outliers and rotated by an angle ϕ_{GT}. Examples of the data sets and the results obtained for rotation angles of $\phi_{GT} = 20$ and $\phi_{GT} = 90$ are reported in Figure 5.5 and Figure 5.6 respectively. In both cases the starting position is displayed at the top left and the results obtained when applying our proposed algorithm at the top right. Results for KC registration and ICP registration are shown at the bottom of each figure.

We evaluate the rate of convergence towards the right solution of our proposed algorithm. As in the previous experiment, we consider the estimated $\hat{\phi}$ to have converged to the ground truth ϕ_{GT} when the error between them is less than one degree ($\|\hat{\phi} - \phi_{GT}\| < 1^o$). Results obtained when running the experiment 100 times (100 different data sets) are reported in Table 5.4. For comparison we also report the results obtained when using KC and ICP algorithms. In all the cases evaluated ($\phi_{GT} = 20^o, 30^o$ and 90^o), our proposed algorithm outperforms both competitors. The only parameters to set in these experiments are the bandwidth (proposed algorithm and KC) and the threshold in the case of ICP. For our MS algorithm we use $h_{max} = 7$ and $h_{min} = 2$. The KC algorithm is used using $h = 5$ which is the value proposed for this data set by the authors.

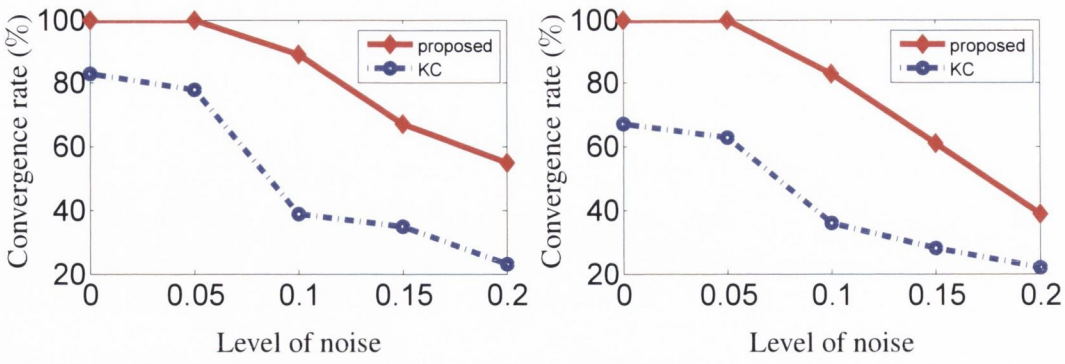


a) Example for $\phi_{GT} = 30^o$ and without noise



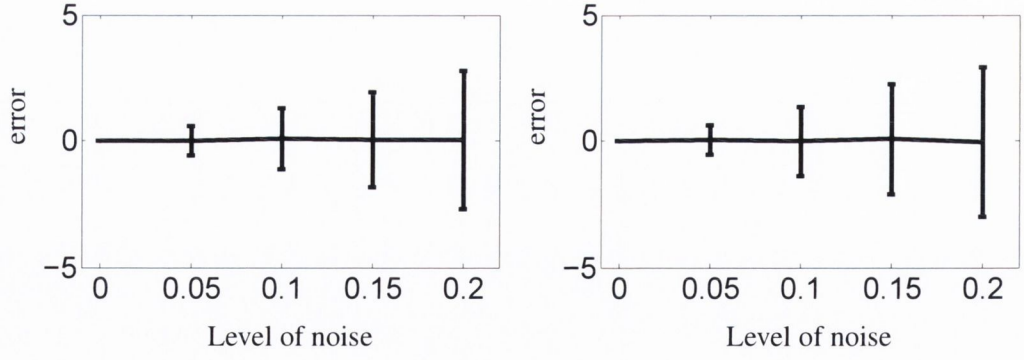
b) Example for $\phi_{GT} = 30^o$ and noise level of 0.2.

Figure 5.2: Examples of the experiment performed for an angle of rotation of $\phi_{GT} = 30^o$. Results are displayed for our proposed algorithm, KC registration and ICP registration.



a) $\phi_{GT} = 20^o$ b)$\phi_{GT} = 30^o$

Figure 5.3: Convergence rate (%) of the estimated solution (y-axis) when perturbing the data with different noise levels (x-axis) using our proposed algorithm (red line) and when using the Kernel Correlation algorithm (blue dot-dash line). In a) we show the results for a rotation angle of $\phi_{GT} = 20^o$ and in b) $\phi_{GT} = 30^o$



a) $\phi_{GT} = 20^o$ b) $\phi_{GT} = 30^o$

Figure 5.4: Error mean with 95% of confidence of the estimated rotation when data is perturbed using different levels of noise (from 0 to 0.2). In a) we show the target angle or rotation is $\phi_{GT} = 20^o$ and in b) $\phi_{GT} = 30^o$

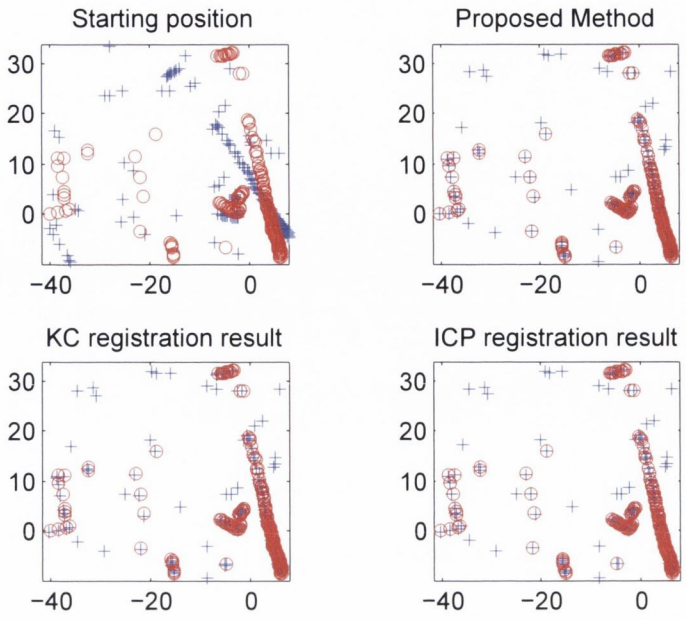


Figure 5.5: Examples when both sets are rotated by 20^o.

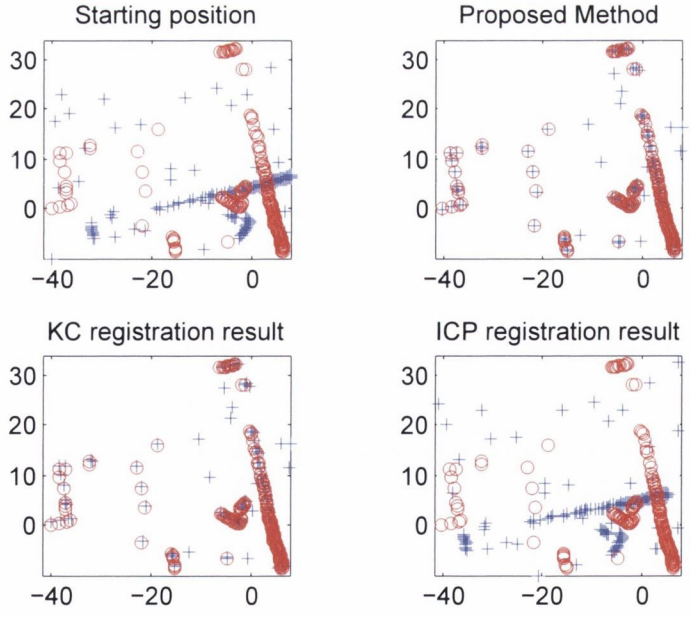


Figure 5.6: Examples when both sets are rotated by 90^o.

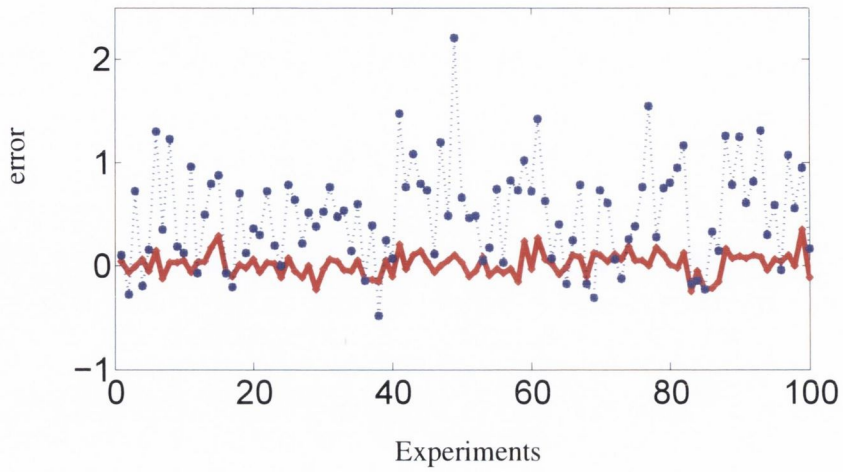


Figure 5.7: The error obtained for 100 experiments when using our proposed algorithm (red line) and the KC algorithm (blue dots). The mean of the error for the 100 runs when using our proposed algorithm is 0.0186 (degrees) and the standard deviation is 0.1105. In the case of the KC algorithm the mean and standard deviation are 0.5024 and 0.4842 respectively.

In the case of the ICP algorithm the threshold is not that trivial to set. However, we use the same parameters used in the experiments performed by [38] for the same data set.

Angle (degrees)	Proposed method	KC Registration	ICP Registration
20^o	100%	91%	77%
30^o	100%	86%	29%
90^o	100%	98%	0%

Table 5.4: Rate of convergence towards the ground truth when considering an error of $\pm 1^0$

Figure 5.7 shows the error obtained for the 100 experiments when the data sets are rotated by $\phi_{GT} = 30^o$. The red line corresponds to our proposed algorithm and the blue dots to the KC registration algorithm.

Run-time performance: The use of annealing in the MS algorithm increases convergence towards the global solution but also increases the time needed for estimating the parameters. The computational complexity of the algorithm will depend then on the bandwidth used and the number of Gaussians in each mixture (f and g).

As reference for the performance of the proposed algorithm, we report the time in seconds in Table 5.5 that was needed for convergence to the correct solution. The time increases considerably as the number of points in the set increases. The ICP algorithm is

the most efficient (fast). However, its convergence toward the right solution is not always achieved. We explore as follows a non-isotropic modelling for the GMM and its effect in the efficiency of the optimisation algorithm.

n^o Points (n_fxn_g)	Proposed method	KC Registration	ICP Registration
5x5	0.051	0.094	0.002
10x10	0.216	0.129	0.001
30x30	2.089	0.310	0.004
50x50	6.311	0.432	0.007
100x100	31.602	0.971	0.024
500x500	630.388	4.025	0.458

Table 5.5: Run-time performance of the algorithm measured as the time it takes the algorithm to converge (in seconds).

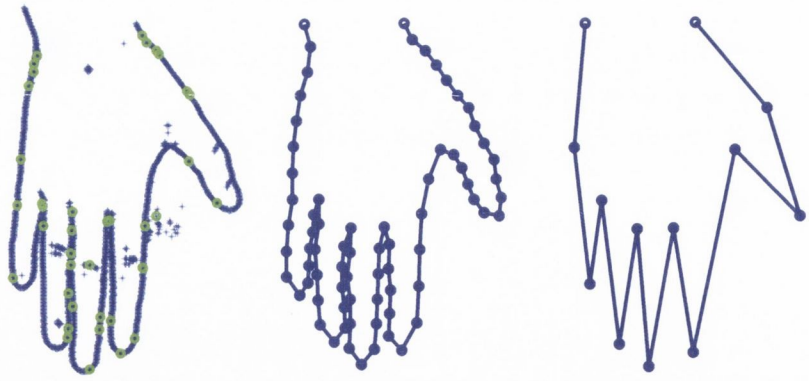
Proposed method	KC Registration	ICP Registration
$\max\{n_f^2, n_f n_g, n_g^2\}$	$n_f n_g$	$n_f + n_g$

Table 5.6: Computational complexity of the algorithm.

This comparison may not be fair since the implementation of the algorithms are not necessarily done with the same degree of optimisation. Our implementations were done using Matlab and no special effort was made on reducing the running time. A more fair comparison is to evaluate the computational complexity instead of the running time. Table 5.6 shows the order of complexity of each method.

5.1.2 Using non-isotropic covariance

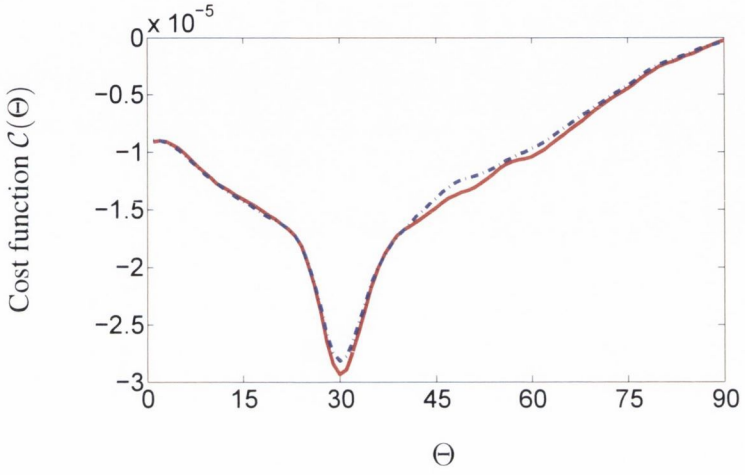
We assume here that one of the data sets is a connected set of points. Therefore, non-isotropic covariance matrices can be modelled for its GMM. The second data set, on the other hand, is modelled using isotropic covariance matrices since there is no knowledge about its structure. Let us consider the example in Figure 5.8. Figure 5.8a shows, in blue, the data sets considered as observations $n_f = 2421$. Figure 5.8b shows the full model of the same class of shape ($n_g = 72$) and in c) a sub-sampled version of the model ($n_g = 13$).



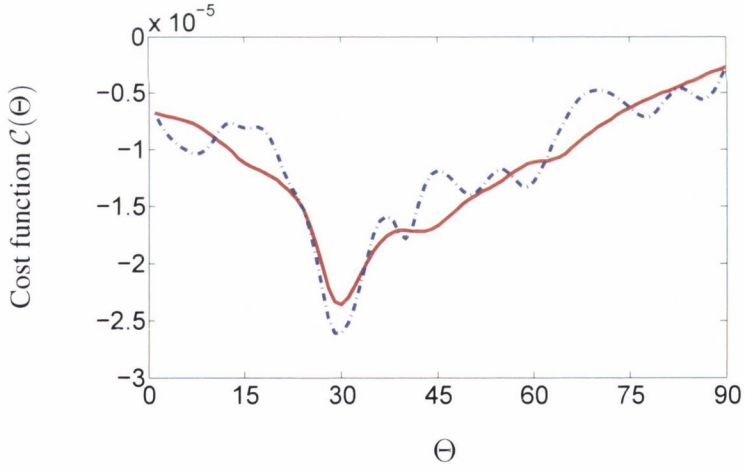
a) Observations b) Full model c) Sampled model

Figure 5.8: Example of a set of observations a) and model of the hand b) used for estimating the rigid transformation when modelling GMM using non-isotropic covariance matrices. In c) we show a sub-sampled version of the hand model.

We rotate the observation with respect to the hand model by $\phi_{GT} = 30^o$ and evaluate the effect on the cost function when using different modellings for the GMM and when sub-sampling the data sets. The first modelling corresponds to the case where both GMM (f and g) are defined using isotropic covariance matrices. The second one corresponds to the case where one of the data sets is modelled using non-isotropic covariance matrices (hand model). Figure 5.9 shows the resulting cost functions computed for angles in between 0^o to 90^o for a bandwidth $h = 10$. The blue dash-line corresponds to the isotropic modelling and the red line to the non-isotropic modelling. Figure 5.9a shows the cost functions when all points are used for modelling the GMM. Figure 5.9b shows, on the other hand, the cost functions for a sub-sampled version of the observations (green dots in Figure 5.8 a) and a sub-sampled version of the hand model (cf. Figure 5.8 c). The total number of Gaussians evaluated are then $n_f \times n_g = 38 \times 13$ which correspond to a small fraction of the original data sets ($n_f \times n_g = 2421 \times 72$). However, the resulting cost function of the sub-sampled data when modelling using non-isotropic covariance matrices preserves its convexity. On the contrary, when both GMM are computed using isotropic covariance matrices, the cost function shows an increment of local solutions which may affect the convergence of the algorithm to the global solution. We evaluate the convergence of the algorithm when running 100 experiments using different initialisations for the optimisation algorithm (randomly generated in the range $[0^o, 60^o]$). We use the same settings for both modellings for the bandwidth ($h = 10$). Note that we do not use annealing for this experiment. Results of the convergence of the algorithm towards the ground truth is reported in Table 5.7 for different samples taken randomly from the observation ($n_f = 120, 80$ and 38). For this experiment we use a sub-sampled version of the hand model $n_g = 13$ (cf. Figure 5.8c). Results show that the non-isotropic modelling helps in the convergence of the algorithm. Moreover, when using a small set of samples



a) Using all the observations and the full model



b) Using a sub-sampled version of the observations and model

Figure 5.9: Cost function computed when both GMMs are modelled using isotropic covariance matrices (blue dash line) and when one of them is modelled using non-isotropic modelling (red line).

Method	$n_f = 120$	$n_f = 80$	$n_f = 38$
Isotropic	0%	0%	0%
Non-Isotropic	98%	95%	97%
Avg. Time	$54sec$	$25sec$	$15sec$

Table 5.7: A comparison of the convergence of isotropic versus non-isotropic modellings for drastically sub-sampled data sets.

from the observations (randomly chosen) the non-isotropic modelling shows a better performance than when modelling both GMMs using isotropic covariance matrices. Figure 5.10 shows two examples of the results obtained in this experiment.

Figure 5.10: An example of results obtained when sub-sampling the data sets and when using isotropic (green dash line) and non-isotropic modelling (red line). The blue dots correspond to the full observation.

5.2 Scaling parameter estimation

We use here the cost function proposed in Equation 4.21 and rewrite it as follows:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{C}(\Theta) = \frac{\mathcal{L}_2(\Theta, h)}{\tau^2} + \sum_{j=1}^{J} \frac{(\theta_j - \mu_j)^2}{\sigma_j^2} \right\}$$

The latent variable only considers the scaling parameters in both directions $\Theta = [s_x, s_y]$. We evaluate the performance of the estimation algorithm when using a non-informative prior and when using a Gaussian distribution as a prior for the parameters to estimate.

5.2.1 Non informative prior

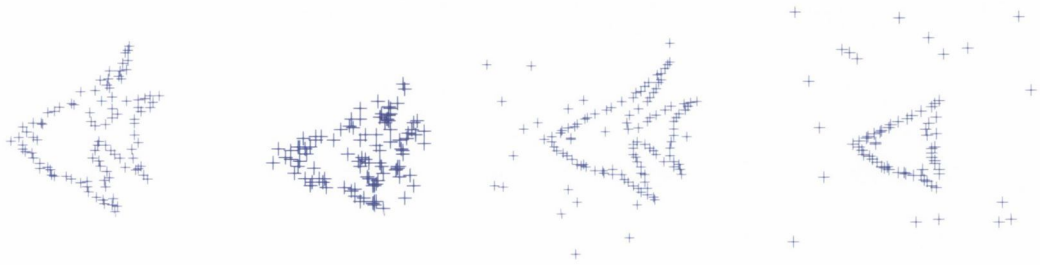
In this section we consider a Gaussian distribution with a large bandwidth as prior for the latent variable. Under this assumption the estimation problem can be interpreted as

minimising the $\mathcal{L}_2$ distance. Moreover, in this case, all the solutions are equally probable. To test the algorithm we use the Fish data set and evaluate the following cases:

1. *Sensitivity to noise*: The data set is perturbed using two levels of noise generated by a Gaussian distribution with zero mean and standard deviation of 0.01 and 0.03 respectively. An example of such a data set is shown in Figure 5.11 a and b.

2. *Robustness to outliers*: In this case we add 20% of outliers uniformly distributed as shown in Figure 5.11 c.

3. *Occlusion*: The occluded data set is generated by discarding the points in the tail of the Fish. We also add in this experiment outliers as in the previous case (cf. Figure 5.11 d).



a) Noise $\sigma = 0.01$ b) Noise $\sigma = 0.03$ c) Outliers d) Occlusion

Figure 5.11: Examples of the data set used for testing

For each case, we scale the data sets in both directions. Each scaling parameter considered as ground truth s_{xGT} and s_{yGT} is generated from the following distribution $\mathcal{N}(x; \mu = 1, \sigma = 0.5)$. We run 100 experiments for each case and evaluate the convergence towards the ground truth. The settings for all the experiments are $h_{max} = 1$ and $h_{min} = 0.01$. The results obtained using our proposed algorithm when evaluating the scaling of the original data set (without noise or outliers) and the other 4 cases described above are reported in Table 5.8. The algorithm shows a high rate of convergence towards the ground truth for the original data and when perturbed with noise and outliers. However, results obtained when occluded data is considered shows only 85% convergence for the scaling parameter associated with the $y-$axis. This is mainly due to the fact that the shape remaining (Fish without a tail) can either fit the body of the fish or the tail. Furthermore, solutions where the shape is partially fitted as it is shown in Figure 5.12 c, are often reached, since it is a valid solution of the algorithm but not necessarily the correct solution. Figure 5.12 show several results. Additional results are presented in Appendix C.1.3.

Scale	Original	Noise (0.01)	Noise (0.03)	Outliers	Occlusion & Outliers
s_x	100%	100%	99%	100%	100%
s_y	100%	100%	97%	100%	85%

Table 5.8: The rate of convergence towards the ground truth when considering error of 5%. We evaluate the cases where: no changes are made in the data (original), data is perturbed with noise (levels: 0.01 and 0.03), data is perturbed with outliers, and occluded data. In all cases 100 experiments were run and the number of successes are reported in the table.

5.2.2 Gaussian Prior

In this section, we assume that an informative prior for the latent variable is available, such as:

$$s_x \sim \mathcal{N}(s_x; \mu_{s_x}, \sigma_{s_x}) \tag{5.3}$$

$$s_y \sim \mathcal{N}(s_y; \mu_{s_y}, \sigma_{s_y}) \tag{5.4}$$

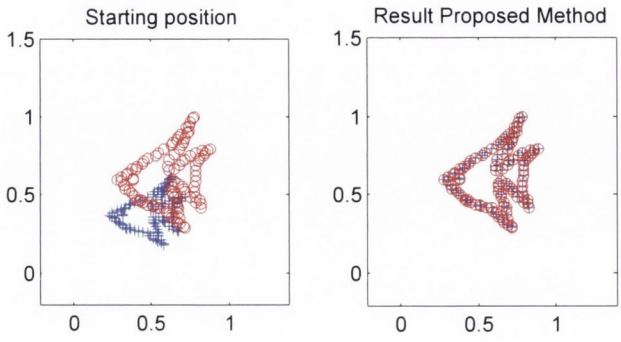
We test the robustness of the algorithm for the case of the occluded data. Two different occlusions are considered for the Fish data set. Results of the estimation of 100 experiments are reported in Table 5.9. In this case there are three parameters to set: h_{max}, h_{min} and the bandwidth associated with the likelihood τ (cf. Equation 4.10). This parameter controls the trade-off between the likelihood and the prior distribution.

Scale	Occlusion 1	Occl. & Outliers 1	Occlusion 2	Occl. & Outliers 2
s_x	100%	84%	100%	95%
s_y	100%	100	100%	93%

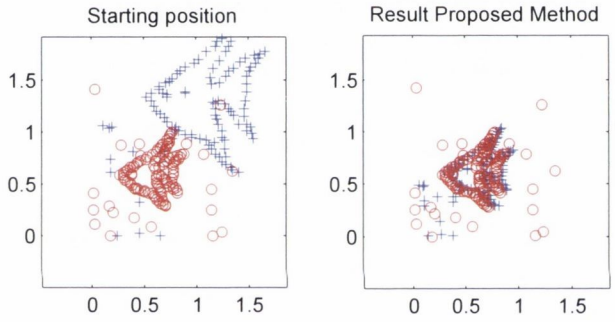
Table 5.9: The rate of convergence towards the ground truth when considering an error of 10% and 5%. The bandwidth for this experiment was set to $h_{max} = 0.5$ and $h_{min} = 0.01$.

5.3 3D reconstruction applications

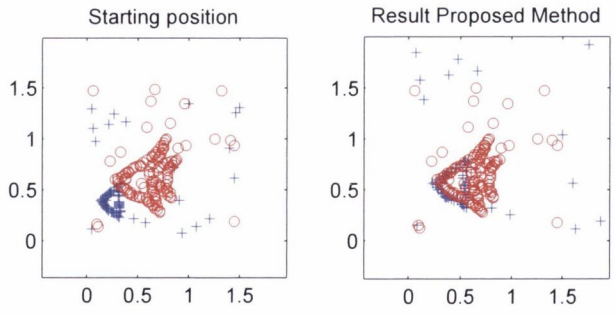
We present two applications in this section for the rigid transformation algorithm discussed in Section 5.1 but extended to 3D data sets. In the first application we aim at improving the 3D shape reconstruction when capturing data using the Kinect sensor. In the second application, we use data captured using a Lidar.



a) Original data



b) Outliers



c) Occlusion

Figure 5.12: Examples of the results obtained when testing our algorithm under different conditions.

5.3.1 Multi-view reconstruction

The goal in this experiment is to recover an accurate 3D reconstruction of an object captured using a Kinect sensor. The Kinect sensor is based on structured light technology (cf. Section 2.3.2). Therefore, the resulting 3D object is quite noisy. We use an algorithm that uses a multi-view strategy to estimate the 3D shape by accumulating the information of a sequence of depth images captured from multiple viewpoints (see experiment setup in Figure 5.13). No prior information is available about the object in view. All the different views are aligned with each other using a extension in 3D of the algorithm discussed in Section 5.1 and presented in Appendix B.1.

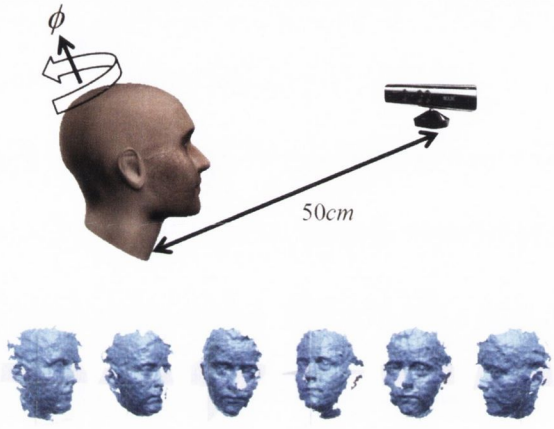


Figure 5.13: Experiment setup. The object to be scanned is placed as close as possible to the Kinect sensor (approximately 50 cm). The object (here a head) is rotated in order to capture different points of view.

We test our approach with different objects: Gnome (height 32cm), Duck (height 26cm) and two human faces (assumed to be rigid in this experiment). In the case of the human faces, the experiment was done according to the setup displayed in Figure 5.13, while for the other objects a turntable was used. The distance between the object and the Kinect sensor is approximately $50-60$cm (the minimum distance allowed by the sensor). For comparison, a 3D Minolta vivid 700 laser scanner was used for acquiring a reference 3D shape of the objects.

After estimating the rigid parameters between views the final point cloud is computed (it corresponds to all views aligned altogether). This point cloud can be presented as a mesh using Rapidform XOR. In Figure 5.14, several 3D reconstructions of one face (profile view) are presented. The first reconstruction (left) has been generated from one depth image from the Kinect sensor. The following 3D reconstructions have merged several point clouds ($n = 4, 6, 9$ views are merged). We can visually appreciate how the noise present in a single capture by the Kinect sensor can be reduced when more point

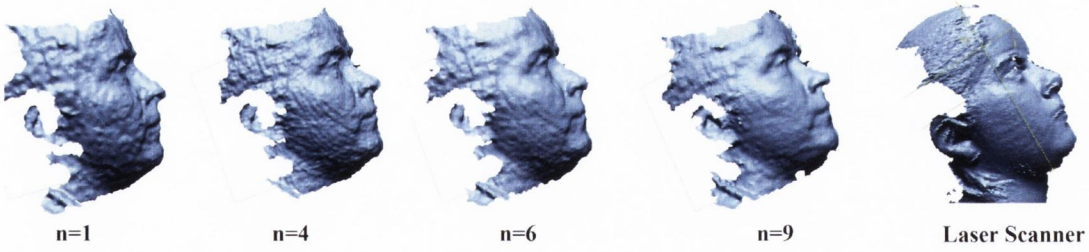


Figure 5.14: Results obtained when aligning different views captured using the Kinect sensor. The noise is reduced as the number of views n merged increase. A laser scanner acquisition is also shown for visual comparison purposes

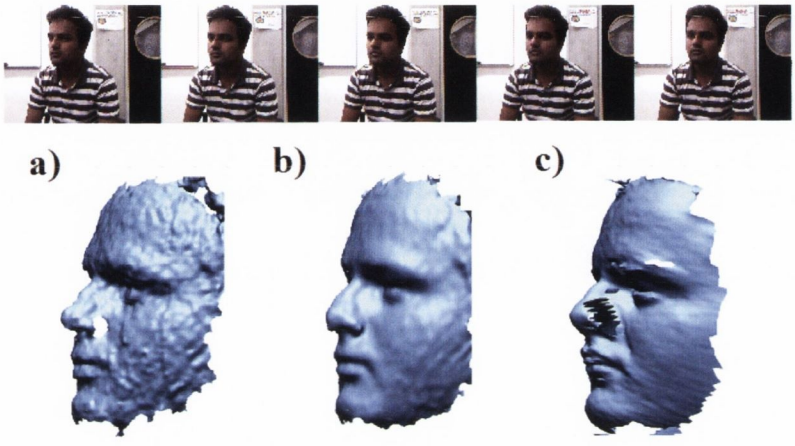


Figure 5.15: Kinect scan alignment for a face. The top row shows a sequence of images captured from different points of view. Bottom row: Mesh created from a) a single acquisition, b) multiples views and c) face captured using a laser scanner.

clouds are merged together. For comparison, the 3D reconstruction from a single scan recorded with a 3D scanner is also shown (right).

