Towards Fully Automatic Optimal Shape Modeling

Karlsson, Johan

2008

Towards Fully Automatic Optimal Shape Modeling

Johan Karlsson

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118
SE-221 00 Lund
Sweden

http://www.maths.lth.se/

Doctoral Theses in Mathematical Sciences 2008:9
ISSN 1404-0034

ISBN 978-91-628-7226-7
LUTFMA-1033-2008

Printed in SWEDEN by MediaTryck, Lund 2008

Preface

When I finished high school (gymnasium), I said I would never again do anything concerned with natural sciences or technology, it was just too boring. I kept my word for seven years, during which I worked as a cinema projectionist and got a B.A. (fil.kand.) in philosophy. I loved it. Then I realized I had to get a job, and since philosophers were not in high demand at the time, I sat down and considered the alternatives. I made a list of educations I could get and the jobs they would lead to and I wrote them down in order of priority based on what I thought I would endure best. Then I applied. At the top of the list was M.D. but my grades were only almost good enough so I didn't get in. In second place I had put down computer engineer and there I got in. I figured I would just barely make it through and then I would get a job. Not a dream job, but it would pay the rent. I didn't just barely make it through, it went very well. Little by little something happened that surprised me more that anyone. I started to enjoy mathematics. When we got to choose courses I directed my education towards mathematics and when I took my first course in mathematical image analysis for Kalle Åström I felt that this was it, this was what I wanted to do. So I took some more courses and then I decided to do my masters thesis in the field. I was still interested in medicine so I talked to Kalle about it and it turned out that he had a Ph.D.-student working with medical image analysis. That was Anders Ericsson and he and Kalle supervised my masters thesis "Automatic positioning of landmarks for shape analysis". That went very well too and things got even more interesting and enjoyable so I started talking to Kalle about doing a Ph.D. and now here I am. Life is funny.

The work of this thesis makes contributions to the field of automatic optimal shape modeling. The reason that the title is "Towards fully automatic optimal shape modeling" instead of simply "Fully automatic optimal shape modeling" is that automaticity and optimality are important properties that I think can never be fully reached, because we can always redefine what we mean by automatic and optimal. We develop our tools in this direction, getting better and better tools. This thesis is part of this eternal development of the tools.

The contents of the thesis is based on the following papers,

Main papers

I J. Karlsson and K. Åström, "Generative Models of Shape and Local Appearance from Unlabeled Images with Occlusions using MDL", *Submitted to IJCV (International Journal of Computer Vision)*.

II J. Karlsson and K. Åström, "MDL Patch Correspondences on Unlabeled Images", *ICPR (International Conference on Pattern Recognition)*, 2008.

III J. Karlsson and K. Åström, "MDL Patch Correspondences on Unlabeled Images with Occlusions", *NORDIA - Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment, CVPR (IEEE Computer Society Conference on Computer Vision and Pattern Recognition)*, 2008.

IV K. Åström, J. Karlsson, O. Enqvist, A. Ericsson and F. Kahl, "Automatic Feature Point Correspondences and Shape Analysis with Missing Data and Outliers", *SCIA (Scandinavian Conference on Image Analysis)*, 2007.

V A. Ericsson and J. Karlsson, "Measures for Benchmarking of algorithms for automatic shape modeling", *JMIV (Journal of Mathematical Imaging and Vision)*, 2007.

VI A. Ericsson and J. Karlsson, "Benchmarking of algorithms for automatic correspondence localisation", *BMVC (British Machine Vision Conference)*, 2006.

VII J. Karlsson and A. Ericsson, "A Geodesic Ground Truth Correspondence Measure for Benchmarking", *ICPR (International Conference on Pattern Recognition)*, 2006.

VIII A. Ericsson and J. Karlsson, "Aligning Shapes by Minimising the Description Length", *SCIA (Scandinavian Conference on Image Analysis*, 2005.

IX J. Karlsson, A. Ericsson and K. Åström, "Parameterisation Invariant Statistical Shape Models", in Proc. *ICPR (International Conference on Pattern Recognition)*, 2004.

Subsidiary papers

X K.M. Henriksson, K. Wickström, N. Maltesson, A. Ericsson, J. Karlsson, F. Lindgren, K. Åström, T.F. McNeil and I. Agartz, "A pilot study of facial, cranial, and brain MRI morphometry in men with schizophrenia", *Psychiatry Research: Neuroimaging*, 2006.

XI A. Ericsson and J. Karlsson, "A Geodesic Ground Truth Correspondence Measure for Benchmarking", in Proc. *SSBA (Swedish Symposium on Image Analysis)*, 2006.

XII A. Ericsson and J. Karlsson, “Aligning Shapes by Minimising the Description Length”, in Proc. *SSBA (Swedish Symposium on Image Analysis)*, 2005.

XIII K. Josephson, A. Ericsson and J. Karlsson, “Segmentation of Medical Images Using Three-Dimensional Active Shape Models”, *SCIA (Scandinavian Conference on Image Analysis), 2005.*

XIV J. Karlsson, A. Ericsson and K. Åström, “Parameterisation Invariant Statistical Shape Models”, *SSBA (Swedish Symposium on Image Analysis)*, 2004.

XV J. Karlsson "Automatic positioning of landmarks for shape analysis", Master’s thesis, Centre for mathematical sciences, Lund University, 2003.

Acknowledgments

My primary thanks go to Anders Ericsson whom I have worked with during a lot of my research. The value of the discussions and teamwork has been immense.

Thanks also go to my supervisors. Primarily to Kalle Åström who's optimism and knowledge have been a great help. I also thank my other supervisors Gunnar Sparr and Lars Edenbrandt.

The colleagues at the department have also been great, I mention none and forget none.

Also, great thanks go to my friends and family who are a haven of support and perspective.

Research projects are not possible without money. I am deeply grateful for the financial support of this project by the Swedish Knowledge Foundation through the Industrial PhD programme in Medical Bioinformatics at Karolinska Institutet, Strategy and Development Office, in cooperation with UMAS and by the Swedish Foundation for Strategic Research (SSF) through the programme Vision in Cognitive Systems (VISCOS).

Contents

List of Figures

Chapter 1
Introduction

Strive not to be a success, but rather to be of value.

Albert Einstein

1.1 Motivation

This thesis deals with statistical shape models and with the effort of building them as automatically and as optimally as possible.

There are many areas where statistical models of shape are useful. One of the main ones is medical image analysis. Medical images are becoming more and more prominent both in medical research and in clinical practice. The standard case involves 3D images, often of high resolution, and often in several different modalities, for example MR, PET, and CT, giving information about physical anatomy as well as different kinds of functionality. These sets of images contain loads of information but analyzing them is difficult and time consuming for the medical staff. One large reason for this is that we humans can not view 3D-data. We can view 2D-slices of the 3D-image and we can view surfaces of 3D objects but these are poor substitutes. Computers do not have this problem, but by themselves they lack human capacities for pattern recognition an analysis. This is where image analysis tools like the models studied in this thesis come into play. Often a specific organ is the subject of examination and if the computer automatically detects and segments this organ and presents it visually in a manner suited for the problem at hand, along with information about its shape and appearance, and possibly a suggested diagnosis, this is very helpful.

A statistical shape model is built from a set of training examples. One of the main issues with statistical shape models is that the result depends on how well the model is built. Building them well is difficult and time consuming. Therefore, there is motivation for automation and optimization of this model building. This thesis makes contributions to this.

1.2 Organization of the Thesis

In Part I Shape Models are introduced in Chapter 2 and a complete chain of steps for building and using such models is presented in Chapter 3. An improvement to the standard approach is presented in Chapter 4. Part II goes deeper into the automatic and op-

timal construction of shape models using Minimum Description Length (MDL). First, construction of models from a training set of object outlines is described. MDL is used for establishing correspondences (Chapter 5) and for aligning the shapes (Chapter 6). In Chapter 7 a solution to a serious problem with the MDL-approach (formerly solved with ad-hoc techniques) is presented. Methods for evaluating the resulting correspondences and the resulting shape models is discussed in Chapter 8. Finally a method for constructing models directly from a training set of images without first segmenting object outlines is presented in Chapter 9.

Here is a chapter by chapter description of the contents.

Part I

Chapter 2 First a background of deformable models is given and then the general theory of shape and shape models is presented since this is the type of models that will be used in this thesis.