Similar results are shown for a face in Figure 5.15. The final reconstruction of the face compares with the laser scanner acquisition (average error of 1.02mm). In Figure 5.16, we shows some results obtained for the Duck and Gnome. We quantify the average distance between the ground truth (scanner) and the Kinect reconstruction after merging a set acquisition. The average noise is reduced from 2.25mm (for a single acquisition) to 1.54mm (for multiples views). In addition to noise reduction, the multi-view strategy also allows us to recover parts of the shape that are hidden in some specific views but seen in others.

5.3.2 Simultaneous localization and mapping

Simultaneous Localization and Mapping (SLAM) is a technique that aims at reconstructing 3D maps of unknown environments. A set of point clouds are captured by using

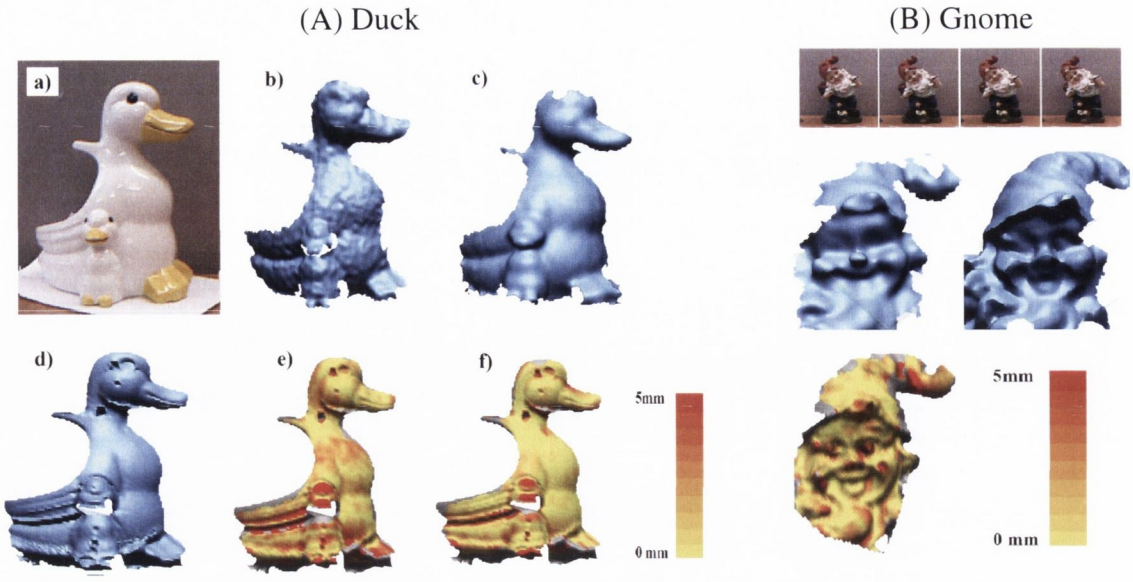


Figure 5.16: a) the picture of the object (Duck) and d) laser scan respectively (ground truth). In b) the reconstruction with one acquisition using the Kinect sensor and e) its error w.r.t. the laser scan has an average of 2.25mm. When merging 15 views together c) the average error f) is reduced to 1.54mm. Details of the Gnome face the average error of the reconstructed shape when four images are merged is 1.7mm.

robots and autonomous vehicles equipped with 3D scanners. Each scan (as a point cloud) is captured in a local coordinate framework given by the position of the scanner at that time (robot/vehicle position). In order to build the global map, all the scans must be matched on a general coordinate framework. A well known strategy for solving this problem is to treat every consecutive pair of scans independently and then to calculate the transformation between them by using points registration techniques.

In this section we test our algorithm for rigid transformation as an alternative to solving the registration problem between consecutive scans captured using SLAM. We use the *Hannover1* database[1] for all our experiments in this section. It contains 468 3D scans, each of them with approximately 20000 data points. In addition to the scans it also contains the position estimates for each scan in a common reference frame. We use these estimated positions as a reference for comparison and validation of our algorithm.

Similarly to the Kinect application, we consider two consecutive scans S_z and S_{z+1}. Using one of them as a reference, we aim at estimating the parameters (rotation and translation) that align them. For all the experiments we consider an isotropic modelling of the density functions (cf. Section 4.18). The bandwidth for all the Gaussians of the two density functions are identical ($h = h_i = h_k$, $\forall\ i \in n'$ and $\forall\ k \in n$). We choose an initial value of $h_{max} = 100$cm. This value is decreased iteratively until $h_{min} = 1cm$ thanks to the annealing strategy.

[1]Captured by Oliver Wulf from the Leibniz University and available at `http://kos.informatik.uni-osnabrueck.de/3Dscans/`.

To define the weight of each Gaussian (w_i) a more careful inspection of the data contained on the scans must be done. Figure 5.17 shows two scans from different points of view. The distribution of the points captured around the sensor is more dense than the points captured along the full scene. If we analyse the amount of points in this area (in particular on the ground) we observe that they represent a significant percentage of the full data set (approximately 40.6%).

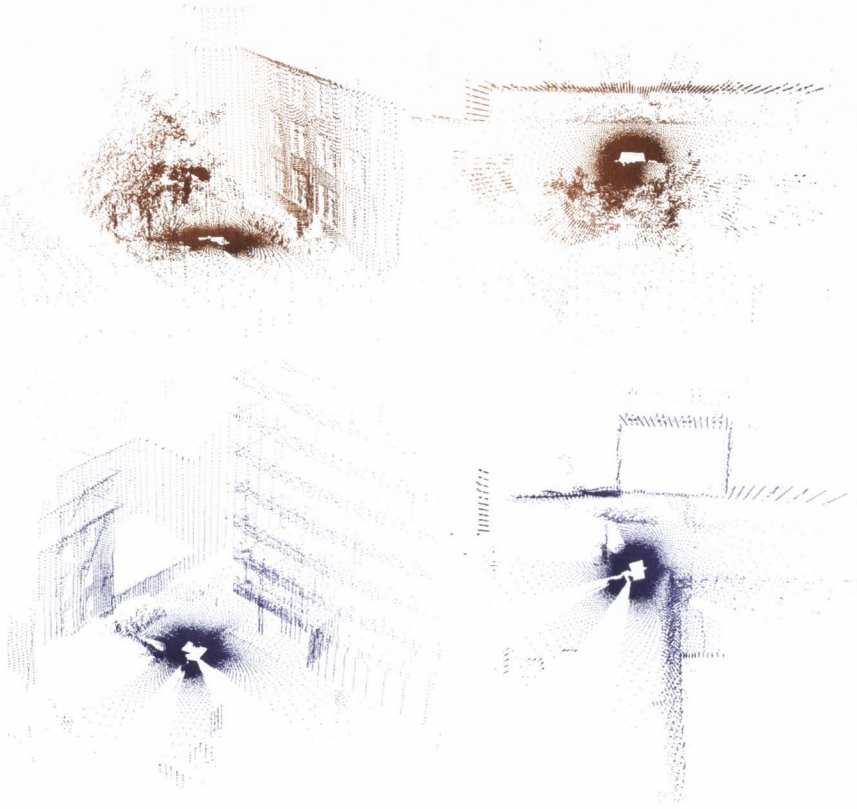


Figure 5.17: Two scans displayed from different viewpoints

The annealing strategy implemented in our algorithm helps in the convergence to the global solution. However, the two shapes ($S_z$and S_{z+1}) are not exactly the same (they offer different points of view of the environment captured). Then, the global solution may not necessarily be the right solution we are looking for. For instance, since the ground represents an important percentage of the full data and is already aligned between the scan (all shapes by default are centered at the robot/sensor position), an equally weighted modelling for the density function may converge to zero for the rotation and translation. However, this can be avoided by modelling the weight of each kernel in the density functions according to the structures or shape we are interested in being aligned.

In this particular case (the *hanover1* data base) we are interested in aligning the building in the scene. This suggests a weighting strategy that gives more importance to the

points corresponding to walls and reduces the weight to points on the ground. A simple implementation is to set all the weights of the points on the ground to zero ($w_i = 0 \; \forall i \in$ ground of S_z and $w_k = 0 \; \forall k \in$ ground of S_{z+1}). This modelling of the density function helps us to ensure a convergence of the algorithm to the right solution and it also reduces the total computation needed (the computation time is proportional to the number of points on the data set).

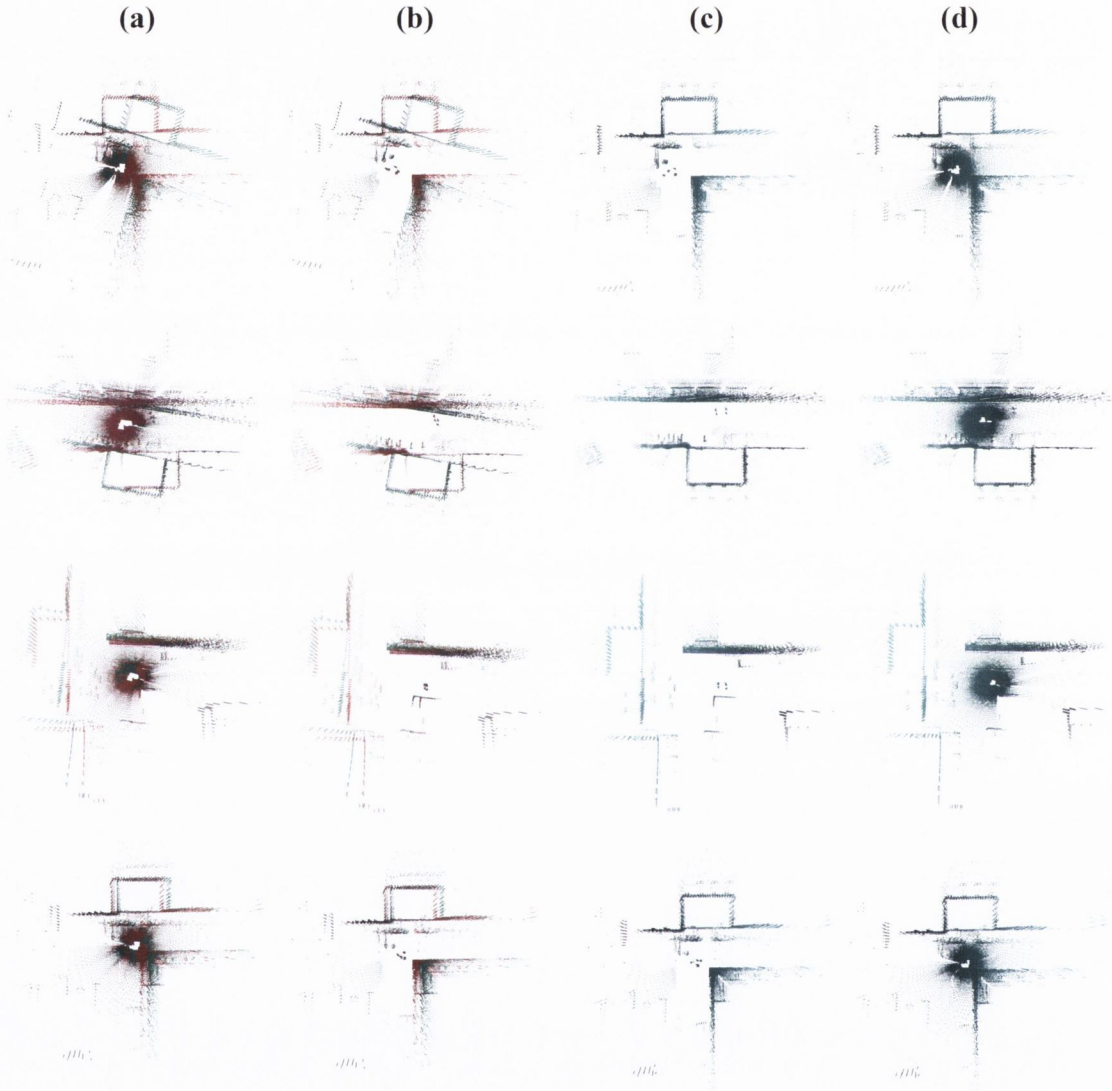


Figure 5.18: A set of examples of the alignment between scans. (a) two consecutive scans (input data), (b) the two scans after removing the ground, (c) the two scans aligned according to the transformation parameters estimated with our proposed algorithm and (d) is the estimated transformation applied to the full scan.

An example of the results obtained are shown in Figure 5.18. In the first two columns (a and b), four examples of pairs of consecutive scans are displayed. The two scans using the full data points are shown in a) while in b) we present the data on the scans after

Figure 5.19: A set of scans aligned together. The figure on the left shows a set of 6 consecutive scans. The middle figure shows the results after estimating the transformation parameters between the scans. The same result from another point of view is displayed in the right figure

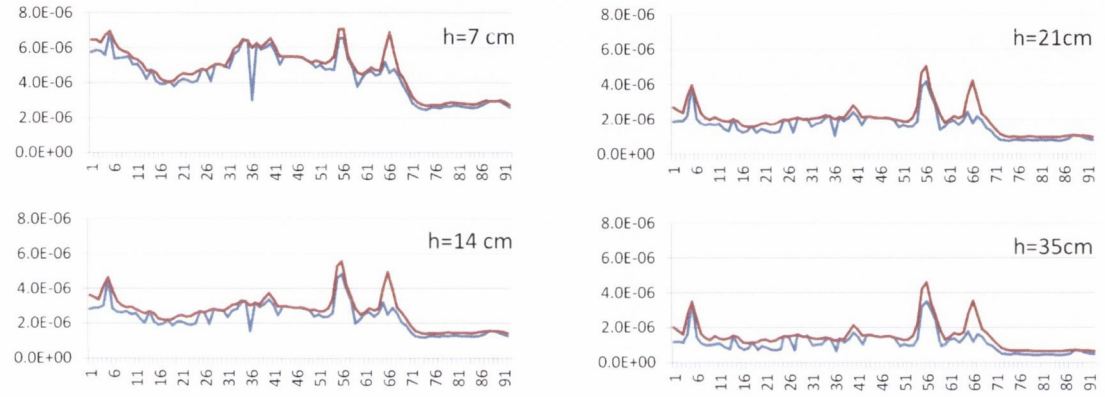


Figure 5.20: The distance between the shape when aligned using the proposed method (blue line) and when aligned using the ground truth (red line). Results obtained when modelling the density function using isotropic bandwidth $h = 7, 14, 21$ and 35.

removing the points on the ground (setting their weights to zero). The alignment of the two scans after compensating the transformation by our estimated parameters is shown in column c). Similar results are shown in d) by applying the estimated transformation to the full scans (using all data points). As can be seen in the examples, the pair of scans have been successfully aligned to each other. Similar results can be seen in Figure 5.19, where the alignment between a set of 6 scans after estimating consecutively the transformation between pair of scans is shown. Different views of the reconstructed environment are displayed in the middle and on the right side of the figure.

A quantitative evaluation of the results is performed by comparing with the alignment provided by authors of the data base (ground truth). We use the Euclidean distance between density functions as a measure of similarity between the consecutive scans (as a measure of the error). The smaller the distance between the density functions, the better the alignment between them. We run the experiment for 92 pairs of scans. The density

function of each scan is modelled using the same bandwidth (h) for all the Gaussian kernels centred on each point on the data set. Using the same modelling we compute the distance between the density functions of the scans aligned using the transformation given by the authors of the database.

In Figure 5.20 we compare the results using different bandwidths on the modelling of the density function ($h = 7, 14, 21$ and 35cm). The blue curve corresponds to the Euclidean distance calculated between the scans aligned using our proposed method, while the red curve was computed using the alignment provided from the database. For all the bandwidths modelled, the distance between the scans using our proposed method is in general smaller than the reference we are comparing with. These results suggest a better performance of our aligning algorithm.

In order to demonstrate the significance of these results we perform a paired difference test. We are interested in testing the hypothesis that the distance between the scans when the reference transformation is applied is greater than when transformation is estimated using our proposed method (with a confidence of 99.9%, $\alpha = 0.001$). Using the computed distance between the density functions of the pair of scans shown in Figure 5.20 (for the 92 pairs of scans) we calculate d as the difference between the distance calculated with the reference alignment and our method. Then, we state our hypothesis as follows:

$$\begin{aligned} H_o: &\quad \mu_d = 0 \\ H_1: &\quad \mu_d > 0 \end{aligned}$$

Table 5.10 shows the values obtained for the mean $\bar{d}$ and standard deviation σ_d calculated from the $n = 92$ samples for different bandwidths. We compute the t-value for each experiment as $t = \frac{\bar{d}-0}{\sigma_d/\sqrt{n}}$. The critical p-value for $\alpha = 0.001$ and 91 degrees of freedom is 3.182.

For all the sets of experiments performed, $t > 3.182$ (see Table 5.10) so we reject H_o. This allows us to claim with 99% confidence that our proposed algorithm performs better than the aligning provided as reference by the authors of the database. This result validates our algorithm as an alternative for SLAM reconstruction. A graphical example of these results is shown in Figure 5.21 where a close zoom over the two scans shows the errors in alignment given by the reference parameters (right) compared with the result obtained with our method (left).

Table 5.10: Paired difference test: The table shows the value for the mean ($\bar{d}$), standard deviation (σ_d), number of samples (n^o) and the t-value calculated as $t = \frac{\bar{d}-0}{\sigma_d/\sqrt{n}}$

Value	$h = 7cm$	$h = 14cm$	$h = 21cm$	$h = 35cm$
$\bar{d}$	$3.31E^{-07}$	$3.45E^{-07}$	$3.55E^{-07}$	$3.67E^{-07}$
σ_d	$4.29E^{-07}$	$3.66E^{-07}$	$3.59E^{-07}$	$3.60E^{-07}$
n^o	92	92	92	92
t	7.385	9.037	9.475	9.778
n^o	92	92	92	92
H_o :	*False*	*False*	*False*	*False*

(a) (b)

Figure 5.21: Details of the alignment between buildings. a) using our proposed algorithm for aligning the scans and b) using ground truth.

5.4 Conclusion

We have proposed a method based on the Bayesian-$\mathcal{L}_2$ framework for estimating affine transformation parameters between data sets. We solve the rigid transformation and scaling. When modelling the GMM using isotropic covariance matrices with identical bandwidths the problem to solve becomes similar to two state-of-the-art algorithms: Jian and Vermuri [39] and Kernel correlation registration [38]. However, in this case we implement a MS algorithm that uses an annealing strategy that makes the algorithm less sensitive to the starting guess. This strategy is shown to outperform the competitors, including the well known ICP algorithm. Additionally we show that the bandwidth used to model the GMM *does matter*. It plays an important role in the convergence of the algorithm and it helps to improve the efficiency of the optimisation method. It allows us to sub-sample the data sets without altering the convergence towards the right solution. The setting of the bandwidth h_{max} can be arbitrarily chosen as a large value. The minimum bandwidth h_{min} on the other hand, is chosen according to the error one is willing to tolerate in the optimisation. The algorithm is also tested using two real applications in computer vision: Multi-view object reconstruction and Lidar scan alignment.

One of the limitations of the algorithm is that the global solution may not necessarily represent the right solution. This may occur when there is not enough distinctiveness in the data sets (as a shape descriptor). This problem can be reduced when using non-isotropic modelling. However, some information about the structure of the shape from which the data sets were obtained needs to be available.

In the next Chapter the non-affine transformation is addressed. We explore the case where a shape model is available and the parameters that fit the model to a set of observations are estimated.

Chapter 6

Morphable Shape Fitting with Bayesian-$\mathcal{L}_2$ and Gaussian Prior

We propose in this Chapter to use our methodology to fit a morphable model to a set of observations. A morphable model provides a way to reconstruct any shape of a class of interest as a linear combination of specific components computed using PCA on a training set (cf. Section 2.3.3). The estimation is based on the general cost function:

$$\hat{\Theta} = \arg\min_{\Theta} \left\{ \mathcal{C}(\Theta) = \frac{\mathcal{L}_2(\Theta, h)}{\tau^2} + \sum_{j=1}^{J} \frac{(\theta_j - \mu_j)^2}{\sigma_j^2} \right\}$$

with the prior term (means and variances) given by the PCA model. We solve the optimisation problem using a dedicated Mean Shift algorithm (cf. Appendix B.2). The shape model and the observations are first aligned with each other using the algorithm proposed in the previous chapter (cf. Chapter 5). We analyse in Section 6.1 the effects of the prior model and the covariance matrices chosen in the robustness and efficiency of the algorithm. In addition, we present in Section 6.2 an application of the algorithm for reconstructing 3D faces from data captured using an RGB-D sensor.

6.1 2D morphable hand model fitting

Let us assume we have a set of observations describing the contour of a specific class of shape for which a shape model is known. The goal is then to estimate the parameters of the model that best fit the observations. The full pipeline of the problem is described in Figure 6.1. Once the data is acquired a) it is processed in order to extract the contours b). From here, two estimation processes take place. First, the data is aligned to the shape model c) and then the parameters of the model are estimated. In this chapter we only discuss the fitting process since the alignment has already been discussed in the previous

chapter.

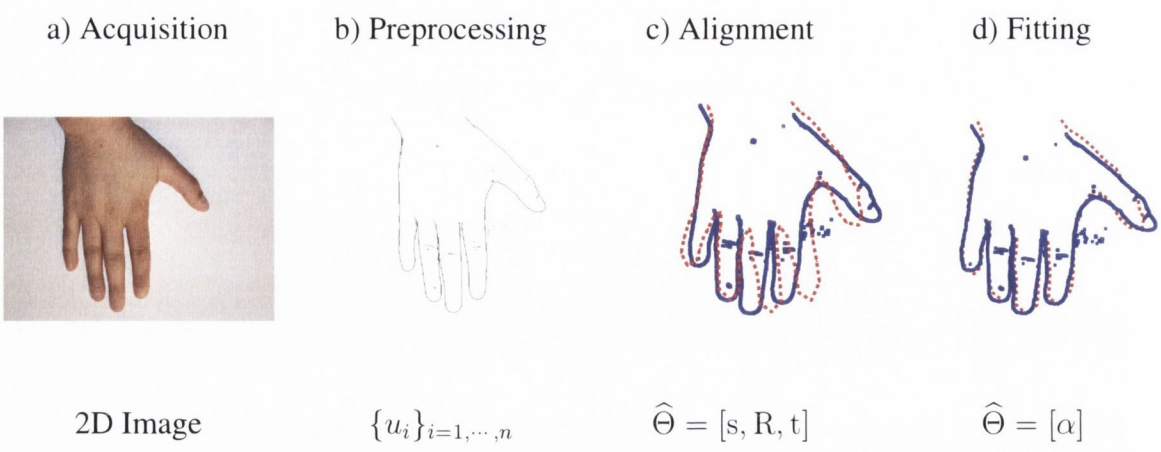


Figure 6.1: Overview of the shape estimation process using a shape model as prior information. The observation as a point cloud $\{u_i\}_{i=1,\cdots,n}$ in b) is obtained after preprocessing the image a) captured by a 2D camera sensor. The parameter estimation is then divided into two steps: c) shape alignment $\Theta = [\mathrm{s}, \mathrm{R}, \mathrm{t}]$ and d) shape fitting $\Theta = [\alpha]$.

We consider the cost function given by our Bayesian-$\mathcal{L}_2$ framework (cf. Equation 4.21). The latent variable to estimate corresponds to the parameters defining the shape model $\Theta = [\alpha_1, \alpha_2, \alpha_3, ...\alpha_q]$ (cf. Section 4.1.2). In this case, we have the following additional information:

1. The prior for the latent variable is assumed Gaussian centred on zero and with variances given by the eigenvalues computed for the PCA model (cf. Section 2.3.3).

2. The structure of the points on the shape model is known so a non-isotropic modelling for the density function $g(\mathbf{x}|\Theta)$ can be implemented (cf. Equation 4.19).

3. There is no knowledge about the structure of the observation. Therefore, $f(\mathbf{x})$ is modelled using isotropic covariance matrices for the Gaussian Mixture. (cf. Equation 4.18)

We propose to solve the optimisation problem using a dedicated Mean Shift algorithm (cf. Appendix B.2). We use a 2D hand model for all the experiments in this section. This model has been trained using the annotated data sets of 18 hands provided by Tim Cootes[1]. Here we report a set of experiments performed to evaluate the proposed algorithm. We discuss the role of the prior information in the convergence of the algorithm and the impact of the non-isotropic modelling for the Gaussian Mixture.

[1] `http://personalpages.manchester.ac.uk/staff/timothy.f.cootes/data/hand_data.html`

6.1.1 Role of the prior

We generated 100 random shapes from the hand model using $J = 10$ eigenvectors. Starting from an initial guess $\alpha^{(o)}$ our algorithm converged systematically towards the correct shapes (cf. Figure 6.2a). The same results are obtained when adding outliers normally distributed on the shape space (cf. Figure 6.2b). The use of the prior in our modelling provides the information needed for preserving the shape of the object during the optimisation. When it is not used (the cost function becomes the $\mathcal{L}_2$ distance) the structure of the shape is lost and the algorithm does not converge to the correct solution (Figure 6.2c and d).

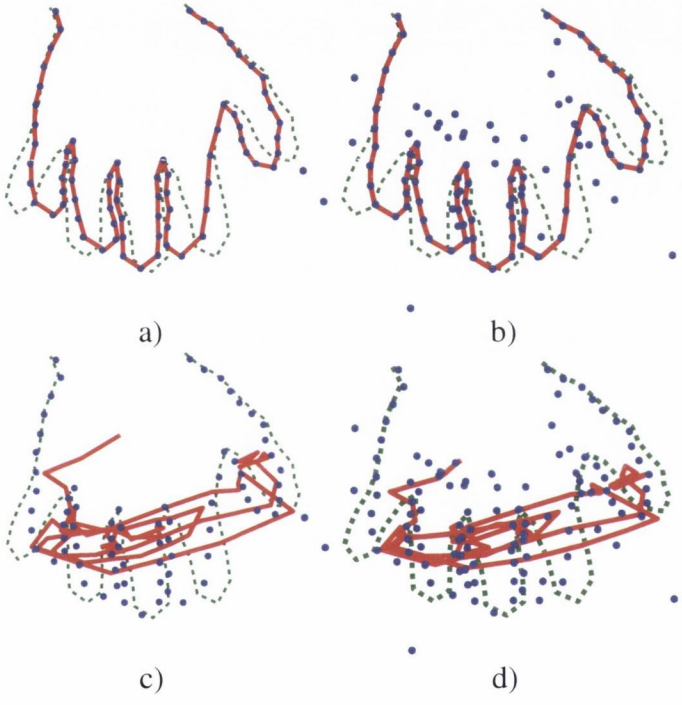


Figure 6.2: Results obtained when using synthetic hands generated from the model a) and when adding random noise b). Figures c) and d) show the estimated shape when the prior information is not considered in the algorithm. In all figures: observations (blue dots), initial guess (green dash) and estimated solution (red line). Setting: $h_{max} = 50$, $h_{min} = 5$.

Similar results are obtained when fitting the 2D hand model to point clouds obtained from images (see Figure 6.3). The number of points on each observed hand are 2421 (top hand) and 2329 (bottom hand). The 2D hand model only contains 72 vertices. Since there is no need for one-to-one correspondence between the data sets, all the observations are considered during optimisation. Figure 6.3 (column c) shows our estimated hands (red dash) and the observations (blue dots). An additional experiment was performed using the algorithm without prior (cf. Figure 6.3 column d). In this case the algorithm does not converge to the correct solution. It converges to a solution that misrepresents the shape

contained in the data set.

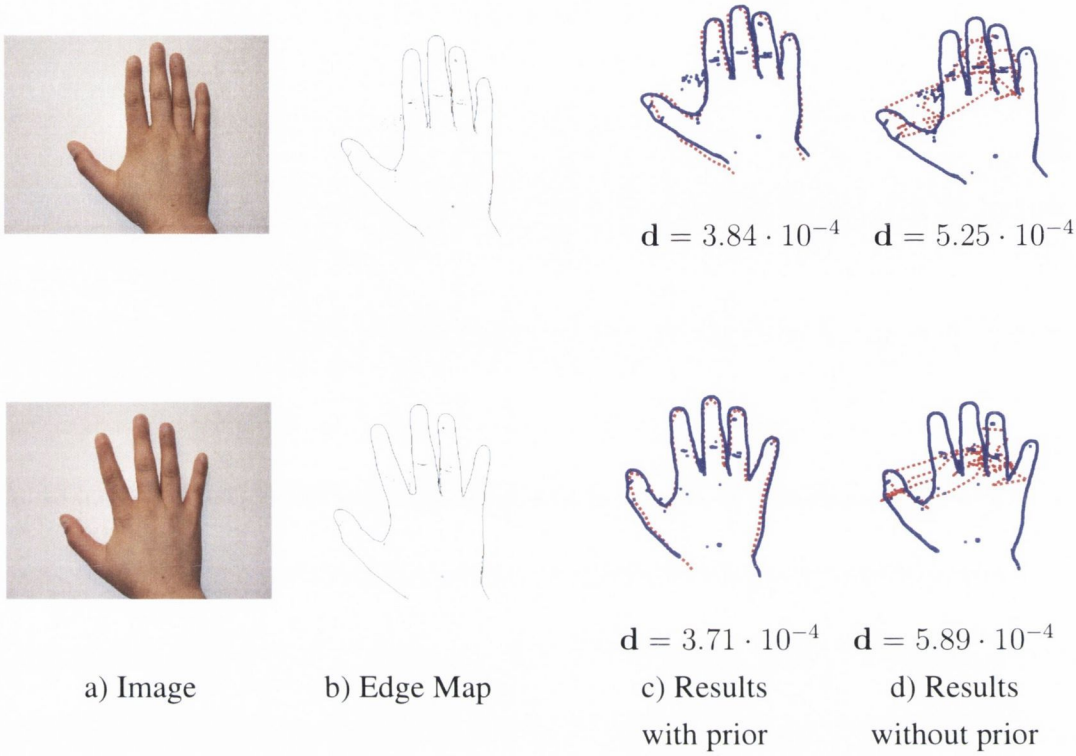


Figure 6.3: Hand model fitting to a point cloud obtained from 2D images. Column a) and b) show the colour image and edge map used for the experiments. In column c) the estimated hands (red dash) and in d) the solution when the prior is not used in the algorithm. For the two experiments we compute the Euclidean distance $\mathbf{d}$ between the model and the observations. When using the prior information the algorithm minimises the Euclidean distance better (results in c) than when the prior is not used d).

6.1.2 Using isotropic and non-isotropic covariance matrices

We show in this section the advantage of using non-isotropic covariance matrices when modelling the GMM for the shape model. We use as observations the image displayed in Figure 6.3 (top left). The data set contains $n_f = 2421$ points. We take four groups with 100 data sets each sampled randomly from the observations. The number of samples to consider in each set of the four groups are $n_f = 50, 100, 300$ and 1000 respectively. We run the algorithm for the 100 data sets in each group using non-isotropic modelling for the GMM corresponding to the shape model (cf. Section 4.19). The setting used for the bandwidths in all experiments are: $h_{max} = 20$ and $h_{min} = 5$. We use for this experiment all the vertices in the hand model ($n_g = 71$). We initialise τ using Θ_o and $h = h_{max}$ (cf. Equation 4.22). Once the algorithm converges for $\hat{\Theta}$ we update τ according to the

n^o Points (n_f)	Non-Isotropic	Isotropic
50	83%	0%
100	92%	0%
300	100%	78%
1000	100%	97%

Table 6.1: Comparison of the convergence rate between isotropic and non-isotropic modelling

new values for h and $\Theta_o = \hat{\Theta}$ (cf. Appendix B.2). An example of the data set used in each group is shown in Figure 6.4. Note that the number of points in each set are not necessarily the same. Moreover, there is no need for correspondences in between points in both data sets (shape model and observation). The estimation results obtained (in percentage) for each group are reported in Table 6.1. For comparison, the same experiments are run using isotropic modelling for the GMM corresponding to the shape model (cf. Equation 4.18). Results show that the convergence towards the ground truth is better achieved when the GMM is defined using non-isotropic covariance matrices. Furthermore, the non-isotropic modelling performs better when the data considered is a small subset sampled from the original observation.

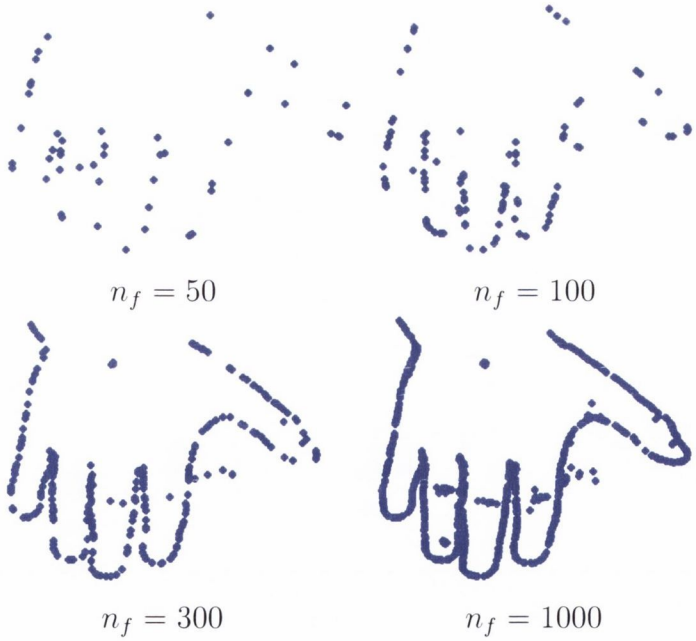
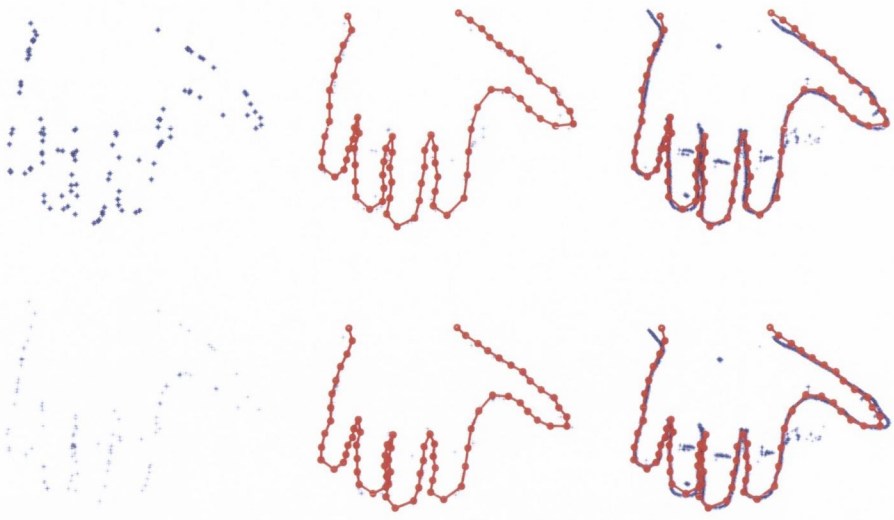


Figure 6.4: Examples of observations when randomly sub-sampling the original data set.

Figure 6.5 shows two examples of the result of the estimation achieved when the observations consist of 100 points randomly sampled from the original data set. Additional results are reported in Appendix C.3.

Non-isotropic modelling not only helps in improving convergence of the algorithm



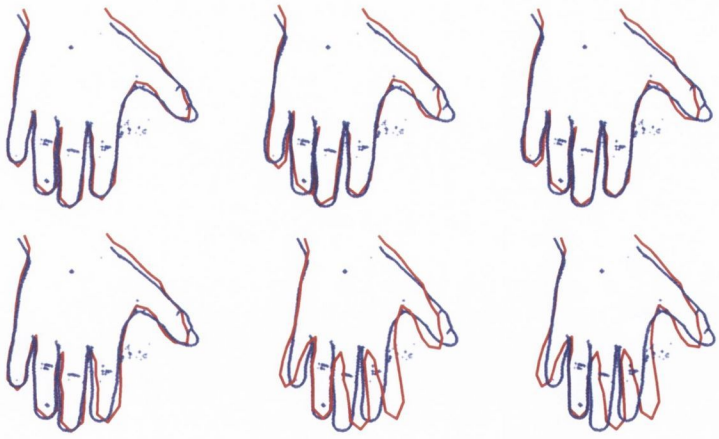
a) b) c)

Figure 6.5: Examples of results obtained when using non-isotropic covariance matrices for the GMM that represent the hand model. Column a) shows the data used as observations (100 samples from the original data set). Column b) and c) show the results obtained (red line) superimposed with the observation used b). Results using the full data set are displayed in c).

when the observation is a small data set of points but also allows the reduction of the number of vertices used in the GMM. Figure 6.6 shows results when reducing the number of vertices to use in the GMM of the shape model (n_g). The estimation when using non-isotropic covariance matrices with $n_g = 71, 54$ and 43 converge towards the correct solution in all cases (cf. Figure 6.6 top row). However, when using isotropic covariance matrices, the estimation degenerates when reducing n_g (cf. Figure 6.6 bottom row).

6.2 3D morphable face model fitting

In this section we show a fully automatic method for fitting a morphable model of the face to data captured using the Kinect sensor. The algorithm is based on the affine transformation proposed in Section 5.1 and shape fitting proposed in Section 6.1 (cf. Section 4.1.3). Our algorithm is assessed with the 3D morphable shape face model provided by Basel University [82] (truncated to keep only the face region). This model has been computed using PCA on 3D meshes of neutral faces (100 males, 100 females) captured with a high-end 3D scanner, and it provides users with a mean mesh face and all eigenmeshes with their eigenvalues. The coordinates α on these eigenmeshes are estimated with our algorithm (as well as the rotation and translation for alignment $\Theta = [\phi_{xy}, \phi_z, t_x, t_y, t_z, \alpha_1, \alpha_2, \alpha_3, ...\alpha_q]$) and it is assumed here that any neutral face can be well reconstructed with this model. Our approach is tested on synthetic data generated from the model (Section 6.2.1) and experiments using real data captured using the Kinect



$n_g = 71$ $n_g = 54$ $n_g = 43$

Figure 6.6: Examples of results obtained when reducing the number of Gaussians for the shape model (n_g). Top row shows results of the estimation when using non-isotropic covariance matrices. The bottom row shows results when using isotropic covariance matrices. In all examples, the red line corresponds to the estimate while the blue dots to the observations.

sensor are reported in Section 6.2.2 (note that in this case, the target faces are different from the faces used to compute the PCA model).

6.2.1 Fitting 3D face model to synthetic data

A set of synthetic target faces are created from the model by randomly generating a set of parameters αs using $J = 10$ eigenvectors (these constitute the ground truth parameters α_{GT}). The rotation and translation between the model and the target is known and only the parameters α are estimated. In this experiment, the covariance matrices are chosen to be isotropic because the two point clouds have been generated from the same synthetic 3D model in this experiment (cf. Section 4.18). This is not the case when we use the Kinect camera to capture the observations for which we will use adaptive variable covariance matrices (cf. Section 6.2.2) as explained in Section 4.19.

Figure 6.7 shows several results of 3D face reconstruction. We can visually recognise that the estimated face using the fitting algorithm is similar to the target face in all four examples.

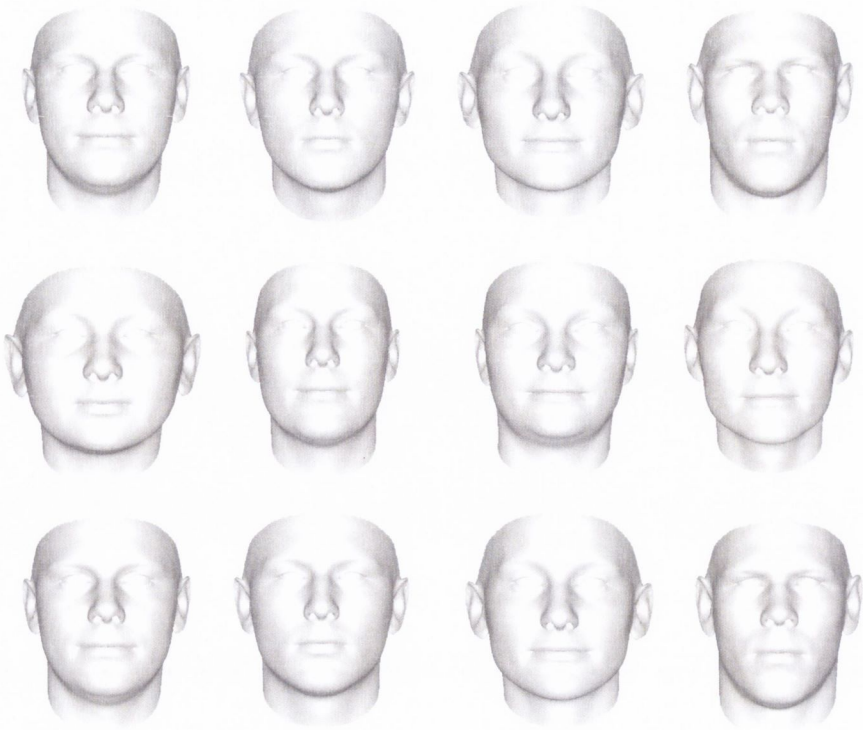


Figure 6.7: Fitting synthetic faces: target face (observations, top row), random starting guess for initialising the algorithm (middle row) and estimated face with our algorithm (bottom row).

Figure 6.8 shows the error surface between the target and the estimated reconstruction. Note that we use a truncated model: neck and ears areas have not been matched. The errors on the face (in light blue) correspond to areas that are not well captured in the $J = 10$ eigenvectors that we have used for reconstruction in this experiment.

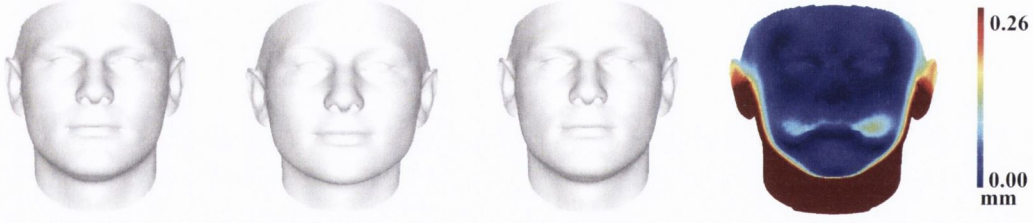


Figure 6.8: Surface error computed between the target face and our reconstruction.

Convergence of the algorithm to the optimal solution is tested by running experiments with different starting points (randomly chosen) and the error is computed between the estimated α and the ground truth. In all the experiments the estimates converge to the ground truth with a root mean square error smaller than 0.0054.

6.2.2 Fitting 3D face model to Kinect data

We have just shown that the algorithm converges well in controlled conditions. In real applications however, the observations originate from a different process than the model and more care needs to be given to modelling the covariance matrices. When the data are not uniformly sampled (or are sparse), the covariance matrices cannot be chosen to be isotropic but need to be chosen such that the pdfs well approximate the surface shapes. Covariance matrices are next set as explained in Equation 4.19. The number of Gaussians in the mixture are reduced as described in Section 4.2.4.

Data capture and preprocessing

The Microsoft Kinect sensor provides a depth map and a colour image of the scene with a resolution 480×640 pixels. The field of view captured is within a range of approximately $50cm$ and $4m$. To obtain the point cloud of the face, we first select the region of the image within a range of $50cm$ to $120cm$ from the sensor (cf. Figure 6.9b). The face region is then detected and converted to a point cloud by using a face and skin detector (cf. Figure 6.9c).

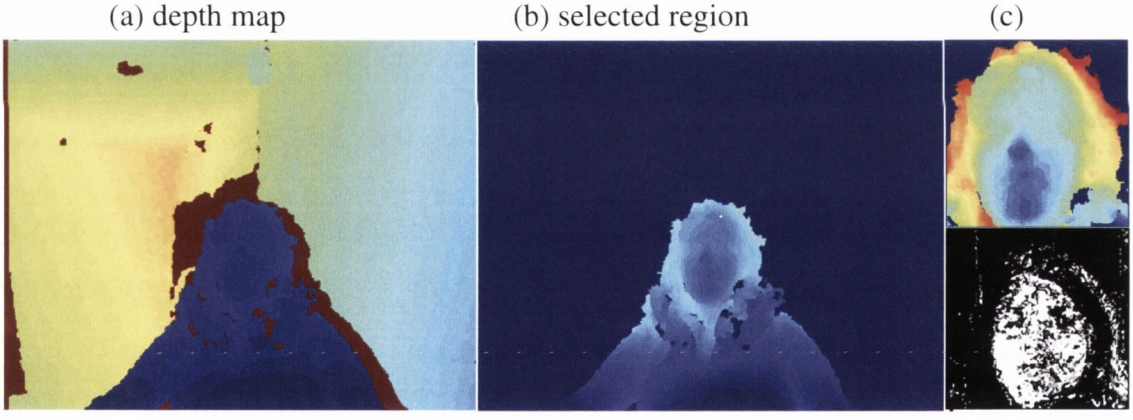


Figure 6.9: Preprocessing for generating the points cloud of the face (target): depth map (a) as captured by the RGB-D sensor, selection of the scene (b) in close range (between $0.5m$ and $1.2m$ from the sensor), extracted face and skin region (c).

Kinect point cloud alignment

A crude estimate of the translation between the target and the model point clouds is computed by matching their barycenters in the 3D space. This estimation is used as a starting guess for our algorithm for rigid transformation.

The settings used for h in the optimisation are $h_{max} = 1cm$ and $h_{min} = 5mm$. Both datasets are downsampled with a ratio of $1 : 10$ in order to reduce the number of Gaussians in the density functions and to speed up the optimisation process.

Figure 6.10 shows an example of the alignment process: before alignment (bottom row) and after alignment (top row). We evaluate numerically the performance of the

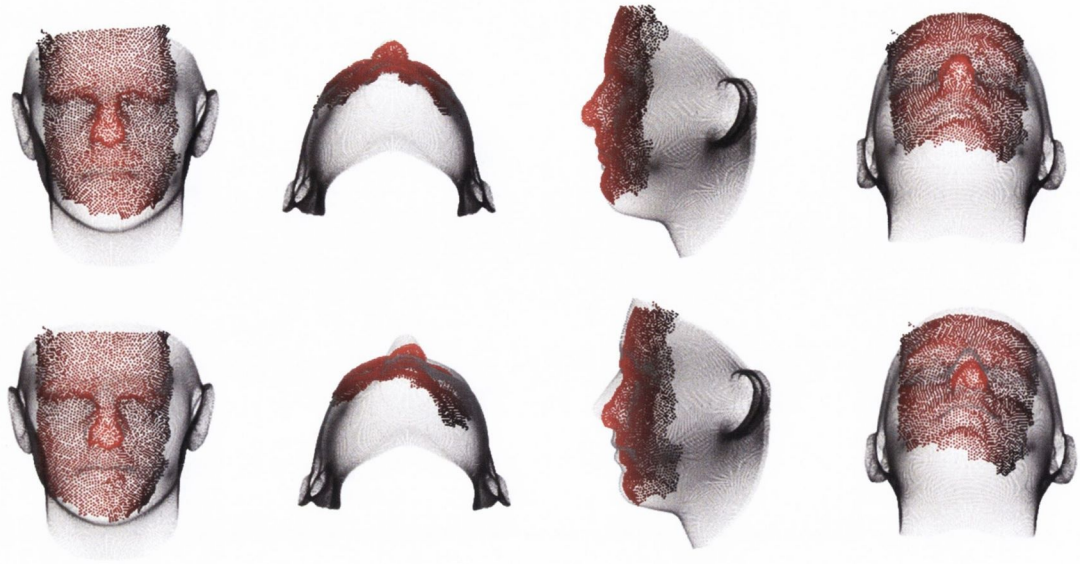


Figure 6.10: Shape alignment: different views of the point clouds (model (grey) and observations (red)) after alignment (top row). Same point clouds displayed before alignment (bottom row).

alignment by comparing the error between the observed point cloud and the average shape before and after the alignment. The error is computed as the Euclidean distance of each point in the observed data set to its closest triangle in the mesh of the average shape. Figure 6.11 shows the histogram of the error of the aligned shape (red line) and before alignment (blue dash). Note the number of pixels closer to the shape model (error close to 0) increases after alignment.

3D morphable shape fitting on Kinect point clouds

Once the target face is aligned to the shape model, the αs are estimated using average shape (cf. Figure 6.12) as a starting guess in our algorithm (e.g. $\alpha_j = 0, \forall j \in \{1, \cdots, J\}$) and with the following settings: $h_{max} = 1.5cm$, $h_{min} = 5mm$ and $J = 20$. Figure 6.13 shows the reconstructed faces for several people (none were used to train the morphable model). The last two faces (F5 and F6) correspond to the same person appearing behind occluding objects. The general shape of the faces is well recovered while some detailed areas are sometimes not accurate (e.g. eyes or mouths). Some areas may not be well described by the first $J = 20$ eigenvectors for these people in this experiment.

Figure 6.14 compares the reconstruction of F5 with a capture using a more accurate laser scanner (Minolta Vivid 700). As can be seen, the laser scan is not perfect either and

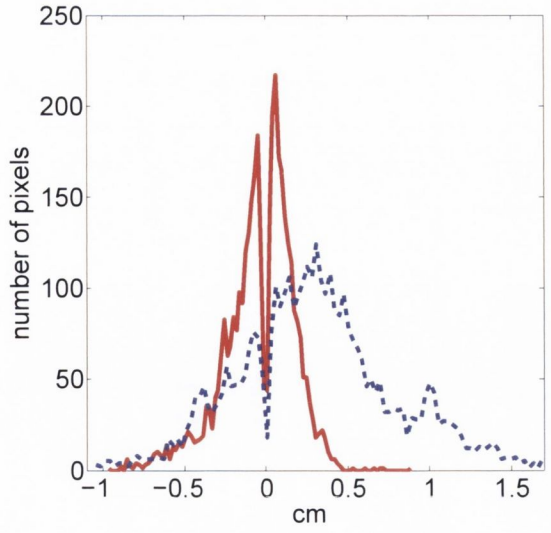


Figure 6.11: Histogram of the errors between the observations to the average shape: before alignment (blue) and after (red). The sum of the absolute error is 6.3456×10^6 (after alignment) compared with 1.7054×10^7 (before alignment).

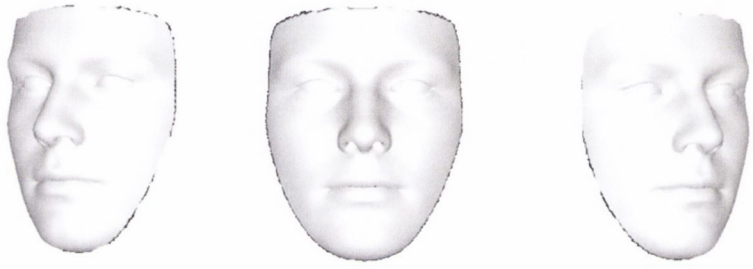


Figure 6.12: Average shape used as initial guess.

the reconstruction with the model has the advantage of recovering a full mesh without any holes.

The Euclidean distance (error) between each point in the target to its closest triangle in the reconstructed shape is computed and the histogram of these errors is shown in Figure 6.15. In all the experiments, the error is significantly reduced after the fitting: the variance of the distribution of the error is smaller for the data computed using the reconstructed face. In both cases, the 90% of observations are within a distance of $3mm$ after the fitting is done (compared with 50% before the fitting is done, see Figure 6.16).

Figure 6.17 shows the estimated $J = 20$ coordinates normalised with the eigenvalues for faces F1 to F6 (cf. Figure 6.13). We can note that the estimates for F5 and F6 are close to each other, corroborating the fact that the same person appears in both captures. Despite the occlusions, the algorithm converges towards the same solution for F5 and F6.

We have computed the Mahalanobis distance (with Λ diagonal matrix of the eigenvalues):

$$d_{i,j} = \sqrt{(\hat{\alpha}_{Fi} - \hat{\alpha}_{Fj})^T \Lambda (\hat{\alpha}_{Fi} - \hat{\alpha}_{Fj})}$$

Figure 6.13: Estimated reconstructed faces (labelled F1 (top), to F6 (bottom)) from 3 viewpoints shown with the colour image captured with the Kinect (left).

for all faces F1 to F6. The results are shown in Table 6.2. The distance $d_{i,j}$ is smaller when the parameters correspond to the same individual (e.g. F5 and F6, $d_{5,6} = 0.0888$) than when considering different people (e.g. F1 and F2, $d_{1,2} = 0.3596$). Although these experiments are preliminary and require further analysis, they suggest the feasibility of robust identification using noisy depth sensors.

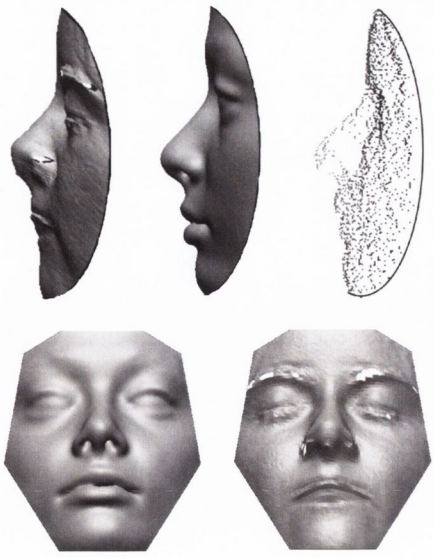


Figure 6.14: At the top, profile view of F5: laser scan (left), reconstruction (middle) and Kinect point cloud (right). At the bottom, frontal view of F5: reconstruction (left) and laser scan (right).

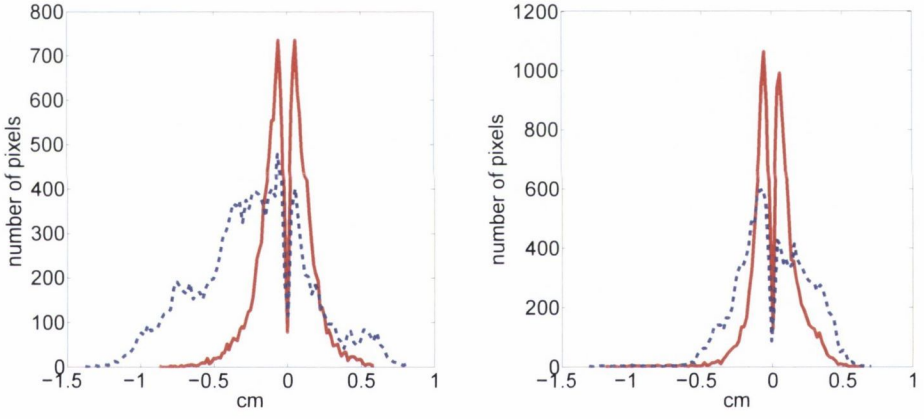


Figure 6.15: Histograms of the errors between the observations and the reconstructed face (red line) and the observations and the average shape of the model (blue dash) for F1 (left) and F2 (right).

Table 6.2: Malahanobis distance $d_{i,j}$ between the estimated parameters of faces F1 to F6 (Figure 6.13).

Faces	F1	F2	F3	F4	F5	F6
F1	0	0.3596	0.5374	0.6925	0.7609	0.7815
F2		0	0.3396	0.5041	0.6141	0.6286
F3			0	0.3885	0.4978	0.5155
F4				0	0.5508	0.5475
F5					0	0.0888
F6						0

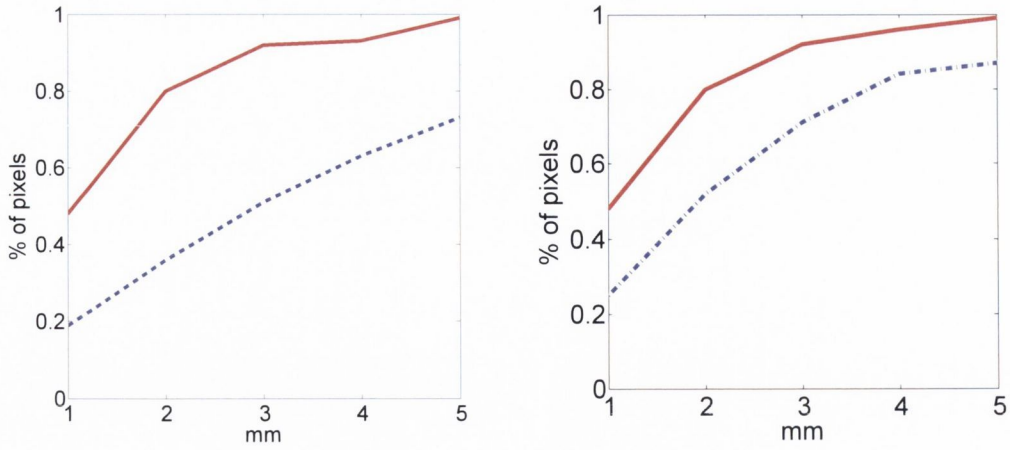


Figure 6.16: Percentage of observations below a distance between 1 and 5mm (reported on the absissa) for F1 (left) and F2 (right), before fitting (blue dash) and after fitting (red line).