Chapter 3 As an illustration of the construction and use of shape models, a complete segmentation system using 3D-shape models for segmenting the femur in the knee in Magnetic Resonance Images (MRI) and the brain in Single Photon Emission Computed Tomography (SPECT) images is presented. The chapter is based on Paper XIII.

Chapter 4 A modification of the segmentation algorithm using integral approximation is here shown to give better results.

Part II

Chapter 5 One of the main tools used for automatic model building is Minimum Description Length (MDL). Here this is defined and the description length of a shape model is presented. The gradient of the description length to be used in optimization is also described.

Chapter 6 When using MDL for establishing correspondences we may want to also align the shapes using MDL instead of using standard Procrustes analysis. The gradient of the description length with respect to rotation is derived. The alignment of the shapes is then optimized using the Gauss-Newton algorithm. This chapter is based on Paper VIII.

Chapter 7 When optimizing Minimum Description Length an undesired side-effect is that landmarks tend to cluster together. This has been addressed using ad hoc methods. Here novel theory to prevent clustering during correspondence optimization is presented.

The standard scalar product is replaced by a new formula that is invariant to mutual reparameterizations and this solves the problem. This chapter is based on Paper IX.

Chapter 8 As there are several different algorithms used for establishing correspondences, there is a need for comparing them to see which algorithm is best. In this chapter severe problems with the standard model quality measures compactness, generality and specificity are discussed. A ground truth correspondence measure (GCM) is proposed for benchmarking and benchmarking is performed on several algorithms. This chapter is based on Papers V-VII.

Chapter 9 In the model building in the thesis so far, it is assumed that the outlines of the objects in the training images are given. In this chapter a new model building technique that works fully automatically from unlabeled images is developed. This chapter is based on Papers I-IV.

Chapter 10 A brief summary and conclusions of the thesis.

Appendix A A description of the datasets used in most experiments.

Appendix B Notations used in the thesis presented in alphabetical order.

1.3 Main Contributions

In Chapter 4 the segmentation algorithm is improved by adding area weights to the objective function as a way of approximating an integral instead of using a sum. This improves segmentation results. This work was all done by the author of this thesis.

Apart from this contribution, all the main contributions are concerned not with using the models but with the establishing of correspondences needed for building the models. This was chosen as the main focus because if this is done poorly the rest of the chain will be of low quality and because the previously existing methods may work reasonably well most of the time in a research setting but both automaticity and optimality can definitely be improved and for applications like medical decision support demands for robustness and precision are very high and in a realistic application setting the existing methods are often not useful.

When optimizing correspondences using MDL you may want to also align shapes using MDL. This has been noted before but in Chapter 6 this is examined more closely than before and the gradient to be used in optimization is derived and the results are analyzed. This work was done in close collaboration by the author of this thesis and Ericsson.

When establishing correspondences by optimizing the parameterizations of the shapes with the standard formulation the problem is ill-posed and the global optimum is useless.

This also gives instability in practice. In Chapter 7 the standard scalar product is replaced with a new formula that makes the parameterization optimization invariant to mutual reparameterizations. This solves the problem. This work was done in close collaboration by the author of this thesis and Ericsson with input from Åström.

Several algorithms exist for correspondence optimization and they must be compared. In Chapter 8 the standard model quality measures compactness, generality and specificity are criticized and a new measure built on ground truth is presented and used to benchmark algorithms for correspondence localization. This work was also done in close collaboration by the author of this thesis and Ericsson.

So far we have started from a set of object outlines and established correspondences by optimizing the parameterizations of these outlines. But getting these outlines is manual work and we want automatic methods. Some such methods exist based on non-linear registration of the entire images of the training set, but this is not always a good approach, especially if the images are not produced in a very controlled environment. In Chapter 9 a new model building technique that works fully automatically from unlabeled images with occlusions is presented. This work was done mainly by the author of this thesis, based on work done mainly by Åström.

Part I

Shape Models

Chapter 2
Shape Model Theory

My own suspicion is that the universe is not only queerer than we suppose, but queerer than we can suppose. I suspect that there are more things in heaven and earth than are dreamed of or can be dreamed of in any philosophy.

J.B.S. Haldane

2.1 Introduction

The general motivation for model based methods is that if you have knowledge about the object you are for example trying to segment you can use that knowledge to get a better result. A model can also be used for interpretation of the shape and appearance of the object or for data compression. This is especially useful if the images are of poor quality with lots of noise and missing data. This is often the case for example in medical image analysis or in surveillance.

2.1.1 Background

The first approaches in this area were either template matching techniques, where a fixed template of the object is fitted to the image (or a processed version of the image for example for edge detection [109]), by global transformations, such as Euclidean or affine transformations, possibly allowing for some simple variation of the template like changing the scale (initial tentative discussion of local template deformation can be found in for example [99]), or techniques where some features are calculated from the image and fed to some classifier [63]. Some of these techniques can handle misalignment of the template, for example by using features that are invariant to things like translation, rotation and scale or by normalizing the data before calculating features [6, 20, 58]. Fit was measured as overlap, for example using Hamming distance [20], or similarity of autocorrelations [68] or cross-correlation between data and template [20] for template matching techniques, or determined by feeding the image-pixels to some machine learning system that typically calculates some features and does statistical tests [57, 75, 95] or runs the features through a decision-tree [143] or some other machine learning process or classifier like a hyperplane classifier [66], a Support Vector Machine (SVM) or a neural net. The template can be a typical specimen or some type of mean object learned from examples or it can be based on a mathematical model of the object. In some cases, the object always looks approximately the same and then this approach can work well but

often the appearance of the object can be different in different images and then there will be problems.

2.1.2 Deformable Models

For many applications there are significant global and local variations of the shape in different instances of objects from the same object class. Therefore different methods of including deformations of the shape in the model were developed. This thesis deals with such deformable models, where the model describes the typical appearance and the ways that the appearance can be deformed and still be a valid instance of the object.

Deformable models are typically curves or surfaces defined within an image domain. They can move and change globally and locally either by searching through parameters or under the influence of internal forces, which are defined within the curve or surface itself, and external forces, which are computed from the image data. The internal forces are designed to control model attributes such as smoothness and to ensure object class validity during deformation. The external forces are defined to move the model towards an object boundary or other desired features within an image. Deformable models can be active in the sense that they can adapt themselves to fit the given data.

Deformable models have been used for a wide range of applications including shape analysis [65], image restoration [3], segmentation [19, 21, 39] , object recognition and identification [72, 98, 112], object tracking [9, 14, 52, 81, 134], database retrieval [13, 145, 153] and medical imaging [131]. In this thesis the focus is on the building of models of shape and appearance, and specifically issues of optimality, automaticity and verification are addressed.

An early type of deformable models were introduced in 1973 by Widrow [147, 148]. He called it "the rubber mask technique", but the deformations were not free rubber sheet deformations in the style of the later popular Thin Plate Splines (TPS). Rather a specific model was built for the object including a number of parameters controlling angles and widths and such. Another approach was Fischler and Elschlager's "spring-loaded" templates [49]. A related but somewhat different problem is handled by Barrow who matches images to 3D symbolic maps by optimizing camera parameters [8].

Since then there has been a lot of research on deformable models and the modern techniques are widely used in image analysis and pattern recognition, see for example [15, 31, 53, 97, 119, 125, 133].

Deformable models can be divided into two types; free-form models and parametric models [71]. Free-form models can represent any shape as long as some general regularization constraint, such as continuity or smoothness, is satisfied, i.e. they are typically not specific for a certain object class. Well known examples are snakes and level set methods. Recently attempts have been made to include some apriori knowledge of the typical shape of the object being searched for into these models. Parametric models typically model a specific object class with its mean shape and its modes of variation. The current shape is

specified by a set of parameters controlling the modes of variation. This thesis deals with parametric models.

One advantage with these models is that a compact description of the shape can be achieved using the parameters. Another important aspect is that robustness and shape validity is more likely to be achieved with a global shape model. This is very important if parts of the object are occluded or of very poor image quality, which is often the case. Another advantage is that after locating and segmenting the object, the parameters for the model can be analyzed and conclusions about the shape can be drawn. For example, in medical imaging a classifier can attempt to diagnose a patient from the shape parameters of some organ.

The mean shape and the shape variation modes can either be specified mathematically (analytical deformable templates) or learned from training data (prototype-based deformable templates) [71].