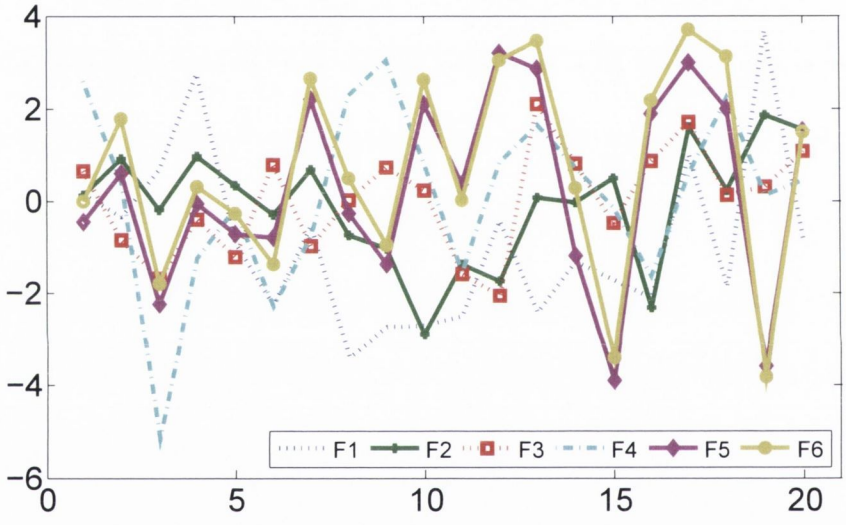


Figure 6.17: Values of the $J = 20$ coordinates of parameter α normalised with the eigenvalues computed for faces F1 to F6.

6.3 Conclusion

We have shown in this chapter the use of the Bayesian-$\mathcal{L}_2$ framework to estimate the parameters of statistical shape models when fitting to a set of observations. This method allows us to include prior information over the parameters to estimate. Unlike other shape fitting methods, ours does not need any correspondence between data sets to estimate parameters. The only parameters to set are the bandwidth associated with the modelling of the shapes as GMMs and the bandwidth associated with the $\mathcal{L}_2$ distance. The latter controls the weight of the prior in the cost function and is set proportional to the initial value of $\mathcal{L}_2$. We solve the optimisation problem using a dedicated Mean Shift algorithm. We tested the role of the covariance matrices when modelling the GMMs. Results show that the non-isotropic covariance matrices can be used for modelling GMMs that best

suit the shape. Therefore, the convergence of the algorithm towards the correct solution is better achieved under this modelling. The algorithm is validated for 2D and 3D model fitting.

The main disadvantage of the fitting algorithm is that the shape model will only deform according to the data that was used for its training. This is a common problem for all learning based methods. In the next Chapter we explore our framework for estimating parameters of a parametric curve such as the ellipse. We will show that GMMs can be used for encoding not only the contour of a shape but also any other relevant information.

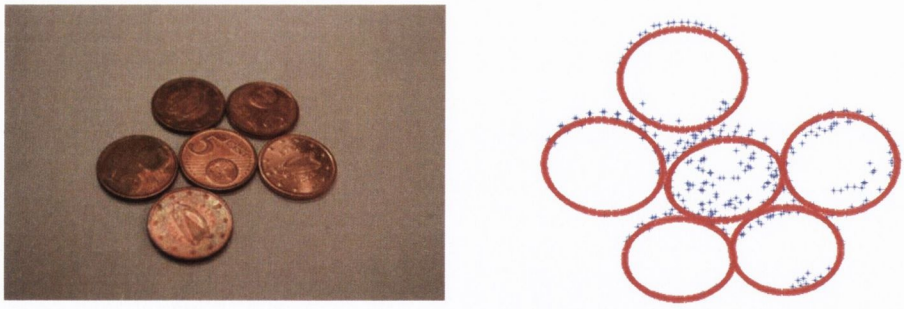
Chapter 7

Extensions of Bayesian-$\mathcal{L}_2$ framework

In this chapter we explore two extensions of our Bayesian-$\mathcal{L}_2$ framework. First, we propose a method for estimating multiple instances of a shape of interest in a set of observations (Section 7.1). Secondly, we explore the use of the multidimensional modelling for the density functions and apply it to the estimation of parametric curves (cf. Section 7.2). We discuss the particular case where the curve corresponds to an ellipse. We test our proposed algorithm when detecting multiple occluded ellipses in a benchmark data set and compare the results with the state of the art algorithms.

7.1 Detecting multiple instances

In the previous two chapters we have explored the Bayesian-$\mathcal{L}_2$ framework in applications where the shape of interest only occurs once in the observations. In this section we explore the case where more than one instance of the shape of interest are considered (i.e multiples coins in Figure 7.1).



a) b)

Figure 7.1: Example of detection of multiples instance (i.e coins).

The cost function of our Bayesian-$\mathcal{L}_2$ proposed in 4.21 allows us to estimate the shape that best fits the observation. Furthermore, the annealing strategy implemented makes the algorithm converge towards the global solution (cf. Algorithm 1, Section 4.2.3). In this

section, the goal is not only to estimate the global solution. We aim to estimate a set of shapes $S = \{\hat{\Theta}_1, ..., \hat{\Theta}_s\}$ that represent all the instances (s) of the shape of interest in the observation.

Let us consider a set of observations $\{u_i\}_{i=1,\cdots,n} \in \mathbb{R}^D$. We propose an iterative algorithm based on the following steps:

1. **Shape parameters estimation:** This step estimates the parameters of the shape $\hat{\Theta}$ given a set of observations by minimising the cost function proposed in Equation 4.21. The general algorithm that solves this problem is described in Section 4.2.3 (cf. Algorithm 1).

2. **Observations update:** Once an instance of the shape is detected $\hat{\Theta}$ all the observations associated with that shape are removed. The remaining observations are then used to detect the following instances of the shape. The updated observation can be expressed as follows:

$$\mu_i \in \{\mu^{updated}\} \text{ If } \frac{p(\mu_i|\hat{\Theta})}{\sum_{i=1}^{n} p(\mu_i|\hat{\Theta})} < t_1$$

   Where, t_1 is a threshold defined by the user and $p(\mu_i|\hat{\Theta})$ is the probability of the observation μ_i being part of the detected shape $\hat{\Theta}$. All observations for which normalised probability of being part of the selected shape is above the threshold t_1 are discarded. The remaining observations are considered for estimating the next instance of the shape of interest.

Those two steps are iterated until all instances of the shape are found. We consider the algorithm has found all the shapes if the observations remaining are less than a given threshold t_2 (cf. Algorithm 4). The values for the threshold t_1 and t_2 are usually set as 0.3 and 0.1(The algorithm will stop when the remaining observations are less than 10% of the original data). An illustration of the iterations of the algorithm are shown in Figures 7.2 and 7.3. Figure 7.2 shows the observations (blue) and detected coin (red) in each step. Note that the observations associated with the detected shape do not appear in the next iteration. Figure 7.3 shows the probability (normalised) of each observation being part of the detected coin.

Algorithm 4 Estimating parameters of multiples instances of shape

Input: $t_1, t_2, \{u_i\}_{i=1,\cdots,n}, \beta$ and $\Theta^{(0)}$

Init $h = h_{max}$ and $\hat{\Theta} = \Theta^{(0)}$

$\{u_i^s\}_{i=1,\cdots,n^s} = \{u_i\}_{i=1,\cdots,n}$

repeat

Init $h = h_{max}$ and $\hat{\Theta} = \Theta^{(0)}$

$\hat{\Theta} \leftarrow$ Estimate $\left(\mathcal{C}(\Theta), \hat{\Theta}, h, \tau, \beta\right)$ (cf. Algorithm 1, Section 4.2.3)

$\hat{\Theta}_s = \hat{\Theta}$

s^{++}

$\{u_i^s\}_{i=1,\cdots,n_s} \leftarrow$ Update $\left(\{u_i^{s-1}\}_{i=1,\cdots,n_{s-1}}, \hat{\Theta}_s, t_1\right)$

until $\frac{n-n^s}{n} < t_2$

Output: $\hat{S} = \{\hat{\Theta}_1, ...\hat{\Theta}_s\}$

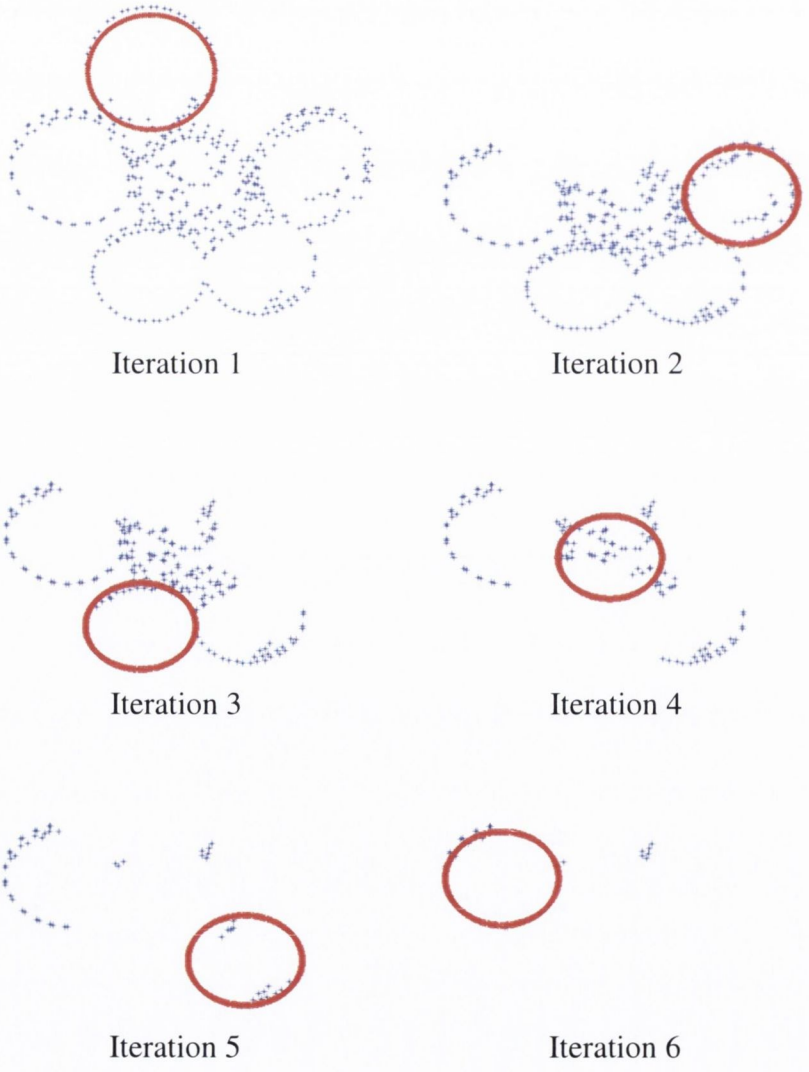


Figure 7.2: Illustration of the iteration of the proposed method for detecting multiple instances of a shape.

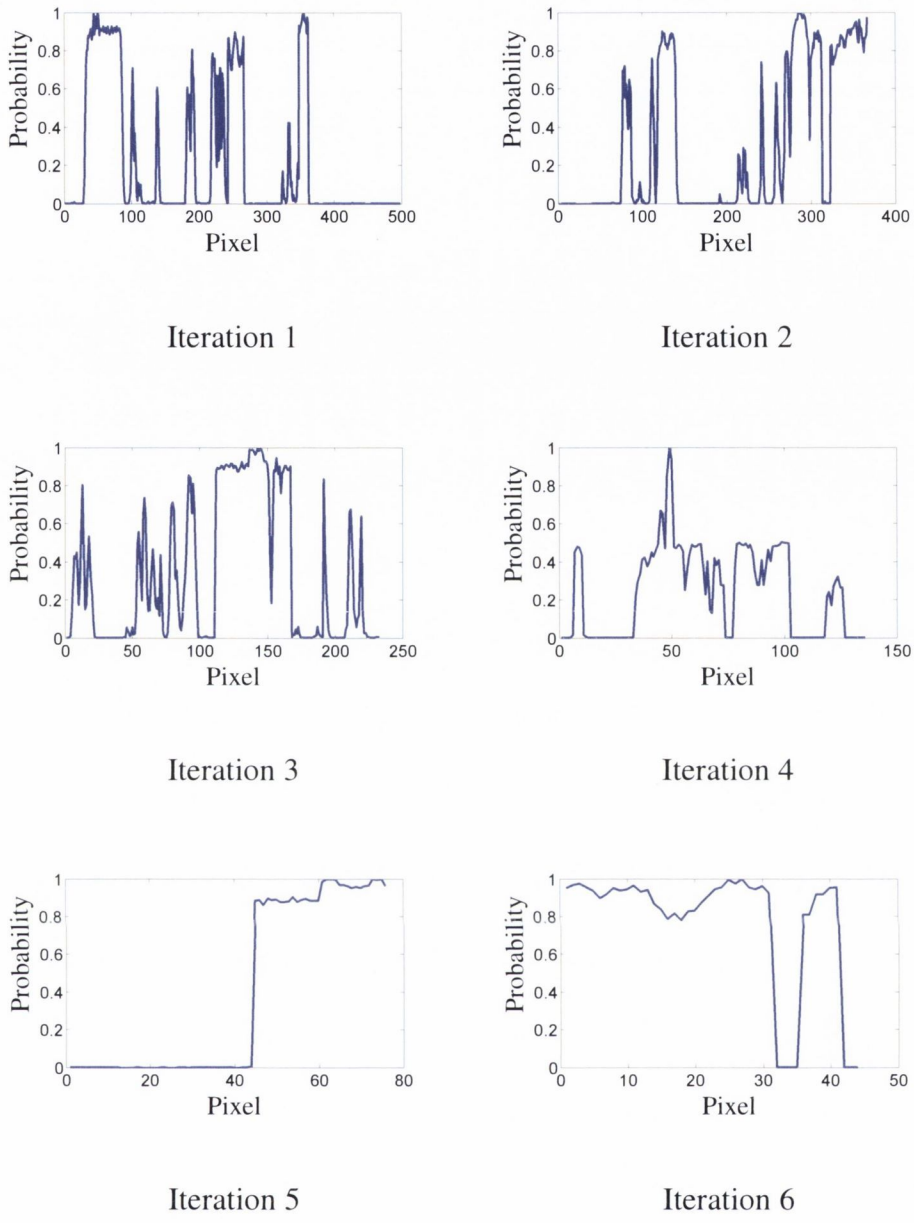


Figure 7.3: Normalised probability of each point of being part of the detected shape.

7.2 Application to ellipse detection

In this section we tackle the problem of estimating the parameters of a known curve given a set of observations. We explore the particular case when the curve corresponds to ellipses and circles. The parameters to estimate are the centre of the ellipse (x_o, y_o), the semi-minor and semi-major lengths a and b respectively and γ the angle of rotation of the ellipse with respect to the horizontal axis ($\Theta = [\gamma, a, b, x_o, y_o]$). In section 7.2.1 we explore the case where the ellipse is fitted to a point cloud. In section 7.2.2 we explore the problem of detecting multiples ellipses in images. The GMM is modelled using the edge map of the images (position) and its curvature as described in section 4.3. We applied

in this section the multiple instance detection algorithm proposed in Section 7.1. Results in both sections are compared with standard algorithms demonstrating the feasibility and competitiveness of our algorithm for ellipse detection.

7.2.1 Fitting one ellipse to noisy observations

In this section we solve the cost function proposed in 4.21 for the ellipse fitting problem. The GMM representing the observation f is modelled using isotropic covariance matrices as described in section 4.18. In the case of the reference set given by the parametric equation of the ellipse a non-isotropic model is implemented as described in section 4.19. In this case we use 20 samples of the ellipse to define the GMM for g (cf. Figure 7.4). We do consider all solutions equally probable. Hence, a prior Gaussian distribution is modelled for the latent variable using a large bandwidth. A gradient ascendant algorithm is implemented to solve the optimisation problem (cf. Appendix B.3). As follows we report two experiments designed for testing the sensitivity of the algorithm to noise and outliers.

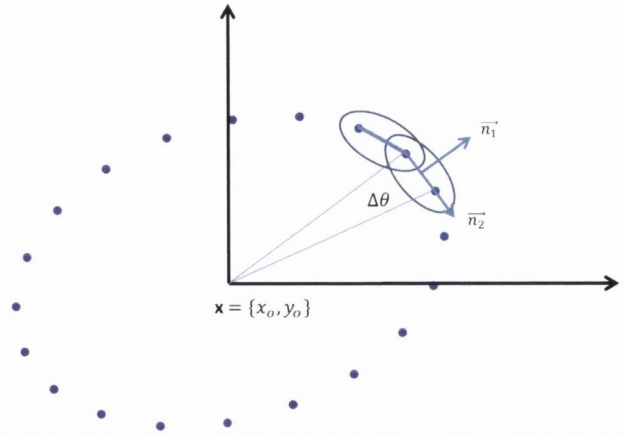
Sensitivity to noise

We define for this experiment a target ellipse (ground truth) centred at the origin and without rotation. The semi-major length equal to 5 and semi-minor length equal to 3. The ellipse is corrupted using five level of Gaussian noise with standard deviation from 0.1 to 0.5. For each level of noise we take 50 points from the ellipse. For each level of noise we generate 50 data sets to perform the experiments. Figure 7.5 shows three data sets (used as observation) using level of noise 0.1, 0.3 and 0.5 respectively.

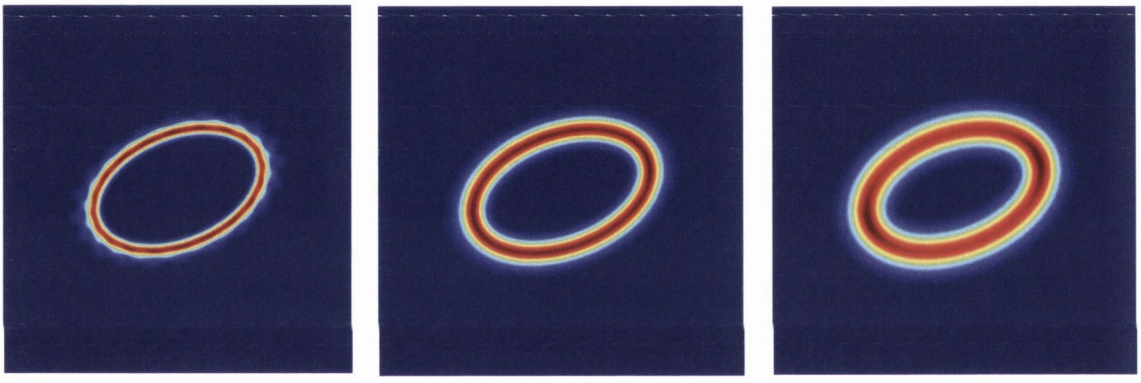
We run our algorithm using the following settings: $h_{max} = 0.9$, $h_{min} = 0.2$ and the geometric decreasing rate for the bandwidth $\beta = 0.8$. The initialisation for all the experiments is $\Theta^o = [0, 1, 1, 0, 0]$, which is basically a circle centred on the origin with radio equal to 1. We compute the error between the estimated ellipse (given by $\hat{\Theta}$) and the target (observation) using the normalized area of the symmetric difference [7]. This metric for error between the true ellipse E_t and the fitted ellipse E_f is defined as follows:

$$Error = \frac{S_{E_t \cup E_f} - S_{E_t \cap E_f}}{2S_{E_t}} \tag{7.1}$$

Where $S_{E_t \cup E_f} - S_{E_t \cap E_f}$ is the area of the symmetric difference and S_{E_t} denotes the area of the true ellipse (cf. Figure 7.6). In order to compute this error rate, we use the function phantom in Matlab to create an image containing an ellipse. The image is represented by assigning a value equal 1 to all pixels inside the ellipse and zero otherwise. One image I_t is created using the parameters of the target ellipse (observation). The second image I_f is created using an estimated ellipse. The symmetric difference between those two ellipses



a) Ellipse representing the shape model using 20 samples from the curve.



b) Bandwidth $h = 0.2$ c) Bandwidth $h = 0.4$ d) Bandwidth $h = 0.6$

Figure 7.4: Example of the GMM of the ellipse computed using different bandwidths.

a) noise $\sigma^2 = 0.1$ b) noise $\sigma^2 = 0.3$ c) noise $\sigma^2 = 0.5$

Figure 7.5: Figure shows three examples of observation used in the experiments. The ellipses were computed using the target ellipse for which the parameters are $\Theta = [\gamma, a, b, x_o, y_o] = [0, 3, 5, 0, 0]$ corrupted using Gaussian noise with standard deviation σ^2 equal to 0.1, 0.3 and 0.5 respectively

is computed as a function of the number of pixels with a value equal to 1 after adding the two images. The error rate in equation 7.1 is then redefined as follows:

$$Error\ rate = \frac{pixel(I_t + I_f == 1)}{2\ pixel(I_t == 1)} \tag{7.2}$$

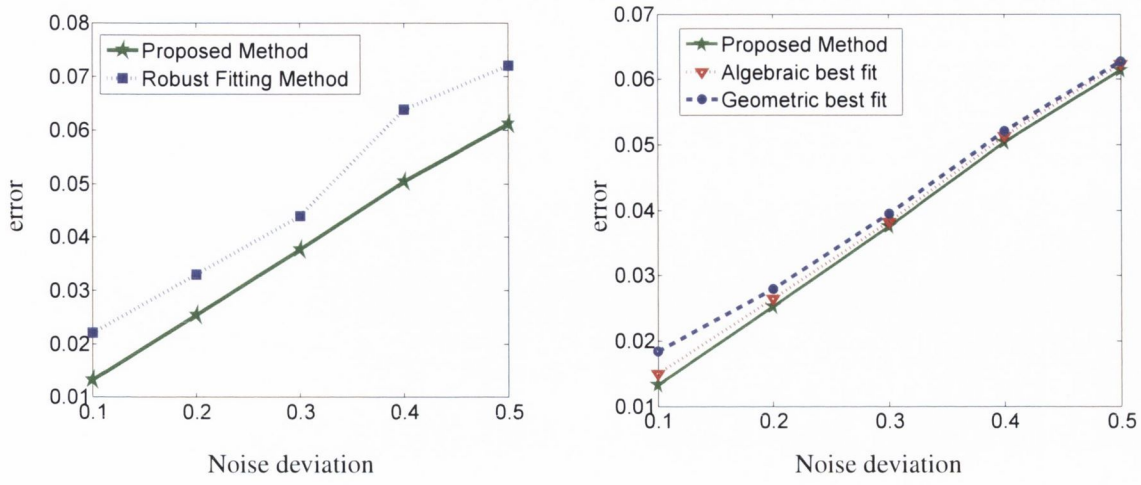
a) b)

Figure 7.6: Example of computing the error rate. Figure a) shows in blue the true ellipse (observation) and in red dots the estimated one. The black region in figure b) represents the area of the symmetric difference in between the two ellipses.

The results obtained for all the data sets are reported in Figure 7.7. In the left plot a) we compare the results obtained using our proposed method (green start) and the method proposed by Yu et al.[7]. On the right hand side of Figure 7.7b) we compare it with the Non Linear Least squares (geometric best fit) and using the direct ellipse fitting algorithm (algebraic best fit). In both cases the error between the estimated ellipse and the true ellipse is better minimised using our proposed method. The standard deviation of the error is also reported in Figure 7.8. Additional results are reported in the appendix C.26.

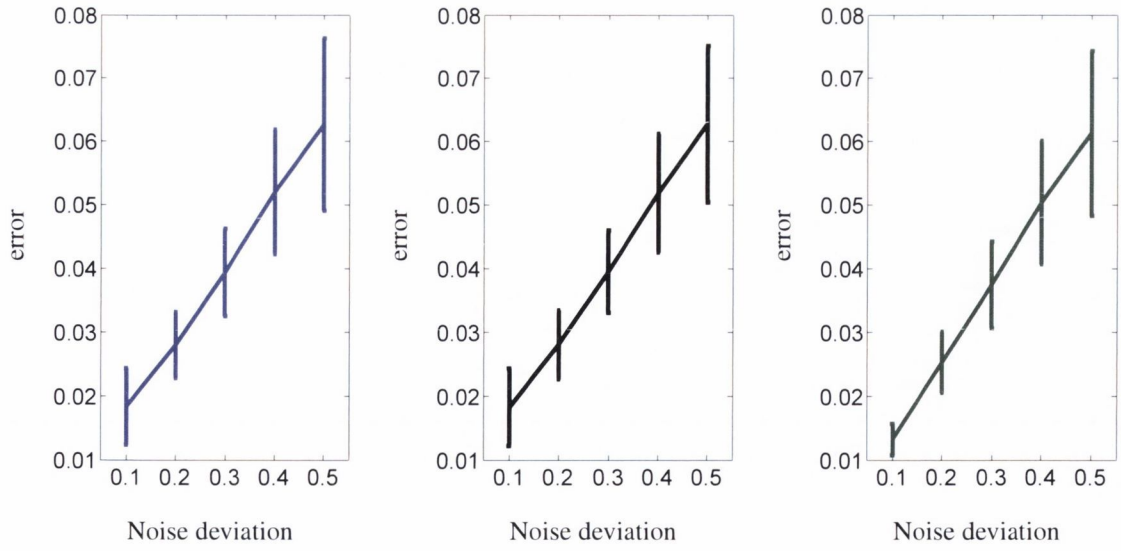
Robustness to Outliers

A second experiment is performed to evaluate the robustness of the algorithm to outliers. We use the same data as in the previous experiment (for all level of noise) but adding 10% of extra points randomly distributed. As it is shown in the Figure 7.9, our proposed method does maintain its performance. On the contrary, the error obtained using the direct fitting and the Least Square methods increase [8, 9]. Those methods are very sensitive to outliers while our proposed method keeps its robustness. Examples of the data used in the trial and the estimated ellipses are shown in Figure 7.10. The results are similar when the outliers correspond to another ellipse presented in the data. Examples of this case are shown in Figure 7.11. The mean error for our proposed method when 50 trials are tested is 0.0133 with a standard deviation of 0.0012. Those values are in correspondence with the values obtained when no outliers at all are presents in the data set. This shows the robustness of the proposed algorithm for fitting ellipses to noise data sets.



a) b)

Figure 7.7: Mean Error obtained for the data sets perturbed with 5 different standard deviation of noise (σ from 0.1 to 0.5). In a) we compare the proposed method (green star) versus a recent robust ellipse fitting algorithm reported by Yu et al. [7] (blue square). In b) our proposed method (green start) is compared with two standard ellipse fitting algorithms. The red triangle represents the algorithm which cost function minimises the algebraic distance while the blue dot is the well known Least Square method that is based on the minimisation of a geometric distance



a) Least Square b) Direct Ellipse Fitting c) Proposed Method

Figure 7.8: Figure shows a detail of the error of each method including its 95% of confidence interval.

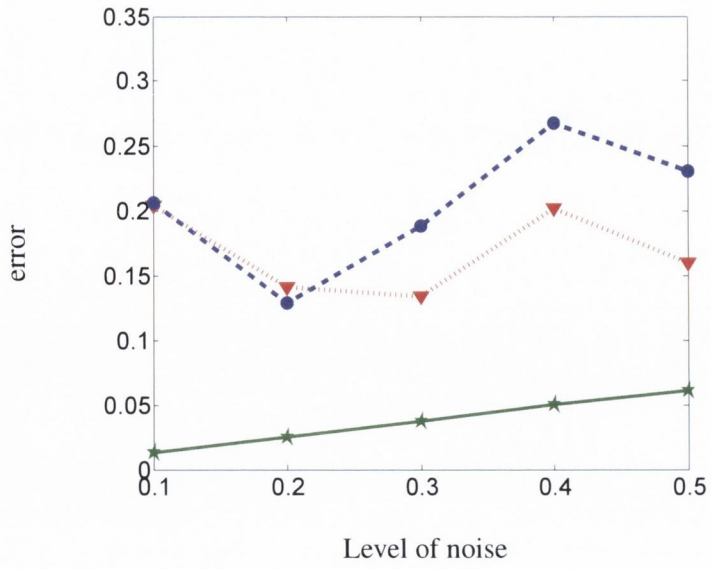
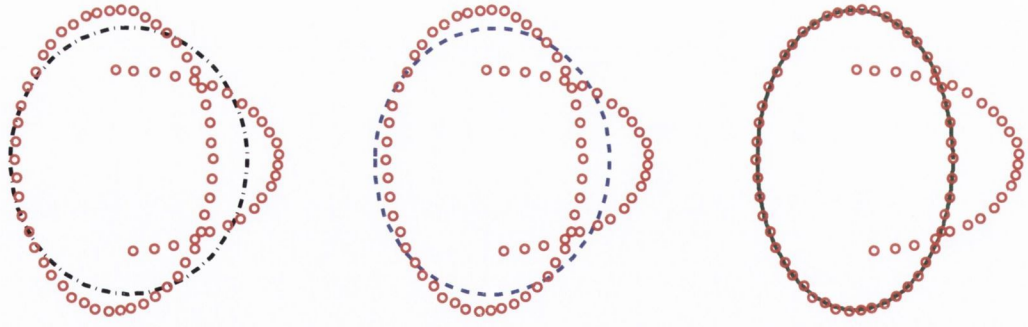


Figure 7.9: Results of the mean error compared with standard fitting ellipse methods for 5 level of noise. This experiment corresponds to the case where 10% of extra points randomly distributed are added to the observation. The error computed when using our proposed method (green start) is compared with two standard ellipse fitting algorithms. The red triangle represents the algorithm which cost function minimises the algebraic distance while the blue dot is the well known Least Square method that is based on the minimisation of a geometric distance [8, 9].



a) b) c)

Figure 7.11: Results compared two ellipses. For our method we run a trial of 50 examples obtaining a mean error of 0.0133 with a standard deviation of 0.0012. When perturbed using $\sigma = 0.1$. In all the figures the red dots correspond to the observation. In a) we show the results obtained using the direct fit algorithm, in b) Least Square method and in c) our proposed method.

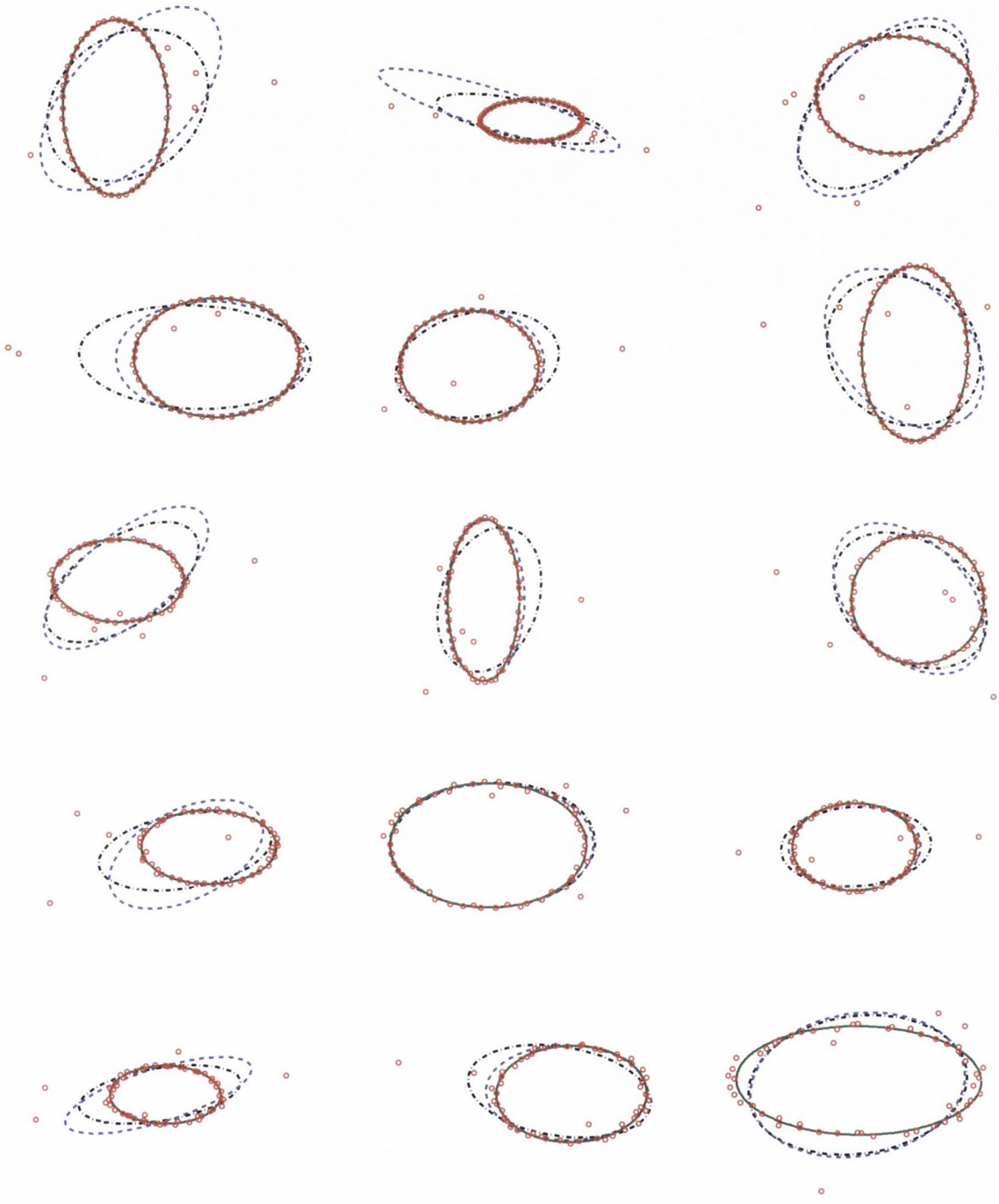


Figure 7.10: Examples of the experiments performed. For all cases, the red dot is the observation. The estimated ellipse for the Least square method (Blue), Direct fit (black) and the proposed method (green).

7.2.2 Detecting Multiples ellipses

In this section we explore the problem of detecting multiple ellipses in digital images. We use the position and curvature of the edge map of the images as information $\mu_i = \{(u_i, \psi_i)\}_{i=1\cdots,n}$. This information is encoded in a multidimensional density function (as described in section 4.3). The details of the optimisation algorithm implemented is described in Appendix B.4.

We use the benchmark data set provided by Chia et al.[10] for testing. This data contain 6 sets of synthetic images. Each set is created using a different number ν of occluded ellipses (with $\nu = 4, 8, 12, 16, 20$ and 24). Each set contains 50 images with a resolution of 300x300 pixels. An example of the images contained in the data base is shown in Figure 7.12. In the same Figure the information extracted from the images and used as observation is also displayed (input data) for our proposed algorithm. This data is computed using the edge detector of Matlab (canny) used with the default parameters. The normal vectors are computed using the convolution function with default parameters and applied in both directions of the image $(x - y)$.

The detection evaluation is performed using the overlap error. This error is computed using the detected ellipse E_d and the true ellipse E_t as follows:

$$OverlapError(E_d, E_t) = 1 - \frac{Area(E_t) \cap Area(E_d)}{Area(E_t) \cup Area(E_d)} \tag{7.3}$$

An ellipse is considered detected when the overlap error is less that 0.05. A common index for detector evaluation is the F-measure. The F-measure combines the precision and recall of the detector. Precision is defined as follows:

$$P = \frac{Number\ of\ correctly\ detected\ ellipses}{Total\ Number\ of\ ellipses\ detected} \tag{7.4}$$

and the Recall is:

$$R = \frac{Number\ of\ correctly\ detected\ ellipses}{Total\ Number\ of\ ellipses\ present\ in\ the\ test\ image} \tag{7.5}$$

One of the advantages of our algorithm is its simplicity in terms of parameter setting. The only relevant value to set is the bandwidths used for the density function ($h_{max} = 7$ and $h_{min} = 1$) and the threshold used for updating the observation $t_1 = 0.3$ and $t_2 = 0.1$ (cf. Section 7.1). We define the covariance matrices as described in section 4.19. In order to optimise the efficiency of the algorithm the number of Gaussians in the mixture representing the observation is reduced as described in section 4.2.5.

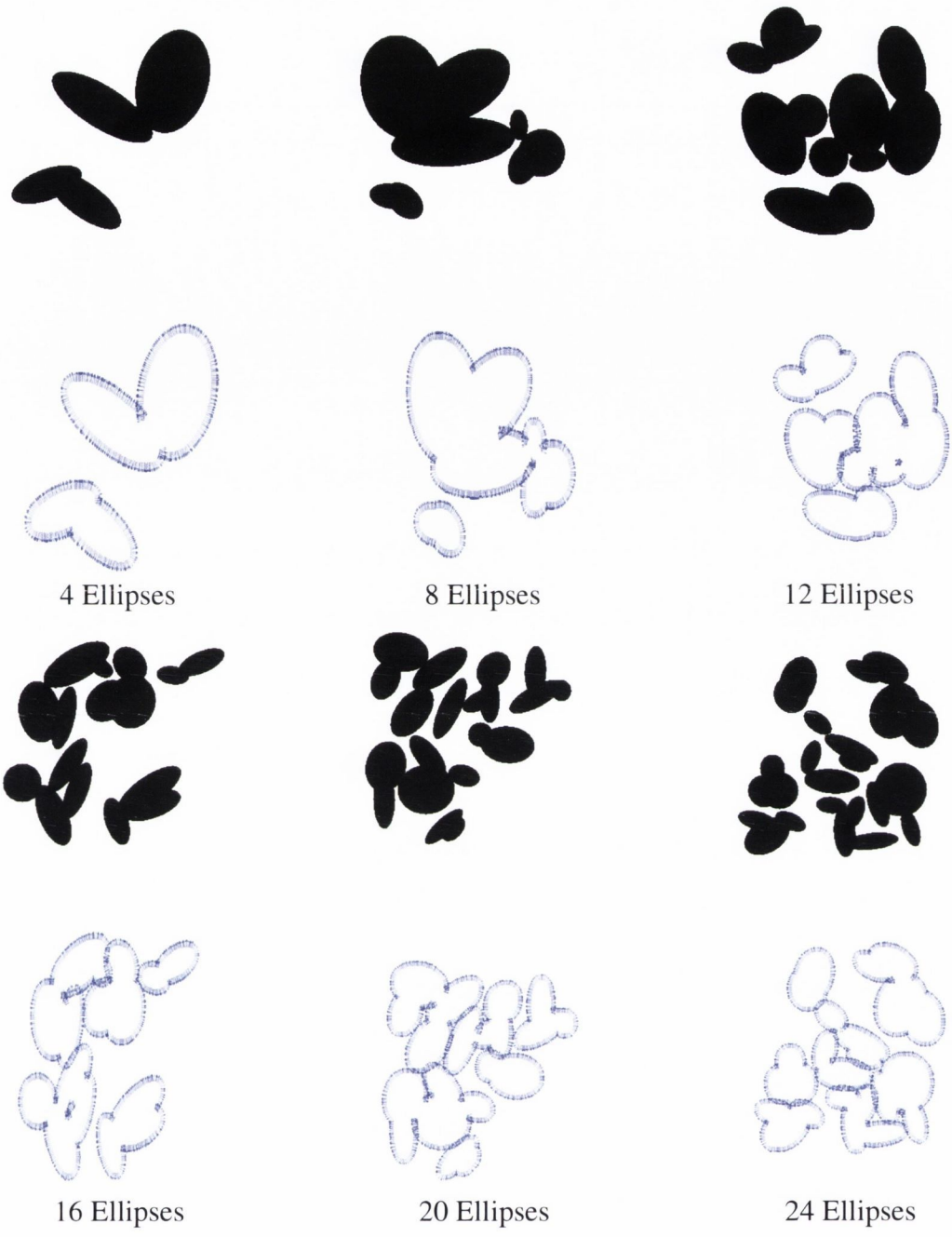


Figure 7.12: This Figure shows an example of the data set used for testing the algorithm. This data set was published by [10] and it contains 6 sets of 50 images. Each set contains a different number of occluded ellipses from 4 to 24. In the first and third row the original images are displayed while in the second and forth the observations extracted are shown as a map of normal vector corresponding to the edge points in the original image.

Figure 7.13 shows the results obtained when applying our algorithm to images containing 4 occluded ellipses. In these cases all the instances of the ellipses are detected. However, when increasing the number of ellipses in the image the results deteriorate. An example is shown in Figure 7.14 where the white and black image correspond to the input

image (first and third row). The blue dots correspond to the observations taken from the input image. The red ellipses are the ellipses correctly detected while the green ellipses are detections that do not properly represent the observation. When more than 20 ellipses are present, the proportion of correctly detected ellipses is very small.

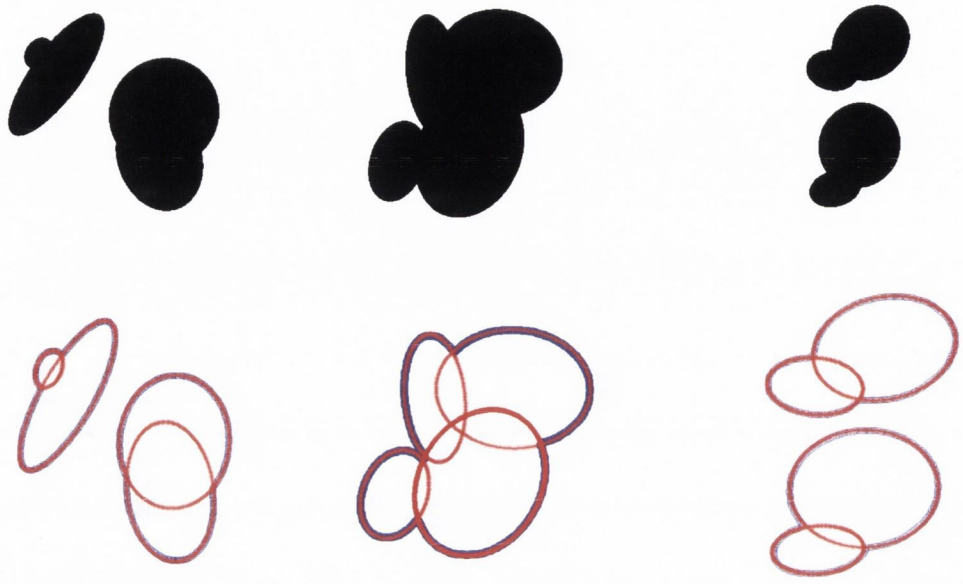


Figure 7.13: Example of the results obtained when applying our proposed algorithm for detecting multiples ellipses. On the top row the original images and at the bottom the detected ellipses (red) and the observations (blue).

We analyse the metrics of recall and precision when evaluating all images in the data set. The results are reported in Figure 7.15 (performance of state of the art algorithms are also reported for comparison [10, 11, 12, 14, 13]). Figure 7.15 suggests that our method outperforms methods based on the Hough Transform [14, 13]. Methods based on connected edges, on the other hand, show better performance than our algorithm. The algorithm proposed by Chia et al. [10] shows outstanding performance and it is able to detect a high number of ellipses. However, it only works when the observations are a set of connected edges. In contrast, our algorithm does not have any limitation and it can still be used when the observation is a sparse data set of points.

Additionally in Figure 7.16 we show the performance of the detector when relaxing the overlap error in between the detected ellipse and the true ellipse. This experiment was performed using the set of images containing 4 occluded ellipses. Additional advantages of our proposed method are its feasibility for including prior information about the parameters to estimate and the fact that it allows us to include any information related to the the shape of interest. Any extra information can be added as an additional dimension of the GMM. Figure 7.17 shows an example where a prior was used for detecting coins in an image.

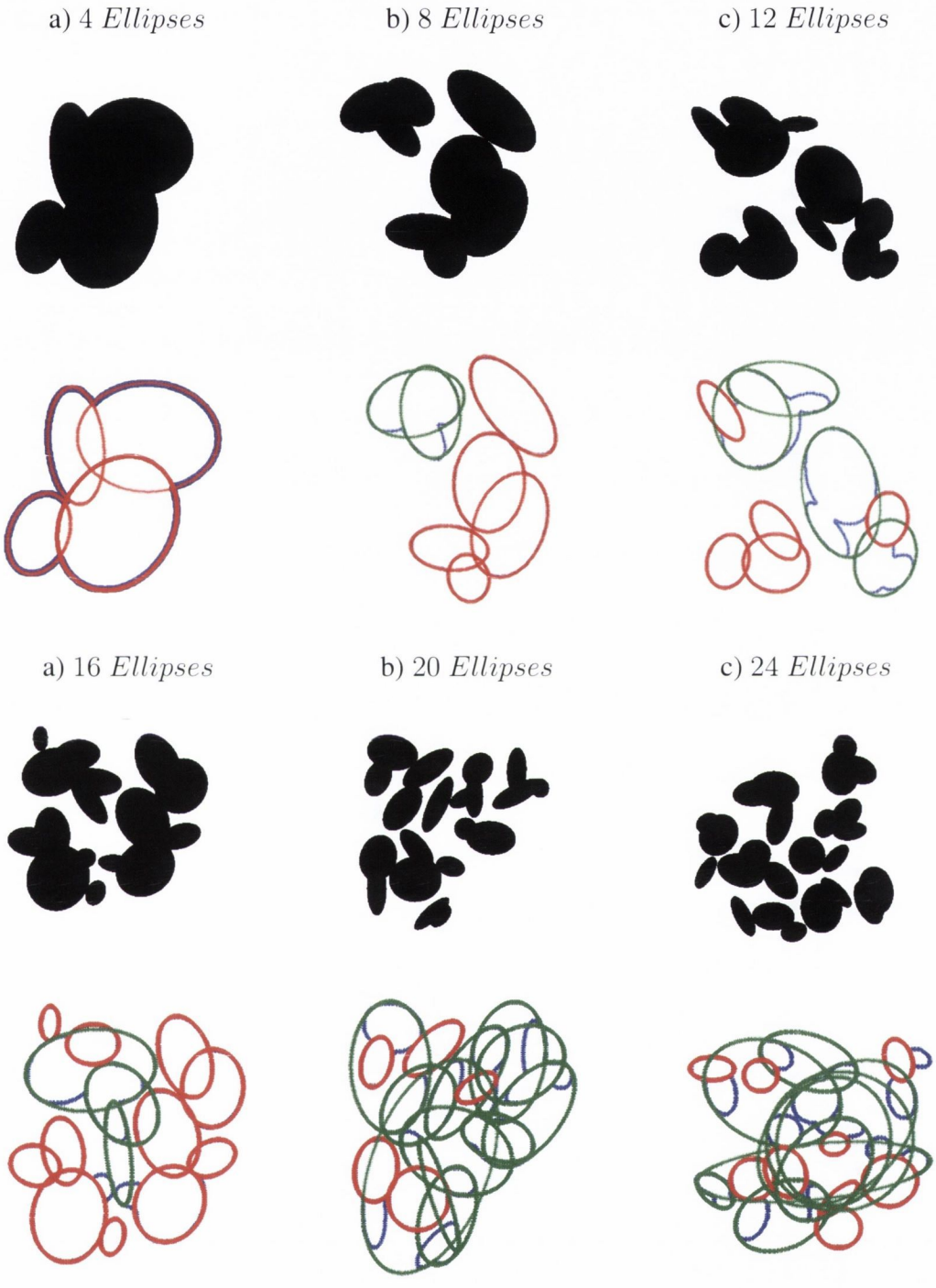


Figure 7.14: Example of the results obtained when applying our algorithm to data sets containing 4,8,12,16,20 and 24 occluded ellipses respectively.

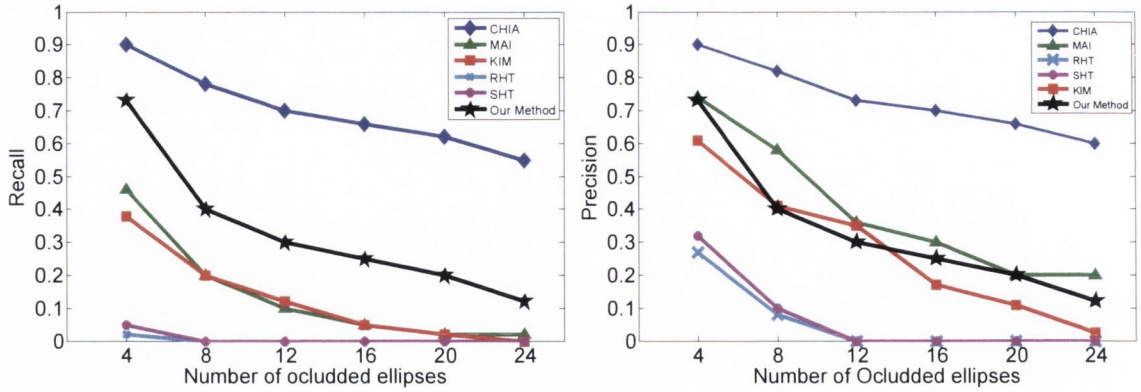


Figure 7.15: Testing on synthetic images containing occluded images. In Figure a) we report the values obtained for the Recall while in b) the results for precision. Each set (from 4 to 24) was evaluated using 50 images. For comparison we report the results obtained using the approaches proposed by Chia et al.[10], Mai et al. [11], Kim et al [12] and the Hough transform based methods RHT and SHT proposed by [13] and [14] respectively.

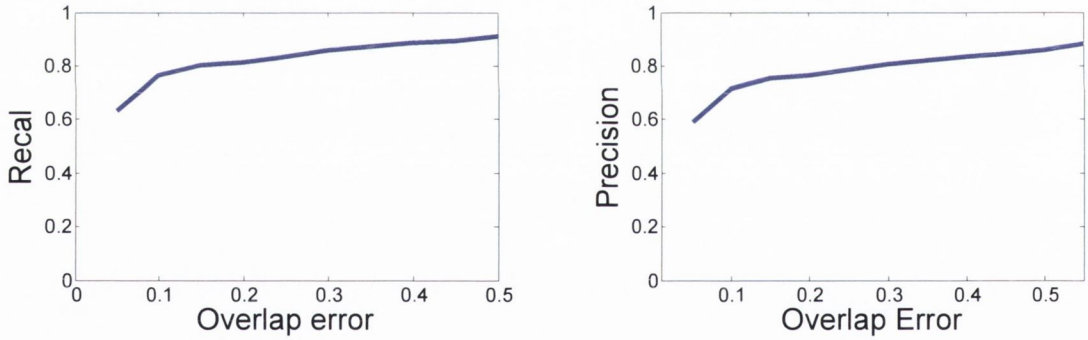


Figure 7.16: Testing our algorithm on 50 synthetic images each containing four 4 ellipses occluded as overlap error is varied from 0.05 to 0.55.

Figure 7.17: Examples of detecting ellipses in RGB images.

7.3 Conclusion

In this Chapter we have shown two extensions to the Bayesian-$\mathcal{L}_2$ framework. The first one corresponds to a method to iteratively estimate multiple instances of a shape. The second one, corresponds to the estimation of parametric curves (ellipse). In this case we use multidimensional modelling for the GMM where the position of the contour pixels and its curvature are considered. We used both extensions to detect multiple ellipses in a benchmark data set. Results compared with state of the art shows the promising performance of our algorithm. The feasibility of using prior information makes our method very attractive for solving particular detection problems where some knowledge about the parameters to estimate is available.

We have proposed here a bottom-up greedy approach to multiple instance detection. This strategy was chosen due the fact we assume there is no information about the number of instances the shape appears in the image. However, when such information is available it could be included in the model by fixing the number of parameters to estimate.

We have shown in this Chapter an evaluation of the proposed method using benchmark data sets. This allowed us to achieve a fairness comparison with state of the art techniques. However, the performance of the algorithm when using real data depends on the characteristics of the data sets, the amount of outliers and the information chosen for describing the shape (i.e edge position, color, curvature, etc.).

Chapter 8

Conclusion

Estimating the parameters of shapes is a crucial task for many applications in computer vision. For this reason many methods have been proposed in literature. The most recent approaches have been based on robust statistical inference methods. We have explored in this thesis the use of the divergence between probability density functions as a robust metric for shape parameters inference. We have studied in particular the $\mathcal{L}_2$ metric and proposed a framework for including prior information when estimating shape parameters. We have described shapes as GMMs and studied the role of the bandwidth in their modelling. The main results and conclusions obtained in this thesis and the future perspective of the work carried out are summarised as follows.

8.1 Summary

We have proposed in this thesis a method for solving the estimation of shape parameters. The main contributions of this method are reviewed here:

1. **Bayesian framework based on the divergence metric $\mathcal{L}_2$ for shape parameters estimation**: We have presented a Bayesian framework for shape parameter estimation. The data term in the Bayesian expression is considered as a sort of likelihood defined using the $\mathcal{L}_2$ distance between two density functions f and g. These density functions f and g are modelled using GMMs and represent the target shape (observation) and the model shape respectively.

2. **Modelling GMMs better suited for representing shapes:** We have shown that the bandwidth plays a key role in statistical inference for estimating parameters and in particular when using divergence metrics such as the $\mathcal{L}_2$ distance. We proposed to model GMMs for representing shapes (curve, surface and images) using non-isotropic covariance matrices. Non-isotropic covariance matrices allow us to include information about the structure of the shape and its geometry. This rep-

resentation not only helps in the shape description but also in the convergence of the optimisation algorithm used for the estimation. Additionally, the non-isotropic representation allows us to reduce the number of Gaussians in the mixture without compromising the representation of the shape. Hence, the efficiency of the optimisation algorithm is improved.

3. **GMMs as a method for encoding relevant information from shapes** We have shown that the dimensionality of GMMs can be increased by adding extra information about the shape of interest (rather than just the contour). Increasing the dimensionality of the GMM improves the convergence of the optimisation algorithm without affecting its efficiency.

We have tested and explored the performance of the proposed framework when dealing with challenging parameter estimation problems. The main results obtained can be summarised as follows:

1. **Method for computing the affine transformation:** We solve the rigid transformation and the scaling between data sets. We propose a Mean Shift algorithm to solve the optimisation when the modelling of the GMM is defined as isotropic. This particular case is comparable to state of the art registration algorithms. However, thanks to the annealing strategy implemented, our algorithm achieves better performance and it has been shown to be less sensitive to the starting guess. Additionally, we have demonstrated the advantage of using non-isotropic modelling for the GMMs. The estimation algorithm, in this case, shows better performance and it allows for the tackling of problems where two data sets are sampled differently (strongly sub-sampled data or with occlusions).

2. **Method for fitting shape models:** We have shown the possibility of using the Bayesian-$\mathcal{L}_2$ framework for fitting morphable models to data sets. We have proposed a Mean Shift algorithm to solve the optimisation problem. We have shown that a morphable model can be successfully fitted even when the observation is a sparse set of points. This method has the advantage that it does not need any correspondence between the morphable model and the observations. Additionally, we have shown the advantage of the non-isotropic modelling for reducing complexity of the optimisation algorithm and improving robustness in the estimation.

3. **Method for fitting parametric curves:** We have shown that the Bayesian-$\mathcal{L}_2$ framework proposed is suitable for estimating parametric curves. We have proposed a method for fitting and detecting ellipses in images. The algorithm allows for the inclusion of relevant information about the shape such as the curvature. This information is included by adding an extra dimension to the GMMs.

4. **Method for estimating multiple instances of a shape:** We have proposed an extension of our framework to detect multiples instance of a shape. It is based on using our algorithm iterativelly such that the observation set is updated to allow the detection of a new instance. We applied this algorithm for detecting multiples ellipses in digital images achieving results comparable to the state of the art techniques.

8.2 Limitations and future perspectives

We have shown here a global method for solving the estimation of parameters between data sets. As a top-down method some inherent limitations have been identified:

- The global solution given by the cost function proposed may not necesarily represent the right solution. This may occur when the proportion of outliers is too big or when there is not enough distinctiveness in the data sets (as a shape descriptor). However, this problem can be reduced when using the non-isotropic modelling for the GMM (cf. Chapter 5).

- When using the algorithm for fitting morphable models we face the same disadvantages of any learning method. The shape model will only deform according to the data used for its training (cf. Chapter 6).

- When detecting multiple intances of a shape the precision and recall rates depend on the first few detections. If any error occurs in the first few detections this is propagated affecting the final result 7. This problem is due the elimination of the observations along each iteration.

The results obtained from the work carried out in this thesis and the limitation identified suggest to further explore the following lines of research:

Including more information in the density function: We have shown in this thesis that GMMs allow for the encoding of relevant information about the shape of interest. We have used contour as the main descriptor of shape as well as the its curvature. However, depending of the estimation problem to solve, a more detailed description of the shape can be included in the GMM. It seems that adding more information such as colour and local texture (e.g SURF or SIFT descriptor) would lead to more powerful GMMs. Adding dimensions to the GMM is not computationally expensive compared to adding new observations.

Augmenting morphable shape models: An inherent limitation of morphable shape models is that they encode only variabilities that were present in the training set of shapes used for learning (this is a standard limitation of all learning techniques). Hence, new shapes (from the same class) may not be well represented by the morphable model. This limitation could be overcome by augmenting the morphable model each time that a new shape is fitted. This can be done by exploring our framework when relaxing the prior term. New constraints and prior can be explored and defined directly over the modelling of the GMM.

Speed-up: We have paid attention to the computational cost of our technique from the point of view of modelling an efficient cost function. However, the computational cost can be further reduced by a more suitable implementation of the optimisation algorithm. This could be an interesting direction for future work where strategies such as parallel programming can be explored.

Appendix A

Mathematical Expressions

A.1 Closed form solution for $\mathcal{L}_2$

Given the two Gaussian distributions over the random variable $\mathbf{x}$ in $\mathbb{R}^D$:

$$\mathcal{N}(\mathbf{x}; u, \Sigma_u) = \frac{1}{(2\pi)^{\frac{D}{2}}\sqrt{|\Sigma_u|}} \exp\left(-\frac{1}{2}(x-u)^T\Sigma_u^{-1}(x-u)\right) \tag{A.1}$$

$$\mathcal{N}(\mathbf{x}; v, \Sigma_v) = \frac{1}{(2\pi)^{\frac{D}{2}}\sqrt{|\Sigma_v|}} \exp\left(-\frac{1}{2}(x-v)^T\Sigma_v^{-1}(x-v)\right) \tag{A.2}$$

The integral over the random variable $\mathbf{x}$ of the product of the two density functions is expressed in terms of their means and covariance matrices as follows [141, 144]:

$$\int_{\mathbb{R}^{d\mathbf{x}}} \mathcal{N}(\mathbf{x}; u, \Sigma_u)\, \mathcal{N}(x; v, \Sigma_v) d\mathbf{x} = \frac{1}{(2\pi)^{\frac{D}{2}}\sqrt{|\Sigma_u + \Sigma_v|}} \exp\left(-\frac{q(u,v)}{2}\right) \tag{A.3}$$

where,

$$q(u,v) = u^T\Sigma_u^{-1}u + v^T\Sigma_v^{-1}v - \mu^T\Sigma^{-1}\mu$$

$$\Sigma^{-1} = \Sigma_u^{-1} + \Sigma_v^{'-1}$$

$$\mu = \Sigma\left(\Sigma_u^{-1}u + \Sigma_v^{-1}v\right)$$

A.2 $\mathcal{L}_2$ and $\mathcal{L}_2$E using isotropic covariance matrices

Let us consider two sets of points modelled as density functions with the same covariance matrices $\Sigma_i^g = \Sigma_i^f = h^2 I \quad \forall i$, as follows:

$$g(\mathbf{x}|\Theta) = \sum_{i=1}^{n_g} w_i^g\, \mathcal{N}(x; \mu_i^g(\Theta), \Sigma_i^g)$$

$$f(\mathbf{x}) = \sum_{i=1}^{n_f} w_i^f \, \mathcal{N}(x; \mu_i^f, \Sigma_i^f)$$

When considering a rigid transformation between the two sets, the cost function is:

$$\hat{\Theta} = \arg\max_{\Theta} \left\{ \mathcal{C}(\Theta) = \int \{ \, g(\mathbf{x}|\Theta) \, f(\mathbf{x}) \, \} \, d\mathbf{x} \right\}$$

$$\mathcal{C}(\Theta) = \sum_{i=1}^{n_g} \sum_{j=1}^{n_f} \frac{w_i^g w_j^f}{(2\pi)^{\frac{D}{2}} \sqrt{2(h)^2}} \exp\left(\frac{-\|\mu_j^f - \mu_i^g(\Theta)\|^2}{4 \, h^2} \right)$$

Let us consider now the case where the reference data set is modelled as a Gaussian Mixture using isotropic covariance matrices $\Sigma_i^g = H^2 I \quad \forall i$. The observation is considered as the empirical distribution such as:

$$f(\mathbf{x}) = \frac{1}{n_f} \sum_{i=1}^{n_f} \delta(x - \mu_i^f)$$

The cost function becomes:

$$\mathcal{C}(\Theta) = \sum_{i=1}^{n_g} \sum_{j=1}^{n_f} \frac{w_i^g w_j^f}{(2\pi)^{\frac{D}{2}} \sqrt{(H)^2}} \exp\left(\frac{-\|\mu_j^f - \mu_i^g(\Theta)\|^2}{2 \, H^2} \right)$$

It is easy to see that those expression are equivalent when $H = \sqrt{2}h$.