The analytical deformable template is defined by a set of analytical curves (e.g. lines, ellipses). Deformations are typically defined for example as axes and positions of such ellipses. Widrow's early "rubber masks" [147], is one example of an analytical shape model. The work by Yuille et al. [149] is another. They built a model of the eye and mouth using circles and parabolic curves for video image segmentation. Object-specific parameters, such as the radius of the iris, are used to deform the model in order to optimize an object specific designed energy function in object specific steps (first finding the iris by looking at valley potentials and so on). Advantages of these models can be simplicity and speed of calculations. Disadvantages are for example the by hand model and objective function design, and of course not all objects are easily described by simple shapes like ellipses if the number of parameters is to be kept small.

The prototype-based deformable template is defined around a prototype template which describes the average shape of a class of objects. This prototype or mean shape and its modes of variation are learnt from training data and formulated so that a relatively small number of parameters control it. Statistics for these parameters can be calculated from training data so that good measures of the probability of different shape instances can be defined [21, 60, 61].

Worth mentioning are also physical deformable models, that attempt to incorporate knowledge about the expected variation of an object based on assumptions based on physics. Pentland and Sclaroff [103] and Nastar and Ayache [100] describe methods of building and using such shape models. These methods are often called Finite Element Methods (FEM) formulations, see also Cohen [23]. In [26] Cootes et al. developed a combination of physical vibrational modes and statistical variational modes. Warping techniques like Thin Plate Splines and Navier Splines also have underlying physical assumptions.

2.1.3 Statistical Shape Models

This thesis deals with a type of prototype-based deformable models often referred to as statistical shape models. Here follows a brief account of these statistical shape models. More details can be found for example in [28, 38, 79, 122].

Statistical shape models attempt to capture the patterns of variability found in a class of objects as defined by a training set of examples. The basic idea is to estimate the population statistics from a set of examples instead of using a single prototype. Some of the most important work in this field, and the starting point for much of the work in this thesis, has been done by Cootes et al. beginning with [24, 25, 29]. Statistical Shape Models have turned out to be an effective tool in image segmentation and image interpretation. Using the model it is possible to describe new formerly unseen shapes, while ensuring that they are reasonable shapes given the shape class. Another important aspect is the possibility of interpreting the shape parameters for example for the purpose of medical diagnosis.

The shapes are typically boundaries of some objects and are commonly defined by landmarks, curves or surfaces. In this thesis the shapes are typically n-dimensional manifolds. Specifically often one-dimensional manifolds, i.e. curves, are considered but the techniques discussed work equivalently for n-dimensional manifolds, for example surfaces. For more complicated shapes consisting of several manifolds problems are usually solved by treating one manifold at a time.

In the most simple case a *Point distribution model* (PDM) is built, representing both the mean shape and the variability modes of the shape class. The mean shape and each mode are represented by a number of *landmarks* defining the shape. In order to build the model, a collection of correctly annotated example shapes is established. This was first done manually and later semi-automatically. The sets of landmarks are then aligned using Procrustes-analysis. After the shapes have been aligned, a principal component analysis (PCA) is performed to reduce dimensionality. Shapes can now be described by a multivariate Gaussian model as the mean shape plus some variation along orthogonal directions defined by a number of modes of shape variation. In this way the shape model allows for considerable meaningful variability while still being specific to the class structure it represents. PCA has also been used to model appearance, i.e. pixel intensities at corresponding shape locations [28, 82], and illumination [73]. Since the introduction of these models there has been a lot of work in this area and this thesis is part of this.

When building these models, it is essential to establish correct correspondences between points on shapes throughout the training set. This can be, and often is, done by annotating landmarks by hand. This can however be difficult and time consuming so automatic and semi-automatic approaches have been developed [31]. This thesis deals with such automatic algorithms for establishing correspondences.

2.2 Shape

2.2.1 Definition

One of the earliest instances of geometrical shape analysis was developed from a biological point of view by D'arcy Thompson [137]. Since then the field of shape modeling and shape analysis has grown and is of great importance in a wide variety of disciplines, e.g. image segmentation, classification and interpretation.

Let us first define the objects whose shape will be modelled.

Definition 2.2.1 (Parameterized Geometric Object)**.**

- A geometric object is a subset of $\mathbb{R}^{n_d}$, where n_d is the dimension, e.g. $n_d = 2$ for planar curves and $n_d = 3$ for surfaces.
- A parameterization of a geometric object assigns a set of parameter values to each point in the object.

Now we need to define what we mean by the shape of such objects.

Definition 2.2.2 (Shape)**.**

- Conceptually, the shape of a geometric object is all the geometric information that remains when differences explained by the effects of translation, rotation and scale are filtered out [78].
- More specifically, two geometric objects have the same shape, if they can be transformed into each other by similarity (translation, rotation and positive scale i.e. no reflection) transformations.
- Two geometric objects have the same size and shape, if they can be transformed into each other by Euclidean (translation and rotation) transformations.

Here similarity transformations are used to define what is meant by same shape. Similarly any group of transformations $\mathcal{G} : \mathbb{R}^{n_d} \mapsto \mathbb{R}^{n_d}$ can be used. Let $\mathcal{S}$ be a geometric object and let

$$\mathcal{GS} = \{\mathbf{G}\mathcal{S} | \mathbf{G} \in \mathcal{G}\} \tag{2.1}$$

be the set of all geometric objects obtained from $\mathcal{S}$ by some transformation $\mathbf{G} \in \mathcal{G}$. This set $\mathcal{GS}$ is called the the $\mathcal{G}$*-orbit* of $\mathcal{S}$. This gives us a more general definition of shape [41].

Definition 2.2.3 ($\mathcal{G}$-shape)**.** $\mathcal{G}$-shape is a function s defined on geometric objects, that is constant on $\mathcal{G}$-orbits and different on different $\mathcal{G}$-orbits. Two geometric objects are hence said to have the same $\mathcal{G}$-shape, if they belong to the same $\mathcal{G}$-orbit.

The definition of shape above is then identified with Similarity-shape ($\mathcal{G}$ is the group of Similarity transformations) and the definition of same size and shape is identified with Euclidean-shape. In the rest of the thesis the shorter notation shape will be used to mean Similarity-shape. The word shape will also be used de refer to specific geometric objects belonging to the $\mathcal{G}$-orbit. Whether the function value of the $\mathcal{G}$-orbit or the specific representative in form of a geometric object is meant should be clear from the context.

Another important use of $\mathcal{G}$-shape is to work with parameterized geometric objects and let $\mathcal{G}$ be the set of reparameterizations, i.e. transformations that do not change the set of points making up the geometric object but changes the assigned parameter values of the points. In this way objects can be said to have the same shape if one can be transformed into the other by only changing the parameter values. Typically this would be used together with for example similarity transformations, so that two object are considered to have the same shape if one can be transformed into the other by a similarity transformation and a reparameterization.

2.2.2 Shape Representation and the Correspondence Problem

In order to model a class of shapes we need s set of geometric objects $\{\mathcal{S}_1, \ldots, \mathcal{S}_{n_s}\}$ in $\mathbb{R}^{n_d}$ from the shape class to be modeled. This set is called the *training set* and should be representative of the shape class.

In order to be able to analyze objects of different shape, a way of measuring the distance between shapes and methods for the statistical analysis of shapes are necessary. The simplest way to accomplish this is to *annotate* a geometric object $\mathcal{S}$ in $\mathbb{R}^{n_d}$ with a number of points $\mathcal{X} = \{\mathbf{l}_1, \ldots, \mathbf{l}_{n_p}\}$ on the object. Such points are called *landmarks*. When choosing landmarks there are many things to consider. For example, they should represent features, which are present in most instances of objects from a certain class of shapes and which can be located on those shapes.

To be able to compare shapes via the landmarks, the landmarks have to be in correspondence. This means that on each shape in the training set, the landmarks with the same number should be located at the same point on the object. For example, if the model is of the outline of a hand, landmark number 5 may be located at the tip of the index finger on each shape. In order for the shape class to be modeled properly we need such corresponding landmarks on all parts of the shape. This is often referred to as establishing a dense correspondence.

Traditionally, landmarks have been described as being of different types: *anatomical*, *mathematical* and *pseudo-landmarks*.

Anatomical Landmarks. An anatomical landmark is a point that is located at a meaningful point on the object that can be located on most instances from the shape class through use of knowledge about the object. This can be e.g. the tip of a finger or the top left corner of the wind shield on a car. Often anatomical landmarks are assigned by experts on the object class being modeled. Often this can actually be one of the best

ways of creating a high quality training set (see Chapter 8). The problem is that it is very difficult and time consuming so in practice it is often not an option and for many types of data it is not feasible at all.