Appendix B

Algorithms

B.1 Mean Shift algorithm for rigid transformation

We define a dedicated Mean Shift algorithm for estimating the transformation parameters (rotation R and translation $\mathbf{t}$) that maximise the cost function $\mathcal{C}(\Theta)$. In other words, making the two density function $f(\mathbf{x})$ and $g(\mathbf{x}|\boldsymbol{\Theta})$, corresponding to the observations and reference data sets respectively, be as close as possible. The two density functions are modeled using isotropic covariance matrices and expressed as follows:

$$g(\mathbf{x}|\Theta) = \sum_{i=1}^{n_g} w_i^g \, \mathcal{N}(x; \mu_i^g(\Theta), (h_i^g)^2 I)$$

$$f(\mathbf{x}) = \sum_{i=1}^{n_f} w_i^f \, \mathcal{N}(x; \mu_i^f, (h_i^f)^2 I)$$

The cost function can then be expressed as follows:

$$\mathcal{C}(\Theta) = \sum_{i=1}^{n_g} \sum_{j=1}^{n_f} \underbrace{\frac{w_i^g w_j^f}{(2\pi)^{\frac{D}{2}} \sqrt{(h_i^g)^2 + (h_j^f)^2}} \exp\left(\frac{-\|\mu_j^f - B(\Theta)\|^2}{2\left((h_i^g)^2 + (h_j^f)^2\right)}\right)}_{E(\Theta)}$$

With,

$$B(\Theta) = \underbrace{\begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}}_{\mathrm{R}} \mu^g + \mathbf{t} \tag{B.1}$$

The expression E is not linear w.r.t. Θ as it needed for defining a Mean Shift algorithm. The function $B(\Theta) = \mathrm{R}\mu^g + \mathbf{t}$ is defined for $\Theta \in [-\pi; \pi] \times \mathbb{R}^2$ to $B(\Theta) \in \mathbb{R}^2$. The

Taylor expansion of this vector valued function around $\Theta^{(0)}$ is:

$$B(\Theta) = \underbrace{B(\Theta^{(0)}) + DB(\Theta^{(0)}) \underbrace{(\Theta - \Theta^{(0)})}_{\delta\Theta}}_{B(\delta\Theta, \Theta^{(0)}, \mu^g)} + h.o.t$$

where $DB(\Theta^{(0)}, \mu^g)$ corresponds to the 2×3 matrix of the partial derivatives of B computed at $\Theta^{(0)}$ (for the point $\mu^g = (\mu^g_x, \mu^g_y)$):

$$DB(\Theta^{(0)}, \mu^g) = \begin{pmatrix} -\mu^g_x \sin\phi^{(0)} - \mu^g_y \cos\phi^{(0)} & 1 & 0 \\ \mu^g_x \cos\phi^{(0)} - \mu^g_y \sin\phi^{(0)} & 0 & 1 \end{pmatrix} \tag{B.2}$$

The first order approximation $B(\delta\Theta, \Theta^{(0)}, \mu^g)$ is now linear w.r.t. $\delta\Theta$ and the expression E is modified as follow:

$$E_{ij}(\delta\Theta, \Theta^{(0)}) = \frac{w^g_i w^f_j}{2\pi\sqrt{(h^g_i)^2 + (h^f_j)^2}} \times \exp\left(\frac{-\|\mu^f_j - B(\delta\Theta, \Theta^{(0)}, \mu^g_i)\|^2}{2\left((h^g_i)^2 + (h^f_j)^2\right)}\right) \tag{B.3}$$

The modified cost function to maximise is now defined near $\Theta^{(0)}$ by:

$$\mathcal{C}(\delta\Theta, \Theta^{(0)}) = \sum_{j=1}^{n_f} \sum_{i=1}^{n_g} E_{ij}(\delta\Theta, \Theta^{(0)}) \tag{B.4}$$

and the estimation of Θ is done iteratively by maximising $\mathcal{C}$ w.r.t $\delta\Theta$ (see algorithm 5).

Algorithm 5 Estimation of Θ.

Input: $t = 0, \Theta^{(0)}, e, M$
  repeat
    $\widehat{\delta\Theta} = \arg\max_{\delta\Theta} \mathcal{C}(\delta\Theta, \Theta^{(t)})$ (algorithm 6)
    $\Theta^{(t+1)} = \Theta^{(t)} + \widehat{\delta\Theta}$
    $t \leftarrow t + 1$
  until $\|\Theta^{(t+1)} - \Theta^{(t)}\| \leq e$ or $t > M$
Output: $\hat{\Theta} = \Theta^{(t)}$

Algorithm 6 Estimation of $\delta\Theta$.

Input: $s = 0, \Theta^{(t)}, \delta\Theta^{(0)} = 0, e, N$
  repeat
    $\delta\Theta^{(s+1)} = \mathrm{A}(\delta\Theta^{(s)}, \Theta^{(t)})\ \mathbf{b}(\delta\Theta^{(s)}, \Theta^{(t)})$
    $s \leftarrow s + 1$
  until $\|\delta\Theta^{(s+1)} - \delta\Theta^{(s)}\| \leq e$ or $s > N$
Output: $\widehat{\delta\Theta} = \delta\Theta^{(s)}$

An iterative Mean Shift algorithm can then be calculated to optimise the cost function $\mathcal{C}$

in (B.4) w.r.t. $\delta\Theta$ (algorithm 6). The 3×3 matrix A is defined as:

$$\mathrm{A}(\delta\Theta^{(s)}, \Theta^{(t)}) = \left(\sum_{j=1}^{n_f} \sum_{i=1}^{n_g} \frac{E_{ji}(\delta\Theta^{(s)}, \Theta^{(t)})}{2((h_i^g)^2 + (h_j^f)^2)} DB(\Theta^{(t)}, \mu_i^g)^T DB(\Theta^{(t)}, \mu_i^g) \right)^{(-1)} \quad \text{(B.5)}$$

The vector b is defined as:

$$\mathbf{b}(\delta\Theta^{(s)}, \Theta^{(t)}) = \sum_{j=1}^{n_f} \sum_{i=1}^{n_g} \frac{E_{ji}(\delta\Theta^{(s)}, \Theta^{(t)})}{2((h_i^g)^2 + (h_j^f)^2)} DB(\Theta^{(t)}, \mu_i^g)^T \, (\mu_j^f - B(0, \Theta^{(t)}, \mu_i^g)) \quad \text{(B.6)}$$

In order to avoid algorithm 5 to be dependent on the initial guess $\Theta^{(0)}$ and to prevent the estimate of Θ to be trapped in a local maximum, an annealing strategy is implemented using the bandwidths as a temperature [145]. Starting with large bandwidths, these are decreased iteratively using a geometric rate up to a minimum value (algorithm 7). The limits for the bandwidths, $\{lh_i^g, lh_j^f\}$, can be set automatically using nearest neighbours or manually chosen.

Algorithm 7 Estimation of Θ with simulated annealing.

Input: $\Theta^{(0)}$, $\{h_i^g, h_j^f\}$ large $\forall(i,j)$, $0 < \beta < 1$
  repeat
    $\hat{\Theta} = \arg\max_{\Theta} \mathcal{C}(\Theta)$ (algorithm 5)
    for $j = 1 \rightarrow n_f$ **do**
      if $h_j^f > lh_j^f$ **then**
        $h_j^f \leftarrow \beta \, h_j^f$
      end if
    end for
    for $i = 1 \rightarrow m$ **do**
      if $h_i^g > lh_i^g$ **then**
        $h_i^g \leftarrow \beta \, h_i^g$
      end if
    end for
  until $h_i^g < lh_i^g \; \forall i$ and $h_j^f < lh_j^f \; \forall j$
Output: $\hat{\Theta}$

B.2 Mean Shift algorithm for shape fitting

We rewrite the cost function in Expression 4.7 as function of the parameters to estimate $\Theta = [\alpha_1, \alpha_2, \alpha_3, ...\alpha_q]$ and according to the prior information given by the a morphable model as follows:

$$\hat{\alpha} = \arg\min_{\alpha} \left\{ \mathcal{C}(\alpha) = \frac{\mathcal{L}_2(\alpha)}{\tau^2} + \sum_{j=1}^{J} \frac{\alpha_j^2}{\sigma_j^2} \right\} \quad \text{(B.7)}$$

Table B.1: Expressions for computing $L_{(j,s)}$ and $b_{(j)}$

$$L_{(j,s)}$$

$$= 2\sum_{k=1}^{n_g}\sum_{i=1}^{n_f} E_2^{ki}(T_{jk}^T\Sigma_k^{-1} - T_{jk}^T\Sigma_k^{-1}\Sigma_{(ki)}^{-1T}\Sigma_k^{-1T})T_{sk} - \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(T_{jk}^T\Sigma_k^{-1}T_{sk} + T_{jp}^T\Sigma_p^{-1}T_{sp})$$
$$+ \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(T_{jk}^T\Sigma_k^{-1} + T_{jp}^T\Sigma_p^{-1})\Sigma_{(kp)}^{-1T}\Sigma_p^{-1T}T_{sp} + \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(T_{jk}^T\Sigma_k^{-1} + T_{jp}^T\Sigma_p^{-1})\Sigma_{(kp)}^{-1T}\Sigma_k^{-1T}T_{sk}$$

$$\mathbf{b}_{(j)}$$

$$= -2\sum_{k=1}^{n_g}\sum_{i=1}^{n_f} E_2^{ik}(\overline{v}_k^T\Sigma_k^{-1})T_{jk} + 2\sum_{k=1}^{n_g}\sum_{i=1}^{n_g} E_2^{ik}((\mu_i^{gT}\Sigma_i^{'-1} + \overline{v}_k^T\Sigma_k^{-1})\Sigma_k^{-1}\Sigma_{c(ki)}^{-1T})T_{jk}$$
$$- \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(\overline{v}_k^T\Sigma_k^{-1} + \overline{v}_p^T\Sigma_p^{-1T})\Sigma_{c(kp)}^{-1T}\Sigma_k^{-1T}T_{jk} - \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(\overline{v}_k^T\Sigma_k^{-1} + \overline{v}_p^T\Sigma_p^{-1T})\Sigma_{c(kp)}^{-1T}\Sigma_p^{-1T}T_{jp}$$
$$+ \sum_{k=1}^{n_g}\sum_{p=1}^{n_g} E_3^{kp}(\overline{v}_k^T\Sigma_k^{-1}T_{jk} + \overline{v}_p^T\Sigma_p^{-1T}T_{jp})$$

The term $\mathcal{L}_2(\alpha)$ comes from the density functions modelled using the two data sets; the observations $\left\{\mu_j^g\right\}_{j=1,\ldots,n_f}$ the model:

$$\mu_i^g(\alpha) = \overline{v}_i + \sum_{j=1}^{q} \alpha_j\, T_{ji} \;\; \forall i = 1, \ldots n_g \tag{B.8}$$

and the observations: The term τ is computed according to Expression 4.22 in section 4.2.3. The cost function $\mathcal{C}(\alpha)$ can then be written according to the closed form solution for the product of Gaussians as follows (cf. Appendix A.1):

$$\begin{aligned}\mathcal{C}(\alpha) = &\sum_{k=1}^{n_f}\sum_{p=1}^{n_f} E_1(u_k^f, u_p^f) - 2\sum_{i=1}^{n_g}\sum_{k=1}^{n_f} E_2(\mu_i^g(\alpha), u_k^f) \\ &+ \sum_{i=1}^{n_g}\sum_{p=1}^{n_g} E_3(\mu_i^g(\alpha), \mu_p^g(\alpha)) + \tau^2\sum_{j=1}^{J}\frac{\alpha_j^2}{\sigma_j^2}\end{aligned} \tag{B.9}$$

The Mean Shift Algorithm is computed by differentiating the cost function $\mathcal{C}(\alpha)$ with respect to α and equalling the result to zero. Starting from an initial guess $\alpha^{(t)}$, the update is computed by:

$$\alpha^{(t+1)} = \mathrm{A}(\alpha^{(t)})^{-1}\mathbf{b}(\alpha^{(t)}) \tag{B.10}$$

with A a $J \times J$ matrix defined as:

$$A_{j,s}(\alpha) = \begin{cases} L_{j,s}(\alpha), & \text{if } j \neq s \\ L_{j,s}(\alpha) + \frac{\tau^2}{\sigma_j^2}, & \text{if } j = s \end{cases} \tag{B.11}$$

The expression for L and b are presented in Table B.1. For simplicity $E_2(\mu_i^g(\alpha), u_k^f)$ and $E_3(\mu_i^g(\alpha), \mu_p^g(\alpha))$ are expressed as E_2^{ik} and E_3^{ip} respectively. The parameter $\overline{v}$ corresponds to the average shape of the model $\overline{v} = u^g(\alpha = 0)$ and the covariance matrices $\Sigma_{(ki)}$ are computed for each pair of distributions (k, i) as it was illustrated in Appendix

A.1. The Mean Shift algorithm is presented in Algorithm 8 with its annealing strategy. Starting from a maximum value h_{max}, the bandwidth is decreased using a geometric rate β until the minimum value h_{min} is reached. Note that the covariance matrix of each Gaussian in the density function of the model is updated at each iteration.

Algorithm 8 Estimation of α

Input: $\alpha_j^{(0)} = 0, \forall j$, h_{min}, h_{max}, $\gamma = 0.5$ and $\beta = 0.8$
  $h = h_{max}$
  repeat
    $\tau = \gamma\sqrt{\frac{\mathcal{L}_2(\Theta,h)}{J}}$
    repeat
      Compute $A(\alpha^{(t)})$ and $\mathbf{b}(\alpha^{(t)})$
      $\alpha^{(t+1)} = A(\alpha^{(t)})^{-1}\mathbf{b}(\alpha^{(t)})$
      Compute $\sum_k \ \forall k$
    until $|\alpha^{(t+1)} - \alpha^{(t)}| \leq e_o$
    $h \leftarrow \beta h$
  until $h \leq h_{min}$

B.3 Ellipse fitting

We have implemented an optimization algorithm for finding the parameters Θ that minimise the cost function in Equation 4.7. A point on an ellipse (i.e. at a given $t_i \in [0; 2\pi]$) can be created with the following equation:

$$\mu_i^g(\Theta) = \begin{pmatrix} a\cos t_i \\ b\sin t_i \end{pmatrix} \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} + \begin{pmatrix} x_o \\ y_o \end{pmatrix} \tag{B.12}$$

Defined by the latent variable to estimate: $\Theta = [\gamma, a, b, x_o, y_o]$ We solve the optimisation problem using a standard Newton Algorithm provided by Matlab combined with an annealing strategy. It is initialised using a big enough bandwidth h_{max} and then reduced using a geometric rate β until reach a minimum value h_{min}. In each iteration we compute the density function of the ellipse $g(\Theta)$ by updating the new sets of points $\mu_i^g(\hat{\Theta})$ sampled from the estimated ellipse. The algorithm implemented is summarised as follows:

B.4 Detecting ellipses in images

Here we summarise the algorithm used for detecting multiples ellipses in digital images. It is based on the Algorithm 9 and Algorithm 4 from Section 7.1. It uses a multiresolution strategy in order to improve its efficiency. We define a set of different levels of sampling (or image resolution) $R = [r^o, ..., r^z]$ and apply them to the image. Where r_o

Algorithm 9 Estimation of Θ.

Input: h_{max}, h_{min}, m, $\Theta^{(t=0)}$, e and β
  $h = h_{max}$
  Compute $\mu_i^g(\Theta^{(t)})$
  Compute $g(\Theta^{(t)})$
  repeat
    repeat
      $\Theta^{(t+1)} = \arg\min_\Theta \mathcal{C}(\Theta^{(t)})$
      update $\mu_i^g(\Theta^{(t+1)})$
      update $g(\Theta^{(t+1)})$
    until $\|\Theta^{(t+1)} - \Theta^{(t)}\| \leq e$
    $h = \beta h$
  until $h > h_{min}$
Output: $\widehat{\Theta} = \Theta^{(t)}$

represent a very low resolution image (highly sub-sampled) and r^z is the high resolution image. From this sub-sampled images the observation are computed. The algorithm estimate first the set of ellipses using the low resolution image and then uses that solution as starting guess for the next step where the observation is computed using the image with an increased resolution. The algorithm is iterated until the last step of resolution is computed. The number of steps in the algorithm (different resolutions) is defined by the user. In the case of the synthetic data set analysed in Section 7.2.2 we only have used two step. The general scheme of the algorithm is shown in Algorithm 10. The main differences in between Algorithm 4 and Algorithm 11 is that the first one uses random starting guess while the second one uses the solution given by the previous step of the general algorithm.

Algorithm 10 Multi-resolution for detecting multiples ellipses $S = \{\Theta_1, ...\Theta_s\}$.

Input: I=Image, $R = [r^0, ..., r^z]$,h_{max}, h_{min}, $\Theta^{(t=0)}$,t_1,t_2 and β
  $\{\mu_i^f\} \leftarrow$ extract observation(I, R^0)
  $\hat{S}^{(k)} \leftarrow$ EstimateEllipses1$(\{\mu_i^f\}, t_1, t_2, \beta, h_{max}, h_{min}, \Theta^{(t=0)})$ Algorithm 4
  repeat
    $\{\mu_i^f\} \leftarrow$ update observation(I, R^k)
    $\hat{S}^{(k+1)} \leftarrow$ EstimateEllipses2$(\{\mu_i^f\}, t_1, t_2, \beta, h_{max}, h_{min}, S^{(k)})$ Algorithm 11
    k^{++}
  until $k > s$
Output: $\widehat{S} = \hat{S}^{(k)}$

Algorithm 11 Estimating parameters of multiples instances of shape given a starting guess

Input: t_1, t_2, $\{u_i\}_{i=1,\cdots,n}$, h_{min},h_{max}, β and $S^{(o)} = \{\Theta_1^{(o)}, ...\Theta_s^{(o)}\}$

Init $\{u_i^s\}_{i=1,\cdots,n^s} = \{u_i\}_{i=1,\cdots,n}$

repeat

Init $h = h_{max}$ and $\hat{\Theta} = \Theta_k^{(0)}$

$\hat{\Theta} \leftarrow$ Estimate $\left(\mathcal{C}(\Theta), \hat{\Theta}, h, \tau, \beta\right)$ (cf. Algorithm 1, Section 4.2.3)

$\hat{\Theta}_s = \hat{\Theta}$

k^{++}

$\{u_i^s\}_{i=1,\cdots,n_s} \leftarrow$ Update $\left(\{u_i^{s-1}\}_{i=1,\cdots,n_{s-1}}, \hat{\Theta}_s, t_1\right)$

until $\frac{n-n^s}{n} < t_2$

Output: $\widehat{S} = \hat{S}^{(k)}$

Appendix C

Additional Results

C.1 Affine transformation

C.1.1 Sensitivity to noise

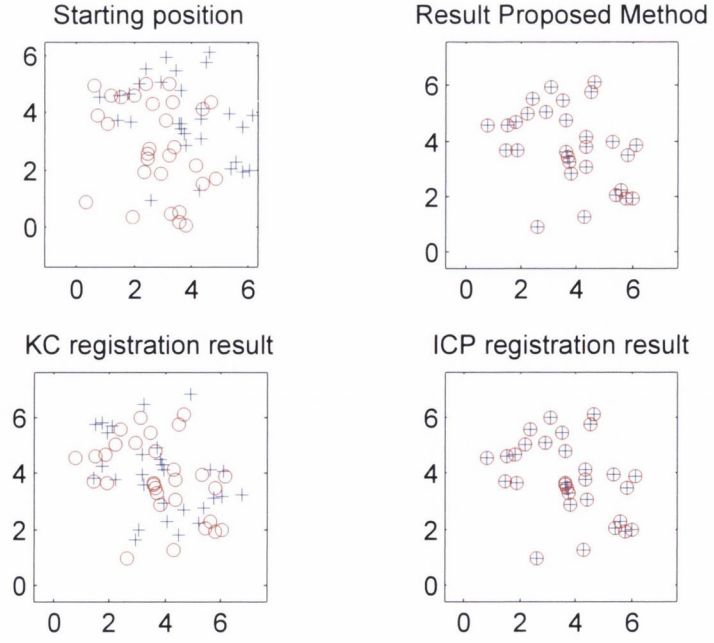


Figure C.1: Example 1 of results obtained for noise level 1 and angle of rotation 30^o

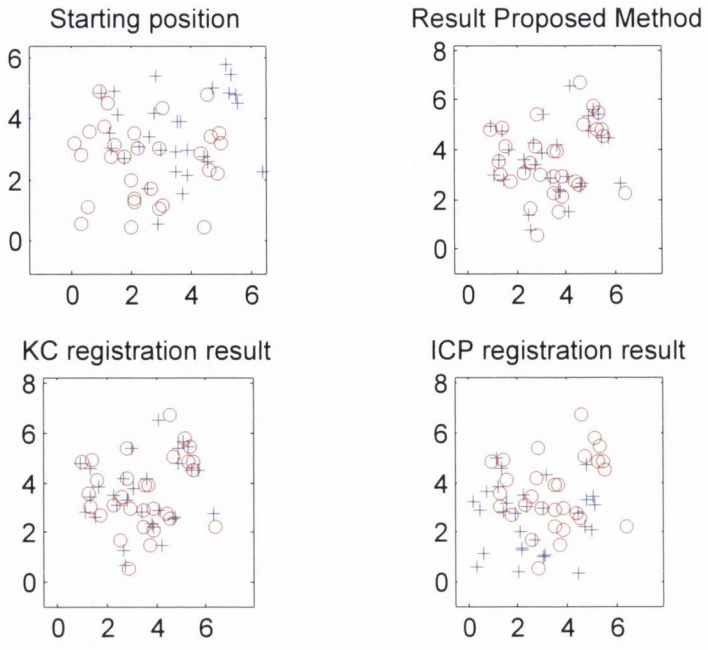


Figure C.2: Example 2 of results obtained for noise level 5 and angle of rotation 30^o

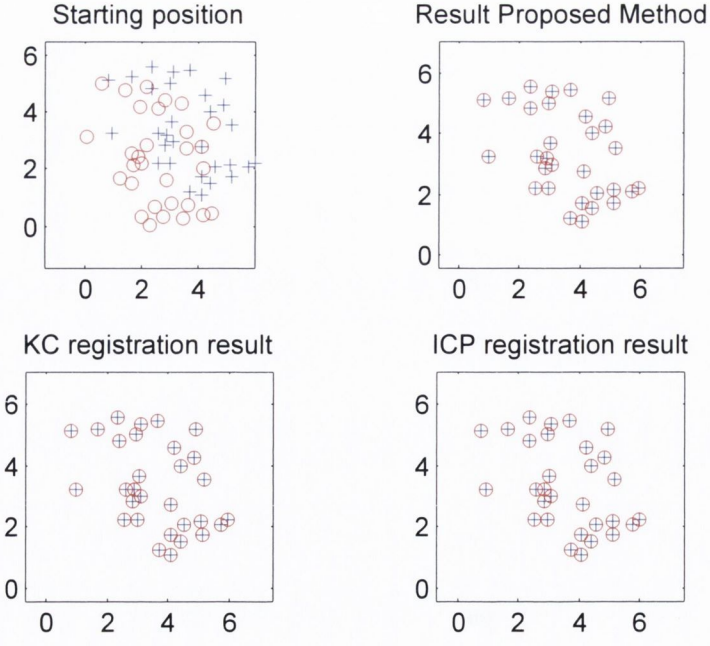


Figure C.3: Example 3 of results obtained for noise level 1 and angle of rotation 30^o

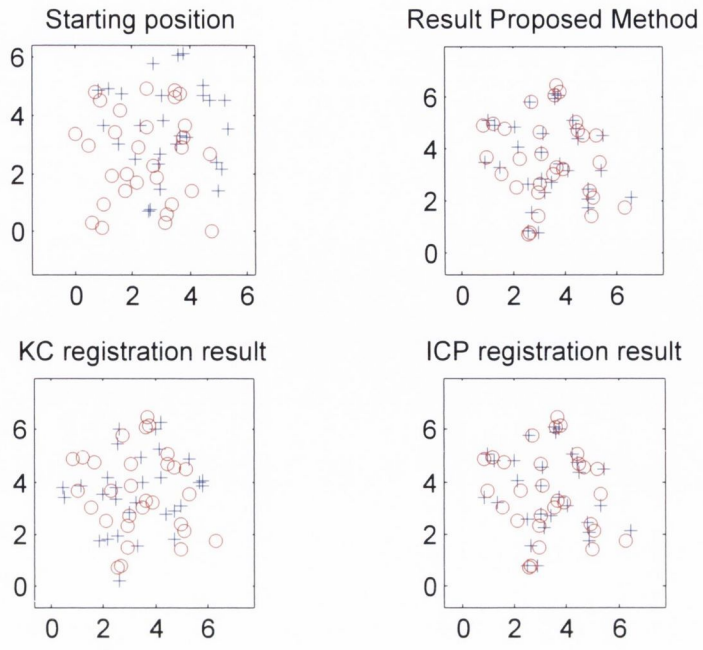


Figure C.4: Example 4 of results obtained for noise level 5 and angle of rotation 30^o

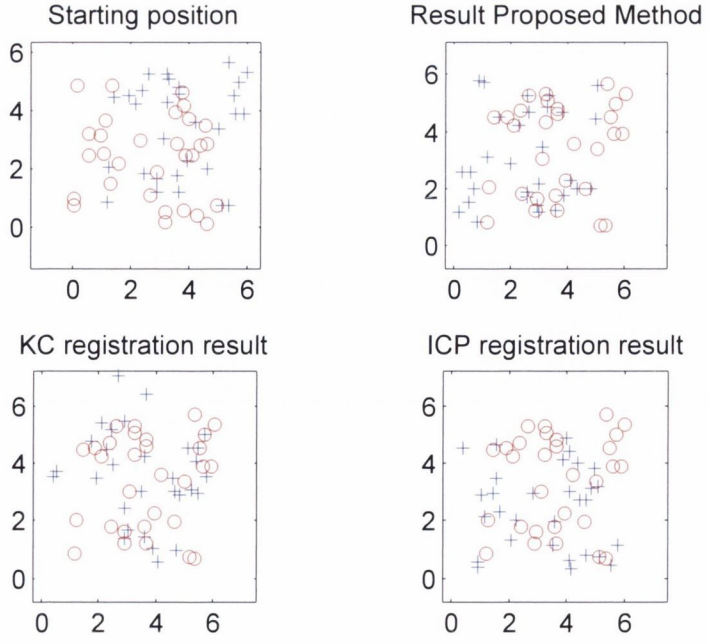


Figure C.5: Example 5 of results obtained for noise level 1 and angle of rotation 90^o

C.1.2 Robustness to outliers

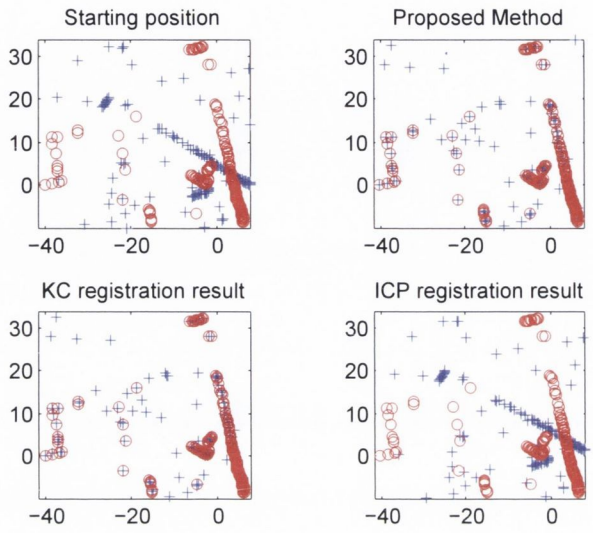


Figure C.6: Example 6 of results obtained when both sets are rotated in 45^o

C.1.3 Scaling estimation

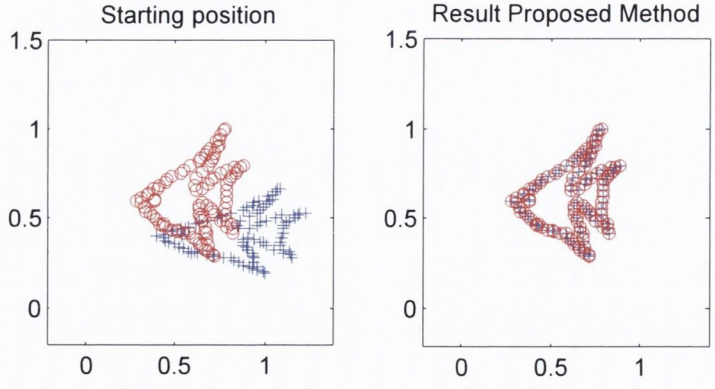


Figure C.7: Example 1 scaling

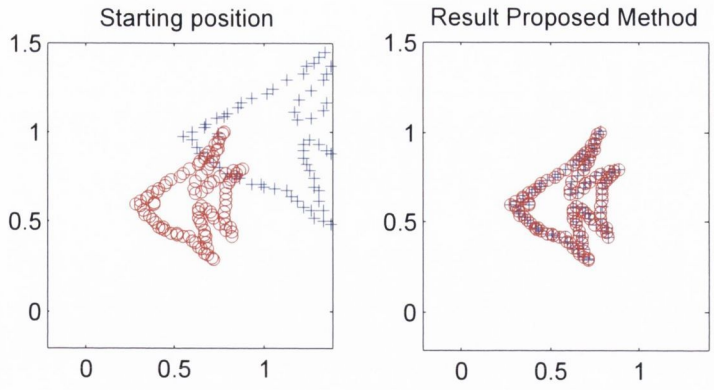


Figure C.8: Example 2 scaling

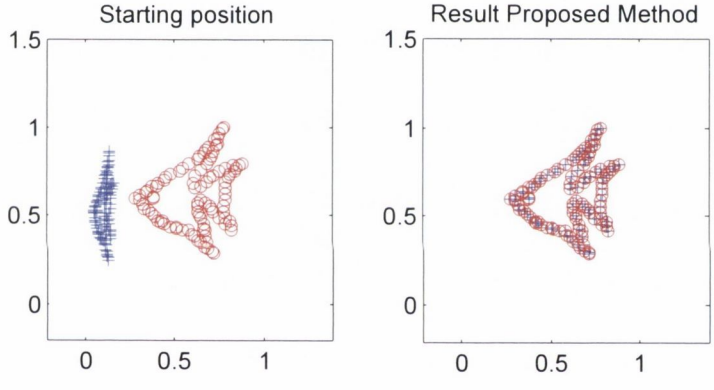


Figure C.9: Example 3 scaling

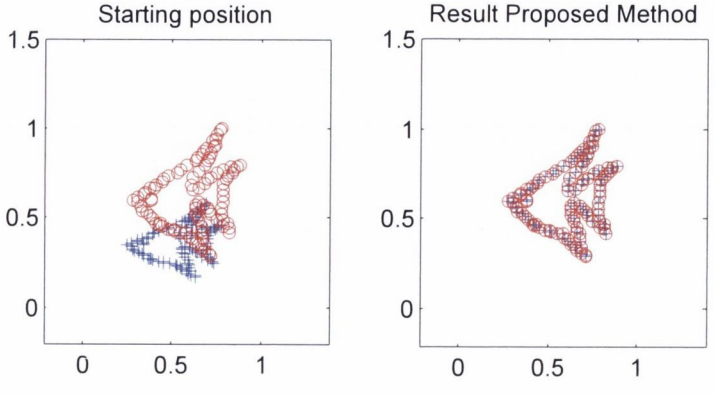
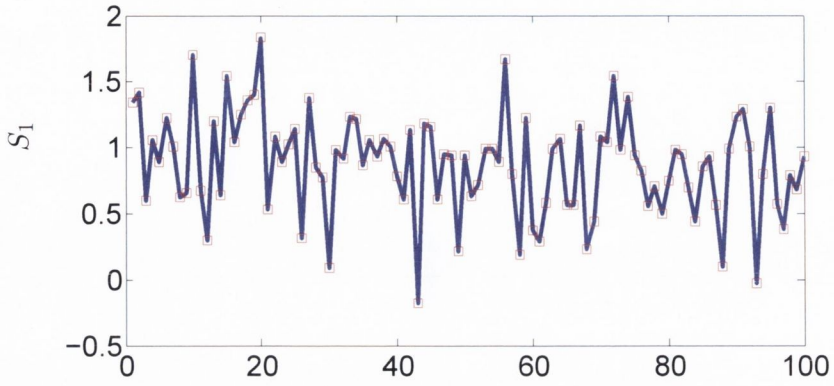


Figure C.10: Example 4 scaling



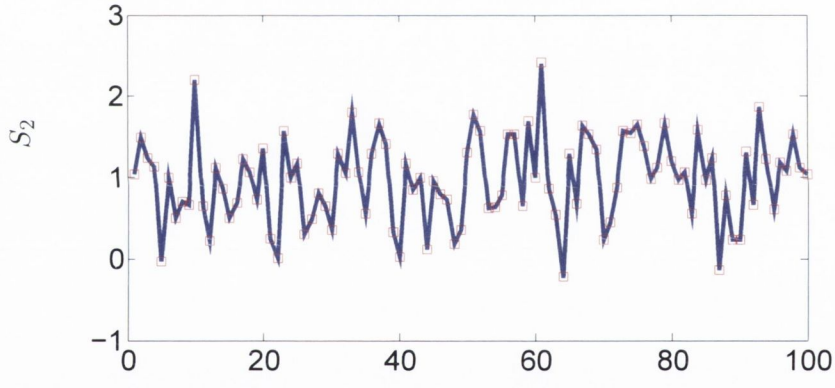


Figure C.11: Top: Results for s_1, estimated (blue line) and ground truth (red square). Bottom shows the results for s_2. The experiments where performed using the same data sets scaled using a normal distribution for both parameters. We run 100 experiments and all of them converge to the ground truth.

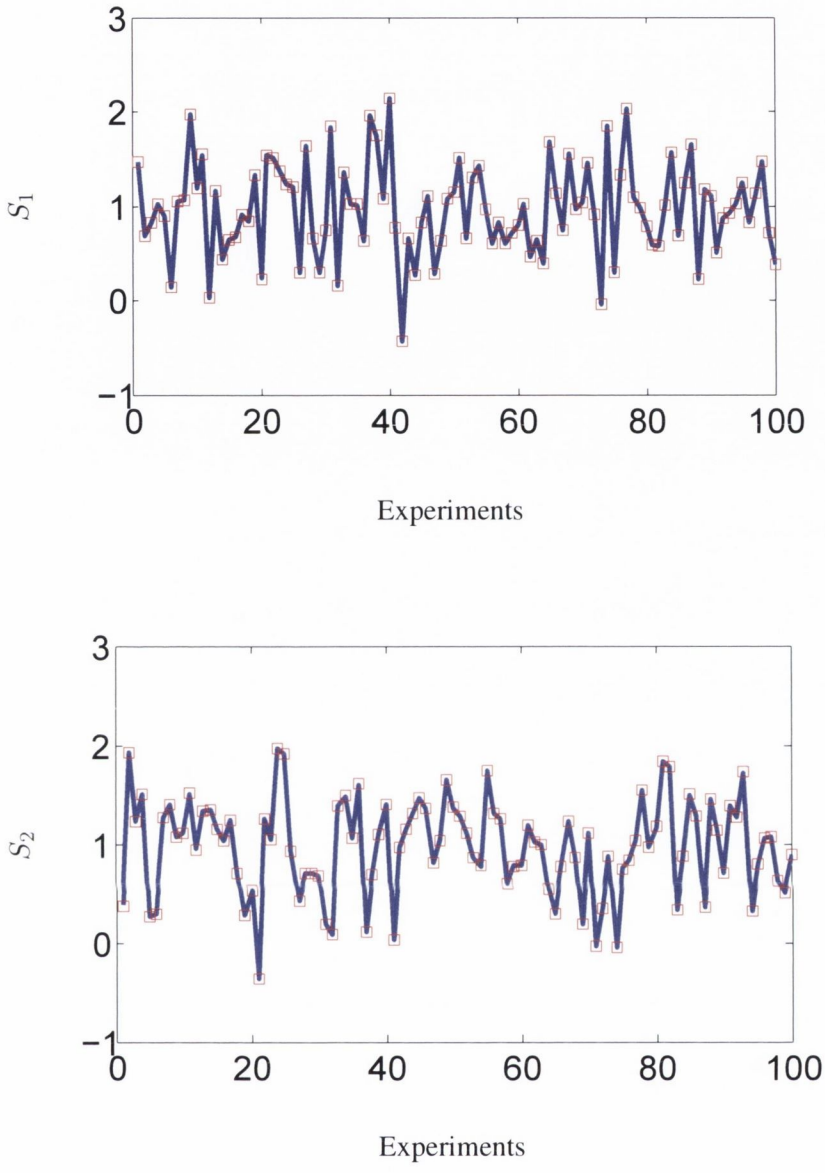


Figure C.12: Top: Results for s_1, estimated (blue line) and ground truth(red square). Bottom shows the results for s_2. The experiments where performed using the same data set but adding noise. The scaling is performed using a normal distribution for both parameters.

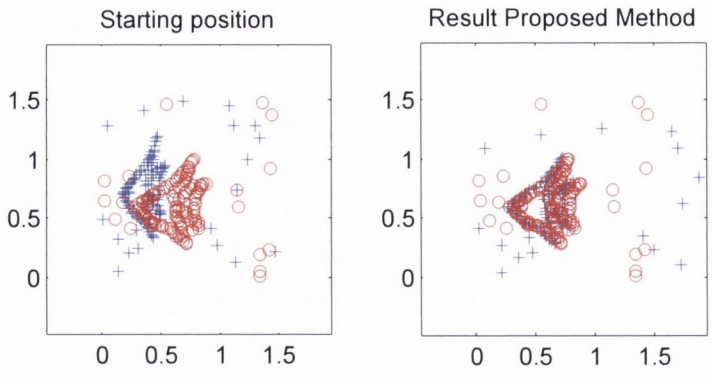


Figure C.13: Example scaling with outliers in the data set

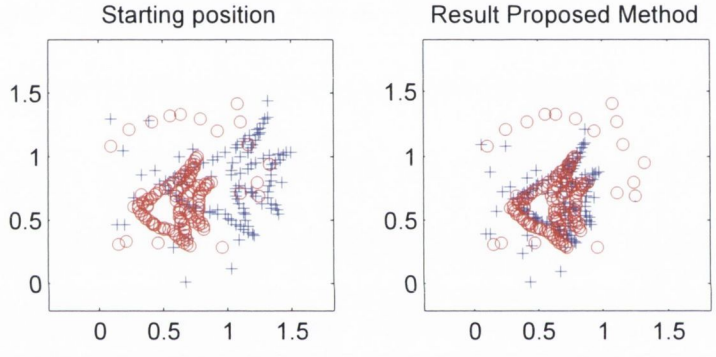


Figure C.14: Example 1 scaling with outliers in the data set

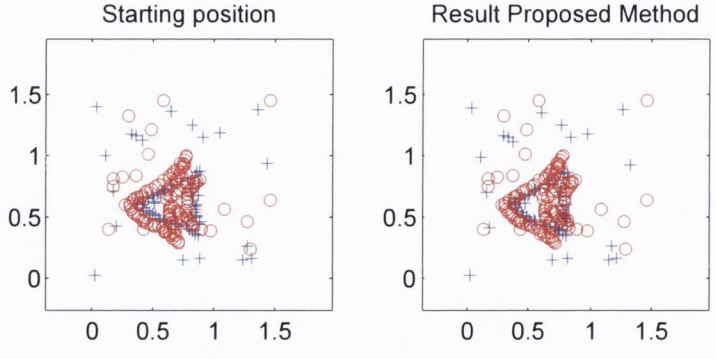


Figure C.15: Example 2 scaling with occlusion in the data set

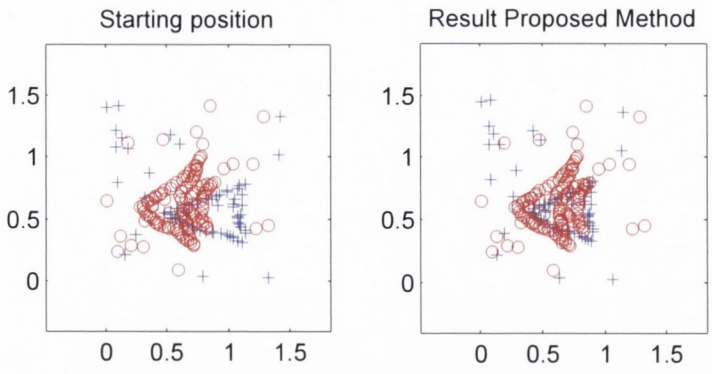


Figure C.16: Example 3 scaling with occlusion in the data set

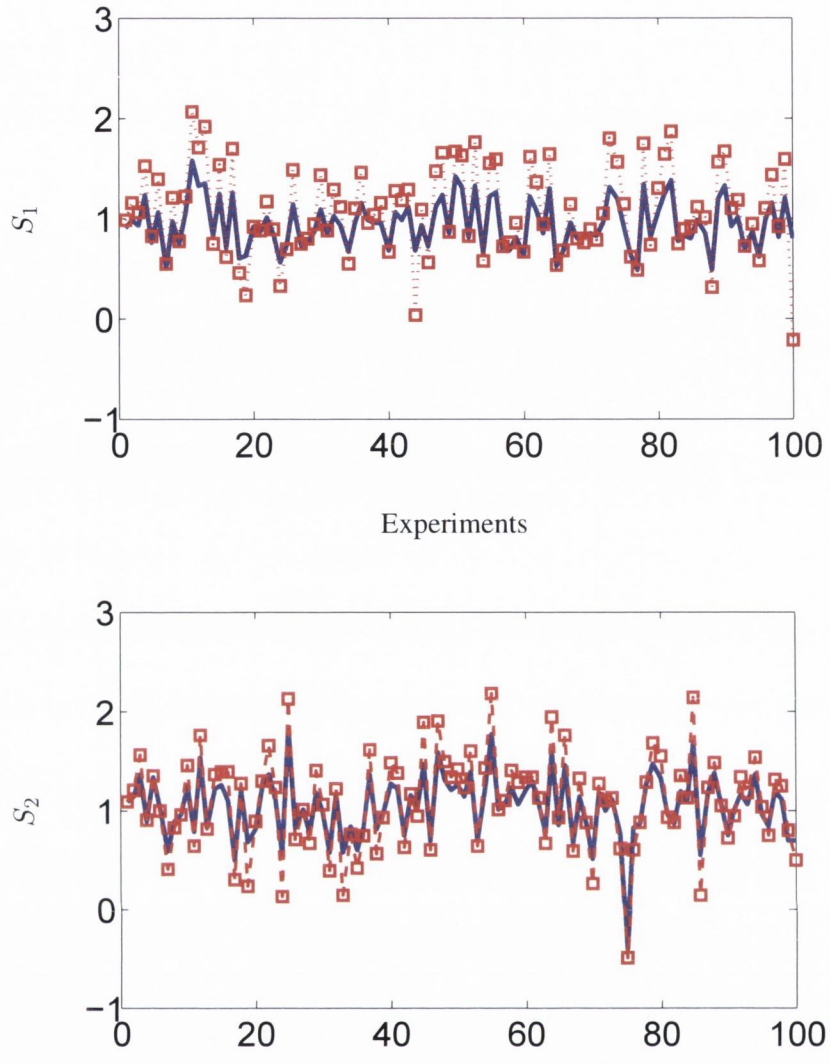


Figure C.17: Top: Results for s_1, estimated (blue line) and ground truth(red square). Bottom shows the results for s_2. The experiments where performed using the occluded data set which was scaled using a normal distribution for both parameters.

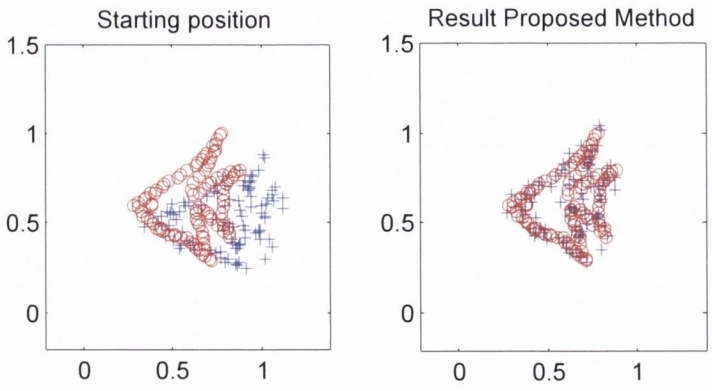


Figure C.18: Example 4 scaling with noisy data set

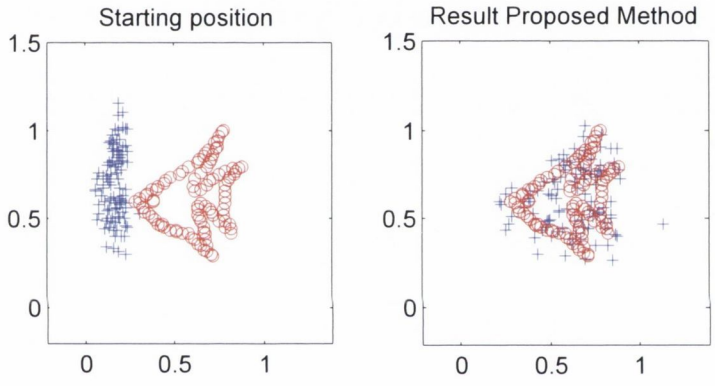


Figure C.19: Example 5 scaling with noisy data set

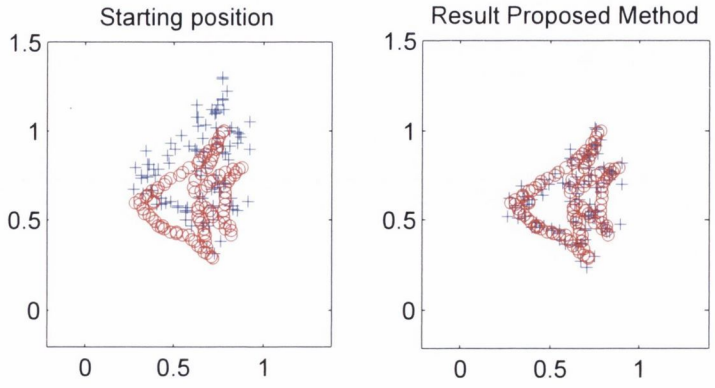


Figure C.20: Example 6 scaling with noisy data set

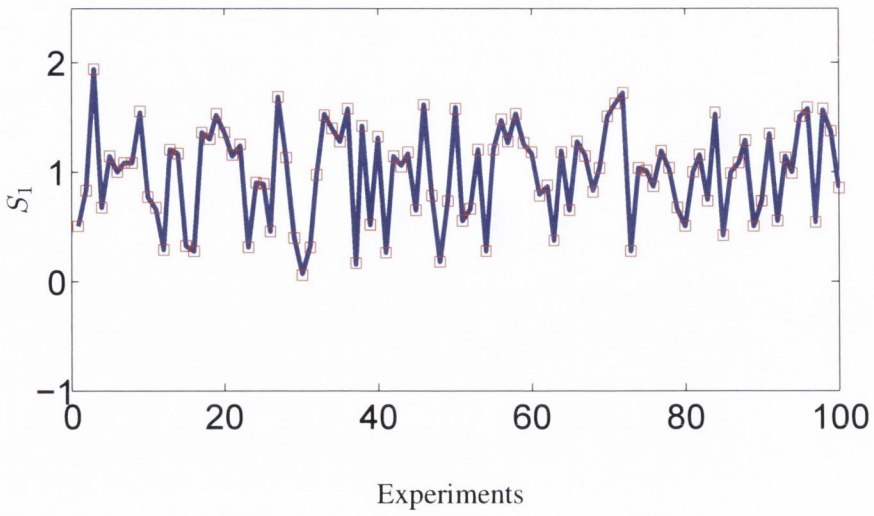


Figure C.21: Example 7 scaling with noisy data set (S_1)

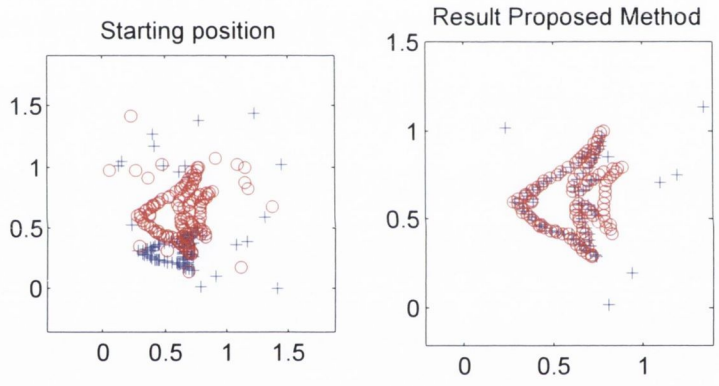


Figure C.23: Example 1 Occlusion (25%) and Outliers (20%) Examples (using prior)

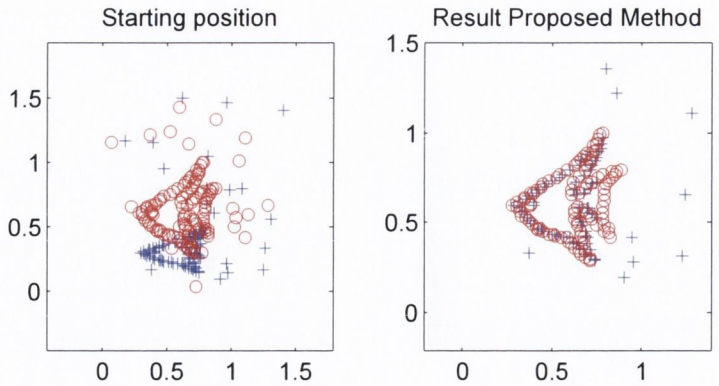


Figure C.24: Example 2 Occlusion (25%) and Outliers (20%) Examples (using prior)

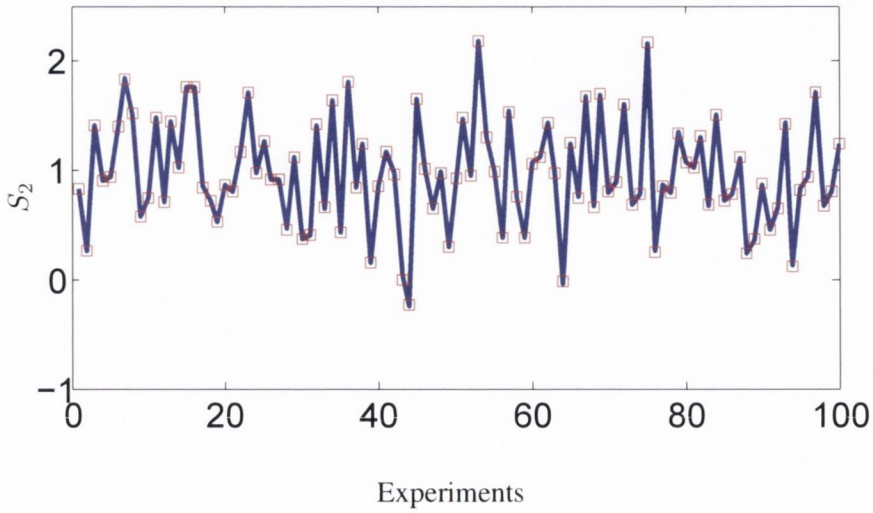
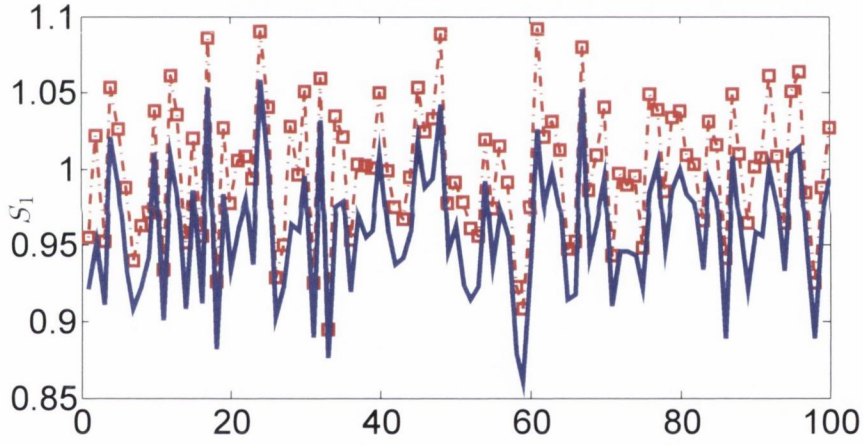


Figure C.22: Example 8 scaling with noisy data set (S_2)



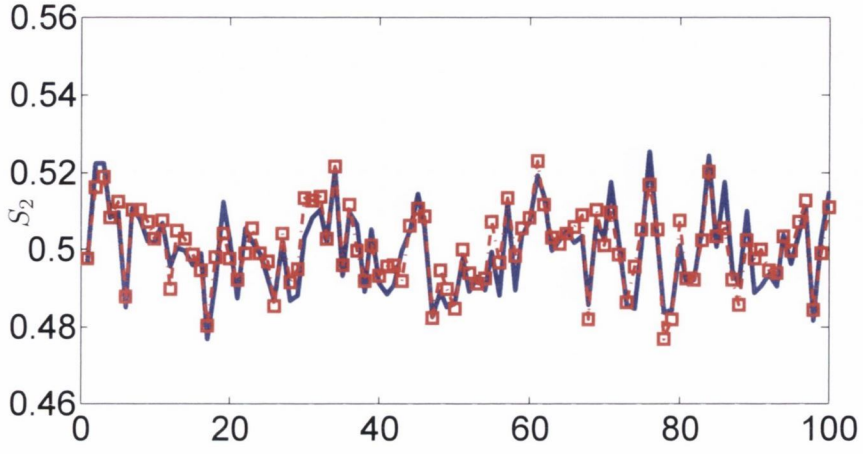


Figure C.25: Example 3 Occlusion (25%) and Outliers (20%) Results for S_1 and S_2 (with prior)

C.2 Ellipse fitting

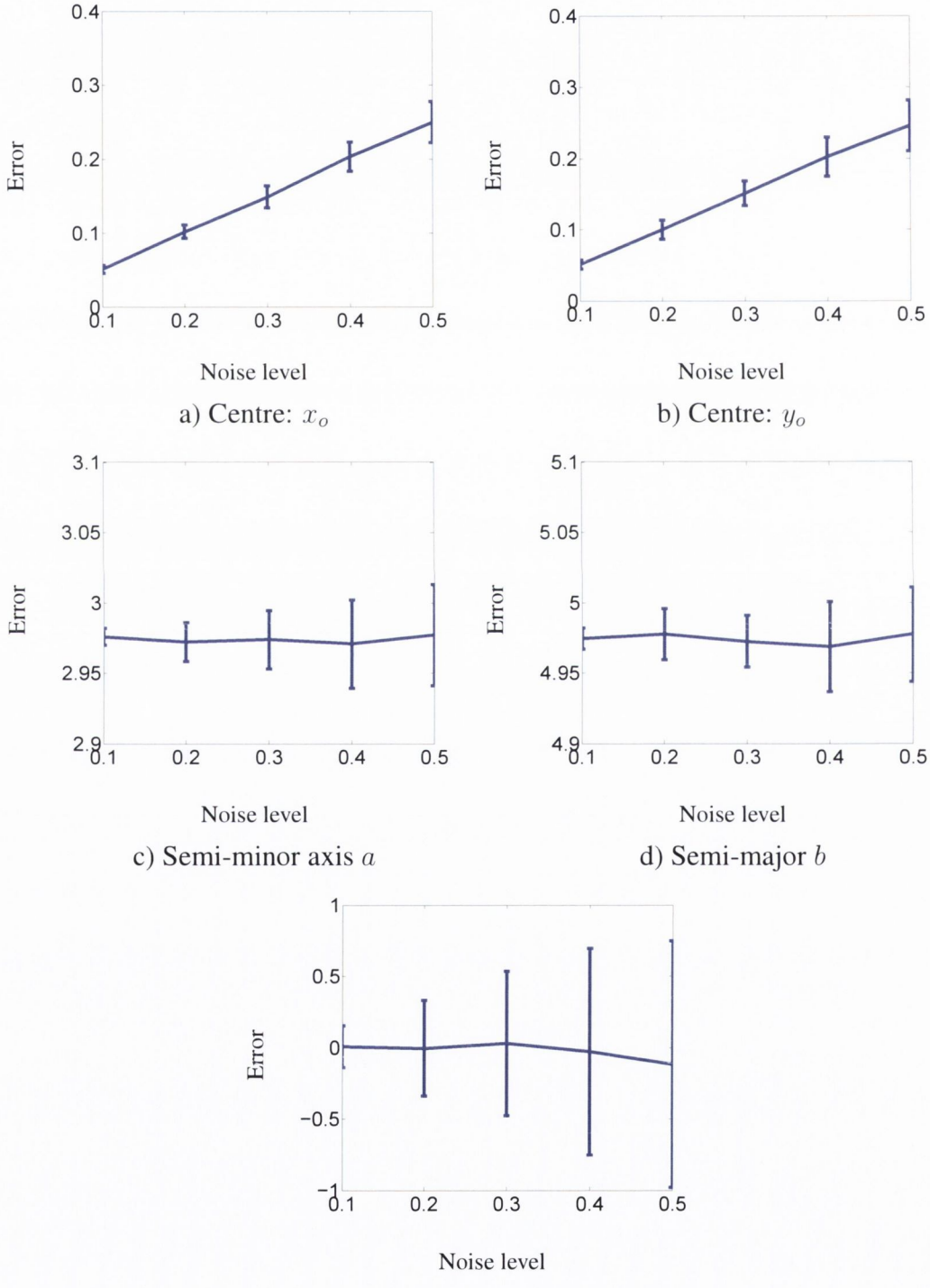


Figure C.26: Results of the parameters estimated using our proposed method.