Mathematical Landmarks. Mathematical landmarks are points whose correspondences are determined according to some mathematical property. In simple cases this can be for example points of extreme curvature or junctions of lines. Automatic algorithms using some mathematical function for establishing correspondences have been an active field of research in the last decade and this thesis deals with this topic.

Pseudo-Landmarks. Anatomical and mathematical landmarks may only give a sparse description of the shape of the object. One way to obtain a richer shape description is to place so called pseudo-landmarks in between the well-defined landmarks. Pseudo-landmarks are points along the boundaries which are typically arranged to be uniformly spaced between well-defined landmarks.

In another approach to handling the correspondence problem the entire images in the training set are automatically registered using the non-linear registration technique of your choice [5, 107, 128, 138]. This in a sense gives you correspondences for every pixel/voxel in the registered images. However, often the images contain other things than the object that is to be modeled and parts of the object may be occluded or undetectable due to poor image quality, which can be problematic for this approach. Also, in many cases expert knowledge is needed for segmentation of the object outline and/or correspondence establishment, which can also be a problem for the registration approach. The image registration approach is not used in this thesis.

To illustrate the importance of establishing good correspondences see Figure 2.1 and Figure 2.2. Both these models are build from the same training set but with different correspondences. Bad correspondences result in false modes of variation which means both that the model cannot represent all legal instances from the shape class in a good way and that the model can produce instances that are illegal instances that do not belong to the shape class.

Instead of explicitly choosing a set of landmarks to represent the shape for further analysis, the parameterization of the object can be chosen so that correspondence is established. In this way, in a sense, every point on the object becomes a landmark. The points with the same parameter values on different objects should then be in correspondence across the whole training set. In this way, it is possible to keep the continuous shape further in the analysis so that the model conceptually handles continuous shapes and the sample points are introduced later for numerical computations only. To perform numerical calculations, in practice the shapes are sampled at a number of corresponding points. Temporarily during calculations then, these sample points function as landmarks, but unlike a standard set of landmarks, this set of sample points can be different at different calculation steps, as sampling strategies are free to differ as long as a good approximation of the shape is achieved. However, whether we have selected explicit landmarks or parameterizations, for analysis of shape through numerical calculations, the geometric object in

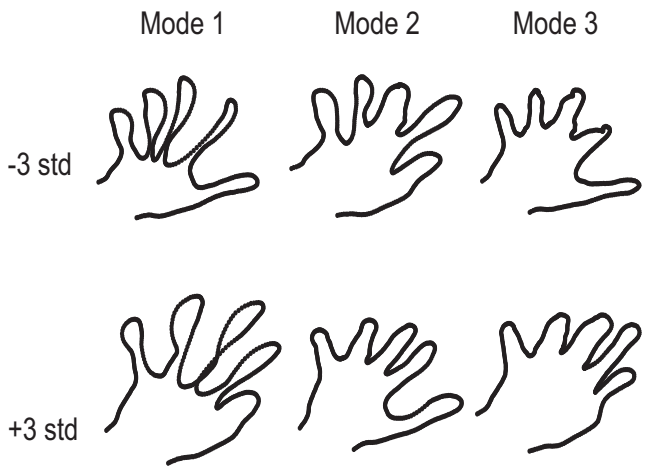


Figure 2.1: Shape model of a hand. Mean-shape ± 3 standard deviations of the first three modes of variation of a model built from arc-length parameterization (equidistant landmarks along the curve). The poor correspondence results in illegal instances of the shape class.

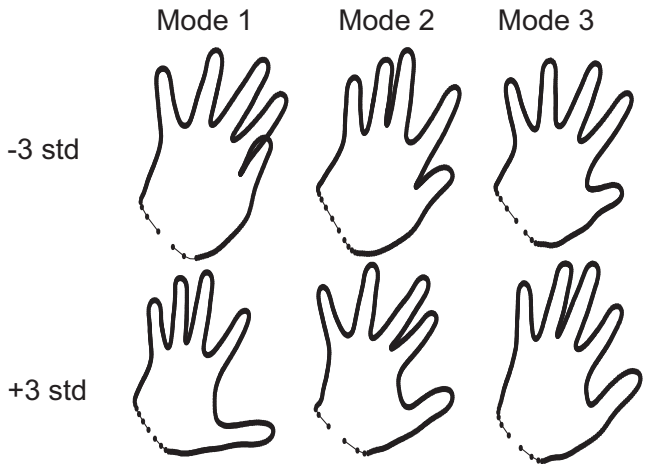


Figure 2.2: Shape model of a hand. Mean-shape ± 3 standard deviations of the first three modes of variation of a model built from optimized correspondences via optimized parameterizations.

$\mathbb{R}^{n_d}$ is represented by a set of n_p landmarks or temporary sample points, each of which is a point in $\mathbb{R}^{n_d}$. Typically, $n_d = 2$ for planar shapes (most typically simple curves) or $n_d = 3$ for shapes in space (typically closed surfaces) or maybe $n_d = 4$ for 3 dimensions of space and one of time. We will sometimes refer to these points as landmarks even though in practice we are typically working with temporary samples from parameterized objects.

The shape is typically represented as a vector by stacking the coordinates of the landmarks or sample points in a suitable order. This $n_d n_p \times 1$ vector $\mathbf{x}$, where $n_p$is the number of landmarks and n_d is the dimension of each landmark, is called a *shape vector*. In two dimensions, it is often practical to use a representation in the complex plane, a landmark $\mathbf{l}$ can be represented by a complex number and $\mathbf{x}$ is a vector in $\mathbb{C}^{n_p}$.

A training set of objects $\{\mathcal{X}_1, \ldots, \mathcal{X}_{n_s}\}$, with established correspondences, can now be represented as a matrix $\mathbf{X}_0 = [\mathbf{x}_1, \ldots, \mathbf{x}_{n_s}]$, where $\mathbf{x}_i$ is the shape vector corresponding to the object $\mathcal{X}_i$ $(i = 1, \ldots, n_s)$.

There are several other alternatives for representing the shape, while still being able to analyze the shape variation by performing a principal component analysis [77, 89]. One interesting example is to represent shapes with signed distance functions as used in level set techniques and doing PCA on column stacked samples of those functions rather than on coordinates of corresponding points on the object outlines [90]. This seems to avoid the correspondence problem but the resulting models may be similar to models built from poor correspondences rather than carefully optimized ones. The strengths and weaknesses of these different approaches are not the topic of this thesis and the most common approach described above will be used throughout.

2.3 Alignment and Mean of the Training Set

Before calculating mean-shapes and doing other types of analysis needed for building the model, the shapes in the training set are usually aligned in some way to remove variation due to translation, scale and rotation, since this is not variation in shape. Methods for alignment are many but the most common approach in this context is so called Procrustes analysis [28]. This is what will used for the most part of this thesis but some alternatives will also be considered.

2.3.1 Calculating the Mean Shape

For modeling the shape class it is necessary to know the mean-shape. The models will represent shapes as the deviation from the mean-shape. There are many ways to define the mean-shape and this is an area of research in itself. Typically, before estimating the final mean shape, the training set must be aligned in some way. This is discussed below. However, approximate mean-shapes are sometimes used during alignment. In this thesis,

for the most part, the mean shape will be straightforwardly calculated by

$$\bar{\mathbf{x}} = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{x}_i \ , \tag{2.2}$$

where n_s is the number of shapes in the set.

2.3.2 Procrustes Analysis

The Procrustes distance is a least-squares measure of the difference in shape between geometric objects [55]. Procrustes alignment gives this distance by minimizing a Euclidean distance between shapes over similarity transforms. Here follows a brief review. For more on this, see for example [38, 56, 79, 122].

Assume that one shape $\mathbf{x}_i = [\mathbf{l}_i^1; \ldots; \mathbf{l}_i^{n_p}]$ is to be aligned to another shape $\mathbf{x}_j = [\mathbf{l}_j^1; \ldots; \mathbf{l}_j^{n_p}]$, where we here think of the shapes as vectors with landmarks as elements. The similarity transform $\mathbf{S}$ used to transform $\mathbf{x}_i$ operates on each landmark $\mathbf{l}_i^k$ separately:

$$\mathbf{S}(\mathbf{x}_i) = \begin{bmatrix} \mathbf{S}(\mathbf{l}_i^1) \\ \vdots \\ \mathbf{S}(\mathbf{l}_i^{n_p}) \end{bmatrix} \ . \tag{2.3}$$

Now choose $\mathbf{S}$ so that the squared Euclidean distance between $\mathbf{S}(\mathbf{x}_i)$ and $\mathbf{x}_j$ is minimized, i.e. solve

$$\min_{\mathbf{S}} ||\mathbf{x}_j - \mathbf{S}(\mathbf{x}_i)||^2 \ , \tag{2.4}$$

where

$$\mathbf{x}_j - \mathbf{S}(\mathbf{x}_i) = \begin{bmatrix} \mathbf{l}_j^1 \\ \vdots \\ \mathbf{l}_j^{n_p} \end{bmatrix} - \mathbf{S}\left(\begin{bmatrix} \mathbf{l}_i^1 \\ \vdots \\ \mathbf{l}_i^{n_p} \end{bmatrix}\right) = \begin{bmatrix} \mathbf{l}_j^1 - \mathbf{S}(\mathbf{l}_i^1) \\ \vdots \\ \mathbf{l}_j^{n_p} - \mathbf{S}(\mathbf{l}_i^{n_p}) \end{bmatrix} \ . \tag{2.5}$$

When $\mathbf{S}$ is chosen optimally $\mathbf{S}(\mathbf{x}_i)$ is called the full Procrustes fit $\mathbf{x_i}^P$.

In 2D, simplifications can be made by working with the 2D landmarks as points in the complex plane. This means that $\mathbf{x}_i \in \mathbb{C}^{n_p}$ and $\mathbf{x}_j \in \mathbb{C}^{n_p}$. The similarity transformation $\mathbf{S}$ in this context is simply $\mathbf{S}(\cdot) = u + v\cdot, u, v \in \mathbb{C}$. Transformation of the landmarks then takes the form $\mathbf{S}(\mathbf{l}) = u + v\mathbf{l}$ and hence

$$\mathbf{S}(\mathbf{x}_i) = u\mathbf{1}_{n_p} + v\mathbf{x}_i \ , \tag{2.6}$$

where $\mathbf{1}_{n_p}$ is a $n_p \times 1$ vector in $\mathbb{C}^{n_p}$ with all elements equal to 1. The minimization then takes the form

$$\min_{\mathbf{S}} ||\mathbf{x}_j - \mathbf{S}(\mathbf{x}_i)||^2 = \min_{u,v\in\mathbb{C}} ||\mathbf{x}_j - u\mathbf{1}_{n_p} - v\mathbf{x}_i||^2 \ , \tag{2.7}$$

which can be calculated as

$$\min_{u,v\in\mathbb{C}}(\mathbf{x}_j - u\mathbf{1}_{n_p} - v\mathbf{x}_i)^*(\mathbf{x}_j - u\mathbf{1}_{n_p} - v\mathbf{x}_i) \ , \tag{2.8}$$

where $(\cdot)^*$ denotes the transpose of the complex conjugate of $(\cdot)$.

Theorem 2.3.1. *(Full Procrustes fit in 2D) If the shape vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ with landmarks in the complex plane have first been centered (transposed to have their mean at the origin) so that $\mathbf{1}_{n_p}^T\mathbf{x}_i = 0$ and $\mathbf{1}_{n_p}^T\mathbf{x}_j = 0$, then the full Procrustes fit $\hat{u} + \hat{v}\mathbf{x}_i$ has the parameters*

$$\hat{u} = 0 \ , \tag{2.9}$$

$$\hat{v} = \mathbf{x}_i^*\mathbf{x}_j/(\mathbf{x}_i^*\mathbf{x}_i) \ . \tag{2.10}$$

Proof. The proof can be found for example in [41]. ■

For the general case with landmarks of any dimension we can not use the representation in the complex plane, but still Umeyama [142] has prooved explicit formulas for the Procrustes alignment.

Theorem 2.3.2. *(Full Procrustes fit) Assume $\mathbf{x}_i = [\mathbf{l}_i^1; \ldots; \mathbf{l}_i^{n_p}]$ and $\mathbf{x}_j = [\mathbf{l}_j^1; \ldots; \mathbf{l}_j^{n_p}]$ are shape vectors with landmarks in n_d dimensions each represented as a $n_d \times 1$ vector. Let*

$$\mu_i = \frac{1}{n_p}\sum_{k=1}^{n_p}\mathbf{l}_i^k \ , \tag{2.11}$$

$$\mu_j = \frac{1}{n_p}\sum_{k=1}^{n_p}\mathbf{l}_j^k \ , \tag{2.12}$$

$$\sigma^2 = \frac{1}{n_p}\sum_{k=1}^{n_p}||\mathbf{l}_i^k - \mu_i||^2 \ , \tag{2.13}$$

$$\Sigma = \frac{1}{n_p}\sum_{k=1}^{n_p}(\mathbf{l}_j^k - \mu_j)(\mathbf{l}_i^k - \mu_i)^T \tag{2.14}$$

and perform a singular value decomposition of Σ to get $\Sigma = \mathbf{UDV}^T$. Now let

$$\mathbf{A} = \begin{cases} \mathbf{I} & if \ \ det(\Sigma) \geq 0 \\ diag(1, 1, \ldots, 1, -1) & if \ \ det(\Sigma) < 0 \end{cases} \tag{2.15}$$

and let $\mathbf{B} = \mathbf{A}$ unless $rank(\Sigma) = n_d - 1$ in which case

$$\mathbf{B} = \begin{cases} \mathbf{I} & if \ \ det(\mathbf{U})det(\mathbf{V}) = 1 \\ diag(1, 1, \ldots, 1, -1) & if \ \ det(\mathbf{U})det(\mathbf{V}) = -1 \end{cases} \ . \tag{2.16}$$

Finally we can state that if $rank(\Sigma) \geq n_d - 1$ *the full Prucrustes fit* $\mathbf{x}_i^P$ *of* $\mathbf{x}_i$ *onto* $\mathbf{x}_j$ *is*

$$\mathbf{x}_i^P = c\mathbf{R}\mathbf{x}_i + t \ , \tag{2.17}$$

where

$$c = \frac{1}{\sigma^2} tr(\mathbf{DA}) \ , \tag{2.18}$$

$$\mathbf{R} = \mathbf{UBV}^T \ \ and \tag{2.19}$$

$$t = \mu_j - c\mathbf{R}\mu_i \ . \tag{2.20}$$

Proof. The proof can be found in [142]. ■

For aligning the whole training set (often referred to as Generalized Procrustes Analysis), usually one of a number of alternative iterative procedures is used, see for example [28, 56, 132]. The steps followed in this thesis are described in the box below.

input : The original training set: $\mathbf{x}_i$, i = 1,...,n_s
output: The Procrustes aligned training set

1 Translate the shapes in the training set so that the mean coordinates of each shape are at the origin
2 Chose one example as initial guess of the mean shape, $\bar{\mathbf{x}}$, and scale it so that $\|\bar{\mathbf{x}}\| = 1$.
3 Record the first estimate as $\bar{\mathbf{x}}_0$ to define the default reference frame.
4 **repeat**
5 Align each of the shapes in the set to the estimated mean shape.
6 Estimate a new mean shape $\bar{\mathbf{x}}$.
7 Normalize the new mean shape by aligning it with $\bar{\mathbf{x}}_0$ and scaling it so that $\|\bar{\mathbf{x}}\| = 1$.
8 **until** *convergence*

Algorithm 1: Iterative Procrustes alignment of training set

In 2D it is possible to avoid the iterative procedure.

Theorem 2.3.3. *(2D Procrustes mean shape) Assume the shapes are sampled in a set of corresponding points and these points are represented as complex numbers. Each shape* $\mathbf{x}_i$ *is then a column of complex numbers. Assume further that the shapes are centered (translated so they have their mean at the origin) and normalized so that they have euclidean size one. The mean shape* $\bar{\mathbf{x}}$ *can then be calculated without iteration as the eigenvector corresponding to the largest eigenvalue of the complex sum of squares and products matrix*

$$\mathbf{S} = \sum_{i=1}^{n_s} \mathbf{x}_i \mathbf{x}_i^* \ . \tag{2.21}$$

Proof. The proof can be found for example in chapter 3 of [38]. ■

Continuing with the 2D case with landmarks as complex numbers and using matrix notation with the shapes as columns in the complex matrix $\mathbf{X}_0$, the mean-shape $\bar{\mathbf{x}}$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{X}_0\mathbf{X}_0^*$. Often the number of sample points is much larger than the number of shapes and then it is better to instead calculate the eigenvector corresponding to the largest eigenvalue of $\mathbf{X}_0^*\mathbf{X}_0$. Call this eigenvector $\bar{\mathbf{x}}_0$. We then get $\bar{\mathbf{x}} = \mathbf{X}_0\bar{\mathbf{x}}_0/||\mathbf{X}_0\bar{\mathbf{x}}_0||$.