C.3 Shape fitting

The performance of the algorithm is assessed when reducing the number of kernels in the shape models. Figure C.3 shows the estimated shapes using the isotropic and non-isotropic shape models with 71, 54 and 43 kernels. its solution is closer to the observations. As the number of kernels decrease, the isotropic shape model loses robustness.

We compute the Euclidean distance between the estimated shapes and the observations as a quantitative measure for similarity in between the shapes. Results are reported in figure C.28.

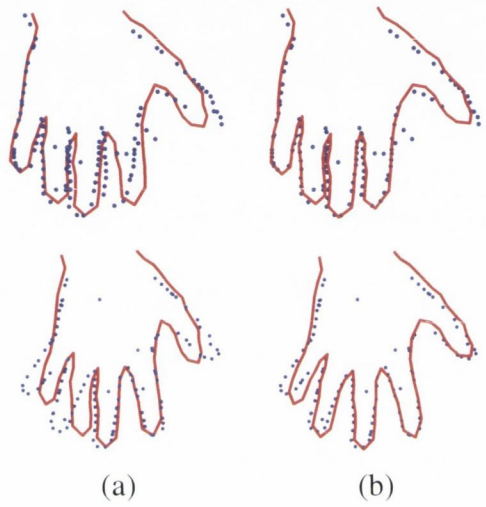


Figure C.27: Estimated shapes (in red solid line) using the isotropic shape model (a) and the non-isotropic shape model (b). The observations are shown as blue dots.

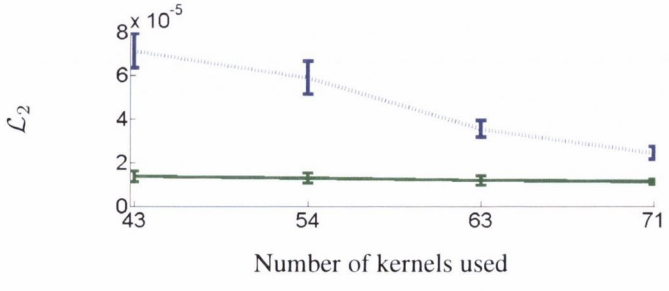
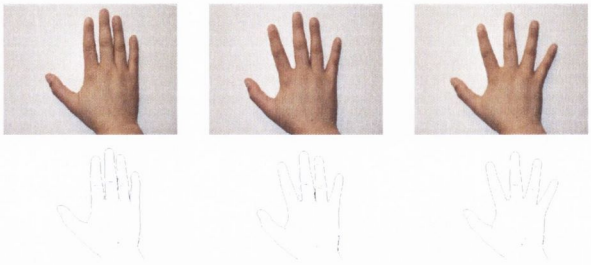
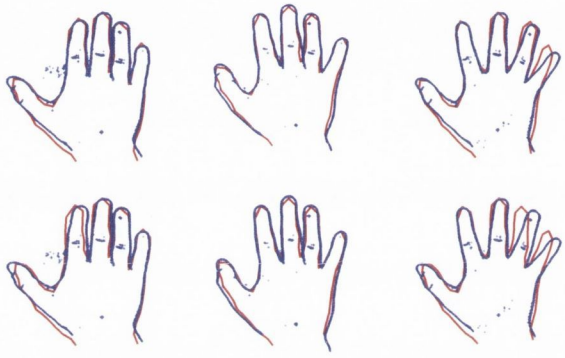


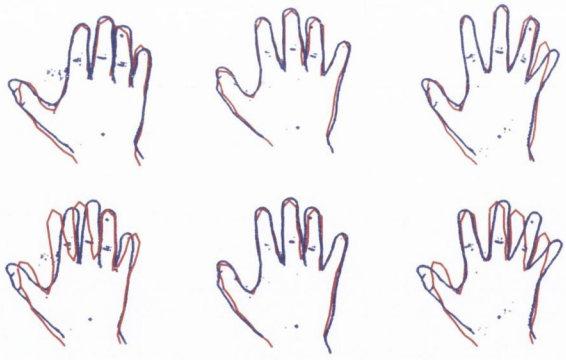
Figure C.28: The green line correspond to the non-isotropic model and the blue dots when using the isotropic model. The abscissa corresponds to number of kernels used in the model.



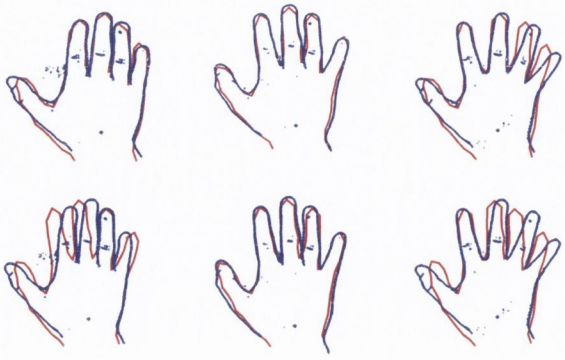
71 kernels



54 kernels



43 kernels



Bibliography

[1] R. Blake and R. Sekuler, *Perception*. Boston: mcGraw-Hill, 2006.

[2] T. Pavlidis, "Algorithms for graphics and image processing," *Computer Science Press, Rockville*, no. 1, p. 143, 1982.

[3] N. Stefanoski and J. Ostermann, "Spatially and temporally scalable compression of animated 3d meshes with mpeg-4/famc," in *IEEE International Conference on Image Processing*, 2008.

[4] J. J. Verbeek, N. Vlassis, and B. J. A. Kröse, "Self-organizing mixture models," *Neurocomput.*, vol. 63, pp. 99–123, 2005.

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

[6] M. D. Levine and Y. (Chris) Yu, "State-of-the-art of 3d facial reconstruction methods for face recognition based on a single 2d training image per person," *Pattern Recogn. Lett.*, vol. 30, no. 10, pp. 908–913, 2009.

[7] J. Yu, S. Kulkarni, and H. Poor, "Robust fitting of ellipses and spheroids," in *Signals, Systems and Computers*, pp. 94–98, 2009.

[8] R. Halır and J. Flusser, "Numerically stable direct least squares fitting of ellipses," in *Proc. 6th International Conference in Central Europe on Computer Graphics and Visualization. WSCG*, vol. 98, pp. 125–132, Citeseer, 1998.

[9] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 476–480, 1999.

[10] A. Chia, S. Rahardja, D. Rajan, and K. Leung, "A split and merge based ellipse detector with self-correcting capability," *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1991–2006, 2011.

[11] F. Mai, Y. Hung, H. Zhong, and W. Sze, "A hierarchical approach for fast and robust ellipse extraction," *Pattern Recognition*, vol. 41, no. 8, pp. 2512 – 2524, 2008.

[12] E. Kim, M. Haseyama, and H. Kitajima, "Fast and robust ellipse extraction from complicated images," in *Proceedings of the IEEE International Conference on Information Technology and Applications*, 2002.

[13] R. A. McLaughlin, "Randomized hough transform: Improved ellipse detection with comparison," *Pattern Recognition Letters*, vol. 19, no. 34, pp. 299 – 305, 1998.

[14] H. Yuen, J. Illingworth, and J. Kittler, "Detecting partially occluded ellipses using the hough transform," *Image and Vision Computing*, vol. 7, no. 1, pp. 31 – 37, 1989.

[15] I. Dryden and K. Mardia, "statistical shape analysis," *Lecture Notes in Computer Science*, vol. 7576, 1998.

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

[17] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[18] R. Chellappa and R. Bagdazian, "Fourier coding of image boundaries," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 102–105, 1984.

[19] H. Kauppinen, T. Seppnen, and M. Pietikinen, "An experimental comparison of autoregressive and fourier-based descriptors in 2d shape classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 201–207, 1995.

[20] E. Persoon and K. S. Fu, "Shape discrimination using fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 3, pp. 388–397, 1986.

[21] Q. M. Tieng and W. W. Boles, "Recognition of 2d object contours using the wavelet transform zero-crossing representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 910–916, Aug. 1997.

[22] K. Kith and E. h. Zahzah, "2d shape recognition using discrete wavelet descriptor under similitude transform," in *Proceedings of the 10th international conference on Combinatorial Image Analysis*, pp. 679–689, 2004.

[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[24] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2007.

[25] A. G. Bors and N. Nasios, "Kernel bandwidth estimation for nonparametric modeling," *Transactions on Systems, Man, and Cybernetics–Part B*, vol. 39, no. 6, pp. 1543–1555, 2009.

[26] D. Comaniciu, "An algorithm for data-driven bandwidth selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, vol. 25, no. 2, pp. 281–288, 2003.

[27] D. W. Scott and S. R. Sain, "Multidimensional density estimation," in *Data Mining and Data Visualization* (E. W. C.R. Rao and J. Solka, eds.), vol. 24 of *Handbook of Statistics*, pp. 229 – 261, Elsevier, 2005.

[28] T. Kohonen, M. R. Schroeder, and T. S. Huang, eds., *Self-Organizing Maps*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 3rd ed., 2001.

[29] H. Li and R. Hartley, "A new and compact algorithm for simultaneously matching and estimation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. iii–5, 2004.

[30] V. Mani and D. rivazhagan, "Survey of medical image registration," *Journal of Biomedical Engineering and Technology*, vol. 1, no. 2, pp. 8–25, 2013.

[31] J. Ho, M. H. Yang, A. Rangarajan, and B. Vemuri, "A new affine registration algorithm for matching 2d point sets," in *IEEE Workshop on Applications of Computer Vision.*, pp. 25–25, 2007.

[32] X. Huang, N. Paragios, and D. N. Metaxas, "Shape registration in implicit spaces using information theory and free form deformations," *Transactions on Pattern Analysis and Machine Intelligence.*, vol. 28, no. 8, pp. 1303–1318, 2006.

[33] B. J. Brown and S. Rusinkiewicz, "Global non-rigid alignment of 3-d scans," *ACM Transactions on Graphics*, vol. 26, no. 3, 2007.

[34] J. Qu, L. Gong, and L. Yang, "A 3d point matching algorithm for affine registration," *International Journal of Computer Assisted Radiology and Surgery*, vol. 6, no. 2, pp. 229–236, 2011.

[35] Y. Gao, Y. Rathi, S. Bouix, and A. Tannenbaum, "Filtering in the diffeomorphism group and the registration of point sets.," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4383–4396, 2012.

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

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

[38] Y. Tsin and T. Kanade, "A correlation-based approach to robust point set registration," in *European Conference in Computer Vision ECCV*, pp. 558–569,2004.

[39] B. Jian and B. Vemuri, "Robust point set registration using gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, 2011.

[40] H. Chui and A. Rangarajan, "A feature registration framework using mixture models," in *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, pp. 190 –197, 2000.

[41] 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.

[42] S. Granger and X. Pennec, "Multi-scale em-icp: A fast and robust approach for surface registration," *European Conference on Computer Vision ECCV*, pp. 418–432, 2002.

[43] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," *Third International Conference on 3-D Digital Imaging and Modeling*, pp. 145–152, 2001.

[44] A. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

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

[46] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm," *Image and Vision Computing*, vol. 23, no. 3, pp. 299 – 309, 2005.

[47] G. Sharp, S. Lee, and D. Wehe, "Maximum-likelihood registration of range images with missing data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 1, pp. 120–130, 2008.

[48] T. Weise, S. Bouaziz, H. Li, and M. Pauly, "Realtime performance-based facial animation," *ACM Transactions Graphics*, vol. 30, pp. 1–10, 2011.

[49] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid icp algorithms for surface registration," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1–8, 2007.

[50] P. Manojkumar and G. Reddy, "Parallel implementation of 3d modelling of indoor environment using microsoft kinect sensor," in *Fourth International Conference on Computing, Communications and Networking Technologies ICCCNT*, pp. 1–6, July 2013.

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

[52] W. Zhang and J. Kosecka, "Generalized ransac framework for relaxed correspondence problems," in *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission 3DPVT*, pp. 854–860, 2006.

[53] S. Gold, A. Rangarajan, C. P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2d and 3d point matching: pose estimation and correspondence," *Pattern Recognition*, vol. 31, no. 8, pp. 1019–1031, 1998.

[54] A. Rangarajan, E. Mjolsness, S. Pappu, L. Davachi, P. Goldman-Rakic, and J. Duncan, "A robust point matching algorithm for autoradiograph alignment," in *Visualization in Biomedical Computing (Lecture Notes in Computer Science)*, vol. 1131, pp. 277–286, 1996.

[55] B. Luo and E. Hancock, "Iterative procrustes alignment with the {EM} algorithm," *Image and Vision Computing*, vol. 20, no. 56, pp. 377 – 396, 2002.

[56] G. McNeill and S. Vijayakumar, "A probabilistic approach to robust shape matching," in *Image Processing, 2006 IEEE International Conference on*, pp. 937–940, 2006.

[57] R. Horaud, F. Forbes, M. Yguel, G. Dewaele, and J. Zhang, "Rigid and articulated point registration with expectation conditional maximization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 587–602, 2011.

[58] B. Luo and E. R. Hancock, "A unified framework for alignment and correspondence.," *Computer Vision and Image Understanding*, vol. 92, no. 1, pp. 26–55, 2003.

[59] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 1, pp. 300–307, 2007.

[60] A. Basu, I. Harris, N. Hjort, and M. Jones, "Robust and efficient estimation by minimising a density power divergence," *Biometrika*, vol. 85, no. 3, pp. 549–559, 1998.

[61] J. Glaunes, A. Trouve, and L. Younes, "Diffeomorphic matching of distributions: a new approach for unlabelled point-sets and sub-manifolds matching," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, vol. 2, pp. II–712–II–718, 2004.

[62] F. Wang, B. Vemuri, and T. Syeda-Mahmood, "Generalized l2-divergence and its application to shape alignment," in *Information Processing in Medical Imaging (Lecture Notes in Computer Science)*, vol. 5636, pp. 227–238, 2009.

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

[64] F. Wang, B. Vemuri, and A. Rangarajan, "Groupwise point pattern registration using a novel cdf-based jensen-shannon divergence," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, vol. 1, pp. 1283–1288, 2006.

[65] M. Liu, B. Vemuri, S. Amari, and F. Nielsen, "Total bregman divergence and its applications to shape retrieval," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 3463–3468, 2010.

[66] T. Chen, B. Vemuri, A. Rangarajan, and S. Eisenschenk, "Group-wise point-set registration using a novel cdf-based havrda-charvt divergence," *International Journal of Computer Vision*, vol. 86, no. 1, pp. 111–124, 2010.

[67] T. Jebara, R. Kondor, and A. Howard, "Probability product kernels," *Journal of Machine Learning Research*, vol. volume 5, pp. 819–844, 2004.

[68] A. Roy, A. Gopinath, and A. Rangarajan, "Deformable density matching for 3d non-rigid registration of shapes," in *Medical Image Computing and Computer-Assisted Intervention, MICCAI*, vol. 4791 of *Lecture Notes in Computer Science*, pp. 942–949, 2007.

[69] M. Zollhöfer, M. Martinek, G. Greiner, M. Stamminger, and J. Süßmuth, "Automatic reconstruction of personalized avatars from 3d face scans," in *Proceedings of Computer Animation and Virtual Worlds, CASA*, vol. 22, pp. 195–202, 2011.

[70] S. Romdhani, V. Blanz, and T. Vetter, "Face identification by fitting a 3d morphable model using linear shape and texture error functions," *European Conference on Computer Vision, ECCV*, pp. 3–19, 2002.

[71] E. Elyan and H. Ugail, "Reconstruction of 3d human facial images using partial differential equations.," *Computers*, vol. volume 2, no. 8, pp. 1–8, 2007.

[72] F. Tsalakanidou, F. Forster, S. Malassiotis, and M. G. Strintzis, "Real-time acquisition of depth and color images using structured light and its application to 3d face recognition," *Real-Time Imaging*, vol. volume 11, no. 5-6, pp. 358 – 369, 2005. (Special Issue on Multi-Dimensional Image Processing).

[73] L. Shi, X. Yang, and H. Pan, "3d face visualization using grid light," *Computing in Science Engineering*, vol. volume 10, no. 2, pp. 48–54, 2008.

[74] V. De Wansa Wickramarante, V. Ryazanov, and A. Vinogradov, "Accurate reconstruction of 3d model of a human face using structured light," *Pattern Recognition and Image Analysis*, vol. 18, pp. 442–446, 2008.

[75] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," *Proceedings of the annual conference on Computer Graphics and Interactive Techniques, SIGGRAPH*, pp. 187–194, 1999.

[76] P. Fua, "Regularized bundle-adjustment to model heads from image sequences without calibration data," *International Journal of Computer Vision*, vol. 38, pp. 153–171, 2000.

[77] J. Lee, B. Moghaddam, H. Pfister, and R. Machiraju, "Silhouette-based 3d face shape recovery," in *Graphics Interface*, pp. 21–30, 2003.

[78] W. A. Smith and E. R. Hancock, "Recovering face shape and reflectance properties from single images," in *IEEE International Conference on Automatic Face Gesture Recognition*, pp. 1–8, 2008.

[79] R. Lengagne, P. Fua, and O. Monga, "3d stereo reconstruction of human faces driven by differential constraints," *Image and Vision Computing*, vol. volume 18, no. 4, pp. 337–343, 2000.

[80] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer, "High resolution passive facial performance capture," *ACM Transactions Graphics*, vol. 29, no. 4, pp. 1–10, 2010.

[81] A. Patel and W. A. Smith, "3d morphable face models revisited," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1327 –1334, 2009.

[82] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3d face model for pose and illumination invariant face recognition," *IEEE Conference on Advanced Video and Signal Based Surveillance*, pp. 296–301, 2009.

[83] C. Basso, T. Vetter, and V. Blanz, "Regularized 3d morphable models," in *IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, pp. 3–12, 2003.

[84] S. Romdhani and T. Vetter, "Efficient, robust and accurate fitting of a 3d morphable model," in *Proceedings of the IEEE International Conference on Computer Vision, ICCV*, pp. 59–66, 2003.

[85] S. Baker and I. Matthews, "Equivalence and efficiency of image alignment algorithms," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1090–1097, 2001.

[86] S. Romdhani and T. Vetter, "Estimating 3d shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 986–993, 2005.

[87] F. Romeiro and T. Zickler, "Model-based stereo with occlusions," in *Proceedings of the 3rd international conference on Analysis and modeling of faces and gestures, AMFG*, pp. 31–45, 2007.

[88] B. Amberg, A. Blake, A. Fitzgibbon, S. Romdhani, and T. Vetter, "Reconstructing high quality face-surfaces using model based stereo," *IEEE International Conference on Computer Vision, ICCV*, pp. 1–8, 2007.

[89] M. Zhao, T. S. Chua, and T. Sim, "Morphable face reconstruction with multiple images," in *IEEE International Conference on Automatic Face and Gesture Recognition, FGR*, pp. 597–602, 2006.

[90] X. Wang, W. Liang, and L. Zhang, "Morphable face reconstruction with multiple views," *International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC*, pp. 250–253, 2010.

[91] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth mapping using projected patterns," *Patent Publication number: US 2010/0118123 A1*, 2008.

[92] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgbd mapping: Using depth cameras for dense 3d modeling of indoor environments," in *In RGB-D: Advanced Reasoning with Depth Cameras Workshop in conjunction with RSS*, 2010.

[93] B. Huhle, T. Schairer, P. Jenke, and W. Straer, "Fusion of range and color images for denoising and resolution enhancement with a non-local filter," *Computer Vision and Image Understanding*, vol. 114, no. 12, pp. 1336 – 1345, 2010.

[94] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J. Frahm, R. Yang, D. Nister, and M. Pollefeys, "Real-time visibility-based fusion of depth maps," in *IEEE 11th International Conference on Computer Vision, ICCV*, pp. 1–8, 2007.

[95] R. Newcombe and A. Davison, "Live dense reconstruction with a single moving camera," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1498 –1505, 2010.

[96] K. Ouji, M. Ardabilian, L. Chen, and F. Ghorbel, "A space-time depth super-resolution scheme for 3d face scanning," in *Proceedings of the 13th international conference on Advanced concepts for intelligent vision systems*, ACIVS, pp. 658–668, 2011.

[97] Y. Cui, S. Schuon, C. Derek, S. Thrun, and C. Theobalt, "3d shape scanning with a time-of-flight camera," in *IEEE conference on Computer Vision and Patern Recognition, CVPR*, pp. 1173–1180, 2010.

[98] J. Ruttle, C. Arellano, and R. Dahyot, "Extrinsic camera parameters estimation for shape-from-depths," in *20th European Signal Processing Conference (Eusipco)*, pp. 1985–1989, 2012.

[99] J. Ruttle, *Statistical Framework for Multi-Sensor Fusion and 3D Reconstruction*. PhD thesis, School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland, 2012.

[100] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR, pp. 127–136, 2011.

[101] H. M, J. Choi, and G. Medioni, "Laser scan quality 3-d face modelling using a low cost depth map," in *20th European Signal Processing Conference (Eusipco)*, pp. 1995–1999, 2012.

[102] D. Schneider and P. Eisert, "Algorithms for automatic and robust registration of 3d head scans," *Journal of Virtual Reality and Broadcasting*, vol. 7, 2010.

[103] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, pp. 38 – 59, 1995.

[104] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 484–498, 1998.

[105] R. Davies, C. Twining, T. Cootes, J. Waterton, and C. Taylor, "A minimum description length approach to statistical shape modeling," *IEEE Transactions on Medical Imaging.*, vol. 21, no. 5, pp. 525–537, 2002.

[106] R. H. Davies, C. Twining, P. D. Allen, T. F. Cootes, and C. Taylor, "Building optimal 2d statistical shape models," *Image and Vision Computing*, vol. 21, pp. 200–3, 2003.

[107] T. F. Cootes, C. J. Twining, and C. J. Taylor, "Diffeomorphic statistical shape models," in *Proceedings of the British Machine Vision Conference, BMVC*, pp. 447–456, 2004.

[108] A. Lanitis, C. J. Taylor, and T. F. Cootes, "Automatic interpretation and coding of face images using flexible models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 743–756, 1997.

[109] M. Roberts, T. Cootes, E. Pacheco, and J. Adams, "Quantitative vertebral fracture detection on {DXA} images using shape and appearance models," *Academic Radiology*, vol. 14, no. 10, pp. 1166 – 1178, 2007.

[110] P. Tresadern, M. Ionita, and T. Cootes, "Real-time facial feature tracking on a mobile device," *International Journal of Computer Vision*, vol. 96, no. 3, pp. 280–289, 2012.

[111] X. Bai, C. Sun, and F. Zhou, "Splitting touching cells based on concave points and ellipse fitting," *Pattern Recogn.*, vol. 42, no. 11, pp. 2434–2446, 2009.

[112] Y. Soh, J. Bae, D. Kim, and H. Kim, "A new method for ellipse fitting in satellite images," in *Second International Conference on Intelligent Computation Technology and Automation, ICICTA.*, vol. 1, pp. 502–506, 2009.

[113] S. Mahdavi, W. Morris, I. Spadinger, N. Chng, O. Goksel, and S. Salcudean, "3d prostate segmentation in ultrasound images based on tapered and deformed ellipsoids," in *Medical Image Computing and Computer-Assisted Intervention, MICCAI*, vol. 5762 of *Lecture Notes in Computer Science*, pp. 960–967, 2009.

[114] L. Ding and A. M. Martinez, "Features versus context: An approach for precise and detailed detection and delineation of faces and facial features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 11, pp. 2022–2038, 2010.

[115] D. Maio and D. Maltoni, "Real time face location on gray-scale static images," *Pattern Recognition*, vol. 33, no. 9, pp. 1525–1540, 2000.

[116] C. Wong, S. C. F. Lin, T. R. Ren, and N. M. Kwok, "A survey on ellipse detection methods," in *IEEE International Symposium on Industrial Electronics, ISIE*, pp. 1105–1110, 2012.

[117] D. Chaudhuri, "A simple least squares method for fitting of ellipses and circles depends on border points of a two-tone image and their 3-d extensions," *Pattern Recogn. Lett.*, vol. 31, no. 9, pp. 818–829, 2010.

[118] W. Gander, G. Golub, and R. Strebel, "Least-squares fitting of circles and ellipses," *BIT Numerical Mathematics*, vol. 34, no. 4, pp. 558–578, 1994.

[119] P. L. Rosin, "A note on the least squares fitting of ellipses," *Pattern Recogn. Lett.*, vol. 14, no. 10, pp. 799–808, 1993.

[120] J. Cabrera and P. Meer, "Unbiased estimation of ellipses by bootstrapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, pp. 752–756, 1996.

[121] W. Chojnacki, M. J. Brooks, A. V. D. Hengel, and D. Gawley, "Fitting surfaces to data with covariances," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1294–1303.

[122] K. Kanatani, "Statistical bias of conic fitting and renormalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, vol. 16, no. 3, pp. 320–326, 1994.

[123] Z. Szpak, W. Chojnacki, and A. Hengel, "Guaranteed ellipse fitting with the sampson distance," in *Computer Vision, ECCV*, vol. 7576 of *Lecture Notes in Computer Science*, pp. 87–100, 2012.

[124] K. Kanatani and P. Rangarajan, "Hyper least squares fitting of circles and ellipses," *Comput. Stat. Data Anal.*, vol. 55, no. 6, pp. 2197–2208, 2011.

[125] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[126] P. Hart, "How the hough transform was invented," *Signal Processing Magazine, IEEE*, vol. 26, no. 6, pp. 18–22, 2009.

[127] D. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111 – 122, 1981.

[128] P. Nair and A. S. Jr., "Hough transform based ellipse detection algorithm," *Pattern Recognition Letters*, vol. 17, no. 7, pp. 777 – 784, 1996.

[129] C. F. Chien, Y. C. Cheng, and T. T. Lin, "Robust ellipse detection based on hierarchical image pyramid and hough transform," *The Journal of the Optical Society of America*, vol. 28, no. 4, pp. 581–589, 2011.

[130] N. Guil and E. Zapata, "Lower order circle and ellipse hough transform," *J. Pattern Recognition*, vol. 30, pp. 1729–1744, 1997.

[131] S. C. Zhang and Z. Q. Liu, "A robust, real-time ellipse detector," *Pattern Recognition*, vol. 38, no. 2, pp. 273 – 287, 2005.

[132] N. Kiryati, Y. Eldar, and A. Bruckstein, "A probabilistic hough transform," *Pattern Recognition*, vol. 24, no. 4, pp. 303 – 316, 1991.

[133] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: Randomized hough transform (rht)," *Pattern Recognition Letters*, vol. 11, no. 5, pp. 331 – 338, 1990.

[134] T. C. Chen and K. L. Chung, "An efficient randomized algorithm for detecting circles," *Computer Vision and Image Understanding*, vol. 83, no. 2, pp. 172 – 191, 2001.

[135] W. Lu and J. Tan, "Detection of incomplete ellipse in images with strong noise by iterative randomized hough transform (irht)," *Pattern Recognition*, vol. 41, no. 4, pp. 1268 – 1279, 2008.

[136] D. K. Prasad, M. K. Leung, and S.-Y. Cho, "Edge curvature and convexity based ellipse detection method," *Pattern Recognition*, vol. 45, no. 9, pp. 3204 – 3221, 2012.

[137] M. C. Jones, N. L. Hjort, I. R. Harris, and A. Basu, "A comparison of related densitybased minimum divergence estimators," *Biometrika*, vol. 88, no. 3, pp. 865–873, 2001.

[138] M. Broniatowski, "Minimum divergence estimators, maximum likelihood and exponential families," *ArXiv e-prints*, Aug. 2011.

[139] S. F. Juarez and W. R. Schucany, "A note on the asymptotic distribution of the minimum density power divergence estimator," *Lecture Notes-Monograph Series*, vol. 49, pp. 334–339, 2006.

[140] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[141] D. Scott, "Parametric statistical modeling by minimum integrated square error," *Technometrics*, vol. 43, pp. 274–285, 2001.

[142] L. Ding and A. Goshtasby, "On the canny edge detector," *Pattern Recognition*, vol. 34, pp. 721–725, 2001.

[143] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[144] M. Helén and T. Virtanen, "Audio query by example using similarity measures between probability density functions of features," *EURASIP Journal on Audio, Speech, and Music Processing*, 2010.

[145] C. Shen, M. Brooks, and A. van den Hengel, "Fast global kernel density mode seeking: Applications to localization and tracking," in *IEEE Transactions on Image Processing*, vol. 16, pp. 1457–1469, 2007.