Note that rotations of the mean shape calculated as the eigenvector in this way are also solutions. For visualization it is often desirable to not introduce a rotation like this. Therefore it is common practice to rotate the calculated mean shape before using it. This can either be done by rotating it to align it to one of the shapes, or some mean of the rotations to all the shapes can be used. This alignment of the mean is also done in the iterative algorithm above to prevent rotational drift.

There is also the possibility of calculating the mean-shape without iteration in higher dimensions (i.e. 3 in practice) using multidimensional scaling methods [80]. However, these methods may not give the correct mean-shape in the sense that the result may not only be rotated (as is the case for the 2D method above) but also reflected. Because of this, and the fact that the iterative methods usually converge quickly, these multidimensional scaling methods are not used in this thesis (and rarely in shape analysis at all).

2.3.3 Discussion on Procrustes Alignment

There are some weaknesses of this alignment approach. One problem is outliers. If one of the landmarks is very wrong for some reason this can destroy the alignment. Another thing is that the error is scale dependent, so without the normalisation of the mean in every step in the alignment of a set everything would converge to the origin. However, with the normalisation, this is not a problem. Also, if some landmark is missing in one or more of the shapes the standard procedure does not work. Finally, the alignment produced may be neither the one a human would think natural nor the one that isolates a mode of variation in the best way. There have been attempts to handle these problems and some methods will be presented later in this thesis.

2.3.4 Alignment-introduced Nonlinearities

During alignment, if there are large variations in shape, non-linearities may be introduced, e.g. as an effect of scaling. This undesired scaling effect can to some extent be countered by projection to the tangent space. The tangent space is the hyperplane of all vectors that are orthogonal to the mean-shape $\bar{\mathbf{x}}$ and that pass through $\bar{\mathbf{x}}$, i.e. all the vectors $\mathbf{x}$ such that $(\bar{\mathbf{x}} - \mathbf{x})^T\bar{\mathbf{x}} = \mathbf{0}$, or $\bar{\mathbf{x}}^T\mathbf{x} = \mathbf{1}$, if $||\bar{\mathbf{x}}||^2 = \mathbf{1}$ [28]. The mean-shape is the origin in the tangent space. However, when there are such large variations in shape, most non-linearities will probably not be artificially caused in this way (but instead will

be an important aspect of the shape variation) and these variations will not become linear by projection to the tangent space and then non-linear shape models (not the topic of this thesis) are needed. It may be noted in this context that when there are only relatively small shape variations the tangent space can be used e.g. in inference hypothesis tests [38]. For the PCA (on such data with relatively small shape variation), the difference between projecting to the tangent space and performing the PCA on the Procrustes residuals is typically very small [38].

2.4 Dimensionality Reduction

2.4.1 Introduction

Let a training set of n_s aligned shapes in n_d dimensions be given. Each shape is sampled in n_p points and the shapes are then represented by vectors $\mathbf{x}_i$ ($i = 1, \ldots, n_s$). Each shape vector can also be thought of as a sample from a probability distribution in $n_p n_d$-dimensional space. By estimating this distribution from the training set, new samples can then be generated from the distribution and they should then belong to the same shape class. Also, a given shape can be analyzed be looking at the likelihood for its shape under the assumption of its belonging to a certain shape class, i.e. being a sample from a certain distribution.

The shape model will be denoted by $\mathcal{M}$ and a shape generated by the model will be denoted by $\mathcal{M}(\mathbf{b})$, where $\mathbf{b}$ is the parameter vector that produces a certain shape $\mathbf{x}$.

Since $n_p n_d$ can often be very large, some form of dimensionality reduction is useful, so that $\mathbf{b}$ is of significantly smaller dimension than $\mathbf{x}$.

The training set is viewed as a set of points in $n_p n_d$ dimensional space. The key now is that for most shape classes this point cloud is approximately contained in a subspace of significantly lower dimension. This has also been stated as this subspace having a low fractal dimension [120].

It would be advantageous to describe the shapes with coordinates in this subspace instead. This is what dimensionality reduction does.

Principal Component analysis (PCA) is the most commonly used technique for dimensionality reduction of the shape space and it is an effective approach. Several other methods have however been proposed, for example Mixture Models [27], sparse-PCA [121], kernel-PCA [140], Independent Component Analysis (ICA) [144], Orthomax [126], Factor Analysis [93], Maximum Autocorrelation Factors (MAF) [40, 87], Minimum Noise Fraction (MNF) [59], Multi-layer perceptron point distribution models (MLPPDM) [124] and manifold learning techniques like ISOMAP [85] and Locally Linear Embedding (LLE) [106].

In this thesis the focus is not on the choice of dimensionality reduction method and therefore PCA will be used throughout. The topics in this thesis are equally relevant when using some other method.

2.4.2 PCA

PCA (also known as Karhunen-Loève transform or Hotelling transform) was introduced in the beginning of the 20th century [69, 102]. The technique computes a new basis for the shape space such that this new basis is adapted to the points in the training set. The first basis vector is chosen as the direction of maximum variance in the point cloud and each of the following basis vectors are in turn chosen as directions with maximum variance with the added constraint that the new basis vectors are orthogonal to the ones already chosen. The points are then described with their positions along these axes. Approximation consists of choosing a small number of axis. An attractive feature of PCA is that the linear approximating reconstruction of the data is optimal, i.e. the error is minimal given linear reconstruction with a certain number of modes.

The principal components axes are typically calculated by performing a singular value decomposition (SVD) of the tangent space coordinates or the Procrustes residuals of the shapes in the training set. Very briefly, any $M \times N$ matrix $\mathbf{A}$ can be factored as

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \sum_{i=1}^{N} s_i \mathbf{U}_i \mathbf{V}_i^T \ , \tag{2.22}$$

where $\mathbf{U} = [\mathbf{U}_1, \ldots, \mathbf{U}_M]$ is an $M \times M$ orthogonal matrix with the columns $\mathbf{U}_i$, $\mathbf{V} = [\mathbf{V}_1, \ldots, \mathbf{V}_N]$ is an $N \times N$ orthogonal matrix with the columns $\mathbf{V}_i$and $\mathbf{S}$ is an $M \times N$ diagonal matrix with non-negative diagonal elements (the singular values). This is the singular value decomposition of $\mathbf{A}$. The singular values s_i $(i = 1, \ldots, N)$ are the positive square roots of the eigenvalues of the matrix $\mathbf{A}^T\mathbf{A}$ and can also be interpreted as the standard deviations of the data along the principal component axes.

Let

$$\mathbf{X} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \ldots, \mathbf{x}_{n_s} - \bar{\mathbf{x}}] \tag{2.23}$$

be an $n_d n_p \times n_s$ matrix containing the vectors describing the tangent space coordinates or the deviations of the shapes $\mathbf{x}_i$ $(i = 1, \ldots, n_s)$ from the the mean shape $\bar{\mathbf{x}}$, i.e. the Procrustes residuals (or alternatively let $\mathbf{X}$ contain the tangent space coordinates).

The maximum likelihood approximation of the covariance matrix of the shapes is then

$$\mathbf{C} = \frac{1}{n_s - 1}\mathbf{X}\mathbf{X}^T \ . \tag{2.24}$$

The principal components of $\mathbf{X}$ can be found by solving the eigenvector problem

$$\mathbf{C}\mathbf{\Phi} = \mathbf{\Lambda}\mathbf{\Phi} \ , \tag{2.25}$$

where $\mathbf{\Phi}$ is an orthogonal matrix with the eigenvectors as columns and $\mathbf{\Lambda}$ is a diagonal matrix containing the corresponding eigenvalues.

This is solved by an SVD-factorization of $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{USV}^T \ , \tag{2.26}$$

where $\mathbf{U}$ and $\mathbf{V}$ are $(n_d n_p \times n_d n_p)$ and $(n_s \times n_s)$ orthogonal matrices respectively and $\mathbf{S}$ is a $(n_d n_p \times n_s)$ diagonal matrix containing the singular values of $\mathbf{X}$.

This gives

$$\mathbf{XX}^T = \mathbf{USV^TVS}^T\mathbf{U}^T = \mathbf{USS}^T\mathbf{U^T} = \mathbf{UDU^T} \ , \tag{2.27}$$

where $\mathbf{D}$ is a diagonal matrix with the first elements of the diagonal being the squares of the singular values. This in turn gives

$$\frac{1}{n_s - 1}\mathbf{XX}^T\mathbf{U} = \frac{1}{n_s - 1}\mathbf{UD} \ . \tag{2.28}$$

From this it is clear that $\mathbf{U}$ is an orthogonal matrix with eigenvectors of $\mathbf{C} = \frac{1}{n_s-1}\mathbf{XX}^T$ as columns and $\frac{1}{n_s-1}\mathbf{D}$ is the diagonal matrix with the corresponding eigenvalues.

Note that, in many practical cases, $n_d n_p$ is much larger than n_s and then there may be a more efficient way of doing these computations by looking at $\frac{1}{n_s-1}\mathbf{X}^T\mathbf{X}$ instead of $\frac{1}{n_s-1}\mathbf{XX}^T$ [28]. Whether this needs to be done depends among other things on the implementation of the SVD.

2.5 Shape Model

Define $\mathbf{\Phi} = \left[\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_{n_m}\right]$, where $\mathbf{\Phi}_i = \mathbf{U}_i$ (column number i) and $n_m \leq n_s - 1$. The columns of $\mathbf{\Phi}$ are the n_m eigenvectors of the shape covariance matrix $\mathbf{C}$ corresponding to the n_m largest eigenvalues. These vectors are the basis that will be used to describe the shapes in the shape class. They will be referred to as modes of variation or shape modes. Remember that they are used for describing the deviation from the mean shape $\bar{\mathbf{x}}$, so if the modeling has been successful, any shape $\mathbf{x}$ from the shape class can now be approximated well as $\mathbf{x}'$ and be described as

$$\mathbf{x} \approx \mathbf{x}' = \bar{\mathbf{x}} + \mathbf{\Phi b} = \bar{\mathbf{x}} + \sum_{j=1}^{n_m} \mathbf{\Phi}_j b^j \ , \tag{2.29}$$

where $\mathbf{b}$ is the parameter vector for describing the shape using the model. If the correspondences of the points in $\mathbf{X}$ were correct and the variations in the shape class are not too chaotic the shape modes should capture relevant and reasonable shape variations, see Figure 2.2.

By choosing to include only the most important principal components the dimensionality can be reduced further. The key is that the principal components are ordered so

that the first component explains the largest part of the variation in the training set and so on. Since the principal components are orthogonal, the total variance in the training set is the sum of the variances along the principal components and these are equal to the eigenvalues of the covariance matrix described above or equivalently the squares of the singular values in the SVD. One common approach is to include enough principal components to cover a certain fraction of the total variance.

Let λ_i ($i = 1, \ldots, N$, $\lambda_i \geq \lambda_{i+1}$) be the N nonzero eigenvalues of the covariance matrix $\mathbf{C}$ of the training data (2.24) and let $\mathbf{\Phi}_i$ be the corresponding eigenvectors. The variance of the data in the direction of the principal component $\mathbf{\Phi}_i$ is equal to the eigenvalue λ_i. The total variance of the model is $V_{Tot} = \sum_{i=1}^{N} \lambda_i$. If p is the fraction of the total variance that should be explained by the model, the number of modes n_m to use can be decided by choosing n_m as small as possible but such that

$$\sum_{i=1}^{n_m} \lambda_i \geq p\, V_{Tot} \ . \tag{2.30}$$

Another intuitive approach is to choose the number of modes so that all shapes in the training set can be approximated so that all landmarks are within a certain distance of the original shape. There are other methods for automatic model selection that can be used, but the details concerning the choice of the number of modes is not the topic of this thesis.

Assuming a model $\mathcal{M}(\mathbf{b}) = \bar{\mathbf{x}} + \mathbf{\Phi}\mathbf{b}$, new shapes are generated by varying $\mathbf{b}$ and a shape $\mathbf{x}$ is approximated by $\mathbf{x}' = \bar{\mathbf{x}} + \mathbf{\Phi}\mathbf{b}$. Put $\mathbf{\Phi}\mathbf{b} = \mathbf{x}' - \bar{\mathbf{x}}$ and multiply with $\mathbf{\Phi}^T$ from the left to get

$$\mathbf{b} = \mathbf{\Phi}^T(\mathbf{x}' - \bar{\mathbf{x}}) \ . \tag{2.31}$$

This formula is exact and if we replace $\mathbf{x}'$ with $\mathbf{x}$ we get the formula for calculating the parameters for optimal approximation of $\mathbf{x}$,

$$\mathbf{b} = \mathbf{\Phi}^T(\mathbf{x} - \bar{\mathbf{x}}) \ . \tag{2.32}$$

The columns of the matrix $\mathbf{\Phi}$ are the n_m modes of shape variation and span a linear subspace in shape space. The approximation consists of orthogonal projection onto this linear subspace.

Another use of this is analysis of shapes. The probability distribution of the $\mathbf{b}$-parameters can be modeled for example as a Gaussian using the training set. The probability of a new shape can then be estimated.

2.6 Measuring Shape Model Quality

Traditionally, three properties have been used to measure model quality. First of all, the model should be able to represent the shape class. A model that is able to represent any

instance of the class is said to be *general*. Secondly, the model should not be able to represent instances not belonging to the shape class. A model which only produces instances of the correct shape class for reasonable parameter values is called *specific*. Thirdly, the ideal model is also *compact* in the sense that it needs few parameters with small values to represent all shapes in the shape class.

Apart from these three properties it is also common to evaluate correspondences, both in the training set and on shapes produced by the model, by ploting the landmarks on the shapes and visually evaluating the correspondences.

When calculating these quality measures, any norm could be used. In this thesis, the norm used is $||f(x)|| = \sqrt{\int |f(x)|^2 dx}$ (or $||f(x)|| = \sqrt{\sum |f(x)|^2}$ for discrete data), i.e. L^2-norm (or the l^2 Euclidean norm for discrete data), unless otherwise stated. As we will see, many of the problems with the standard measures are related to the fact that they measure similarity between shapes using a norm of this type. If there was a general measure that could pick out relevant information and determine shape similarity so that the distance between shapes reflect the way they differ and not only the distance using the L^p-norm and if this measure was invariant to the parameterizations and did not need any ground truth information, many of the problems with the standard measures might be solved. However, such a measure remains to be discovered.

2.6.1 Generality

Qualitatively, by generality is meant that the model should be able to generalize to formerly unseen shapes in the sense that the model should be able to describe all shapes of the class and not only those of the training set. Quantitatively, this is measured by *leave one out* tests, where a model is built using all but one of the shapes in the training set. The left out shape is then approximated using the model. In [31] the error for one left out shape is the squared norm of the difference between the true left out shape and the approximation produced by the model. Generality is then measured as the mean over all n_s left out shapes:

$$G(n_m) = \frac{1}{n_s} \sum_{j=1}^{n_s} ||\mathbf{x}_j - \mathbf{x}'_j(n_m)||^2 \ , \tag{2.33}$$

where $\mathbf{x}_j$ is the left out shape and $\mathbf{x}'_j(n_m)$ is the approximation produced by the model using n_m modes.

2.6.2 Specificity

It is in general desirable that a shape model is specific. Qualitatively, a specific model can only represent shapes from the shape class for valid parameter values. This has been measured by generating a large number (N) of shapes by picking random parameter

values for the model. Each sample shape is then compared to the most similar shape in the training set. A quantitative measure for this [31] is:

$$S(n_m) = \frac{1}{N}\sum_{j=1}^{N} ||\mathbf{y}_j - \mathbf{y}'_j(n_m)||^2 \ , \tag{2.34}$$

where $\mathbf{y}'_j$ are shape examples generated by the model, $\mathbf{y}_j$ is the nearest member of the training set to $\mathbf{y}'_j$ and n_m is the number of modes used to create the samples. Formally,

$$\mathbf{y}_j = \arg \inf_{\mathbf{x}_j \in \{\mathbf{x}_1,\ldots,\mathbf{x}_{n_s}\}} ||\mathbf{x}_j - \mathbf{y}'_j(n_m)||^2 \ , \tag{2.35}$$

where $\{\mathbf{x}_1, \ldots, \mathbf{x}_{n_s}\}$ is the training set.

In [111] this is interpreted as an approximation of

$$S(n_m) = \int_{\mathbf{\Omega}} p(\mathbf{y}'_j)||\mathbf{y}_j - \mathbf{y}'_j(n_m)||d\mathbf{y}'_j \ , \tag{2.36}$$

where $p(\mathbf{y}'_j)$ is the distribution function describing the probability that the model generates the shape $\mathbf{y}'_j$ and $\mathbf{\Omega}$ is the shape space.

A more general definition is

$$S(n_m) = \int_{\mathbf{\Omega}} ||\mathbf{y}_j - \mathbf{y}'_j(n_m)||^2 d\mu_p(\mathbf{y}'_j) \ , \tag{2.37}$$

where μ_p is the probability measure for the shapes $\mathbf{y}'_j$.

In [141] specificity is interpreted as a graph based estimator of cross-entropy and KL divergence. However in order to do this estimation a number of limits and assumptions are needed, which means that this estimation is usually not valid in practice.

2.6.3 Compactness

Qualitatively, a compact model is a model that represents all shapes of the class with as little variance as possible in the shape variation modes and preferably with few modes. A quantitative measure that tries to capture this is the sum of variances of the model [31],

$$C(n_m) = \sum_{j=1}^{n_m} \lambda_j \ , \tag{2.38}$$

where λ_j is the variance in shape mode j and $C(n_m)$ is the compactness using n_m number of modes.

2.6.4 Comparing Quality of Models

For compactness it is common to compare the total variance, i.e. to let $n_m = n_s - 1$, but often we do not want to include all the modes and then this analysis may not be sufficient. It is common to plot the measures over the number of modes used in the model. If the curve of the error-measure over number of modes for one model is below or equal to the curve for another model for all n_m and lower for some n_m, the model represented by the lower curve is often said to be more general or specific depending on the measured quantity [31]. Sometimes, only the total error is considered. A common situation when comparing the quality of two models using the measures described above is that the plotted curves of two models cross each other. It can therefore be hard to choose, which model is actually better.

2.6.5 Discussion

These standard measures have some severe weaknesses which will be analyzed both theoretically and experimentally in this thesis. Also, an alternative model quality measure will be presented.

2.7 Segmentation Using an Active Shape Model

Different versions of segmentation algorithms have been used with Active Shape Models. Since this thesis focuses mostly on the model *building* phase, here only the standard version is applied. This is an iterative approach. After an initial guess the steps in algorithm 2 are iterated.

input : The image and the shape model
output: Optimized model parameters

1 **repeat**
2 Search in the direction of the normal from every landmark to find a suitable point to place the landmark in.
3 Update the parameters for translation, rotation, scale and shape modes to make the best fit to the new points.
4 **until** *convergence*

Algorithm 2: Active Shape Model Segmentation

2.7.1 Getting the Initial Guess

In order to obtain a fast and robust segmentation it is important to have a good initial estimation of the position and orientation of the shape. In the initial guess the shape

is assumed to have the mean shape. This makes it necessary to find values of seven parameters to make a suitable initial guess in three dimensions (three for translation, three for rotation and one for scale). The method to find the initial guess is usually application dependent.

2.7.2 Multi-Resolution Approach for Active Shape Models

To improve the robustness and the speed of the algorithm a multi-resolution approach is used. The idea of multi-resolution is to first search in a coarser image and then change to a more high resolution image when the search in the first image is not expected to improve. This improves the robustness because the amount of noise is less at the coarse level and therefore it is easier to find a way to the right object. The high resolution images are then used to find small structures. The speed increases because there is less data to handle at the coarse levels.

2.7.3 Finding Suitable Points

At every landmark $\mathbf{l}$ in the shape model, the variations of the image appearance is modeled. This model of appearance is later used when searching in the direction of the normal of the landmark to locate its new position.

Modeling Intensity Profiles

In order to model the appearance from the training set, image intensities are sampled at points along the normal of the surface from the landmark in both directions, say in $2k+1$ equidistant points. These intensity values can have a large variation of intensity over the training set. To minimize this effect the derivative of the intensity along the normal is used. The sampled derivatives are put in a vector $\mathbf{g}_i$. These values are then further normalized by dividing with the sum of absolute values of the vector. The resulting vector is sometimes called the profile for the landmark. This is repeated for all surfaces in the training set and gives a set of profiles $\{\mathbf{g}_i\}$ for each landmark in the model. These are assumed to be Gaussian distributed with the mean $\bar{\mathbf{g}}$ and the covariance matrix $\mathbf{S}_g$, calculated from $\{\mathbf{g}_i\}$. This results in a statistical model of the grey level profile at each landmark.

Through the process of manual marking of the objects in the training images and the following surface estimation, small errors in the surface position will inevitably be introduced. This will make the modeled surface not exactly coincide with the true object surface. Because of this, the profiles will be translated a bit which close to image edges will be very noticeable in the image intensities and as a consequence the means and variances of the profiles will be of low quality and the benefit of the profile model will be small. To reduce these problems an edge detection a short distance along the normal to the surface is performed. If the edge detection finds a suitable edge the landmarks are moved to

that position before modeling the image intensity profiles. The edge detection is made by fitting cubic splines to the sampled values in the profile. With aid of the new points that are calculated the edge can be located with sub pixel precision. The edge detection searches for positions where the second derivative is zero. If such a point also has a minimum in the first derivative the point is considered an edge. The detection is made in two steps. First it is done at a coarse level and in a second step the detection is made at the level with the highest resolution.

Getting New Points

When a new point is to be located, while searching in the direction of the normal during segmentation, the quality of the fit is measured by the Mahalanobis distances given by

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \bar{\mathbf{g}})^T \mathbf{S}_g^{-1} (\mathbf{g}_s - \bar{\mathbf{g}}), \tag{2.39}$$

where $\mathbf{g}_s$ is the sample made around the new point candidate. This value is related to the probability that $\mathbf{g}_s$ is drawn from the model. Minimizing $f(\mathbf{g}_s)$ is the same as maximizing the probability that $\mathbf{g}_s$ comes from the distribution and therefore that the point is at the sought-after edge.

To speed up the algorithm only a few of the landmarks are used at the coarse levels. One quarter of the landmarks were kept for every step to a coarser level.

2.7.4 Updating Parameters

When new landmark positions are located the next step is to update the parameters for translation, scale, rotation and shape modes to best fit the new points. This is done by an iterative process. The aim is to minimize

$$\|\tilde{\mathbf{x}} - T_{t,s,\theta}(\bar{\mathbf{x}} + \mathbf{\Phi}\mathbf{b})\|^2, \tag{2.40}$$

where $\tilde{\mathbf{x}}$ is the new points and T is a similarity transformation. The iterative approach as presented by Cootes et al. [28] is given in algorithm 3.

In the segmentation only shapes relatively similar to the shapes in the training set are of interest. Therefore constraints are applied to the $\mathbf{b}$ parameters. Usually those constraints are $|\mathbf{b}_i| \leq 3\sqrt{\sigma_i}$ where σ_i is the eigenvalue corresponding to shape mode i.

input : The Active Shape Model
output: The $\mathbf{b}$-parameters

1 Set the shape parameters $\mathbf{b}$ initially to zero.
2 **repeat**
3 Generate the shape given by the present $\mathbf{b}$.
4 Find the similarity transform that best fits the shape to the new points.
5 Calculate new parameters $\mathbf{b}$.
6 Apply constraints on $\mathbf{b}$.
7 **until** *convergence*

Algorithm 3: Update parameters

Chapter 3
Shape Model Application

For one who has conquered the mind, his mind is the best of friends; but for one who has failed to do so, his mind will be the greatest enemy.

Bhagavad Gita

This chapter presents a complete system for building active shape models and using them for segmentation of medical data and for possible further analysis of that data. Two different types of medical image data are handled. One application is segmentation of the femur in the knee in MR-images for injury analysis and surgery planning. Another application is the segmentation of the brain surface in SPECT images for blood flow analysis and for linkage to other examination data for the purpose of dementia assessment.

For both applications the overall procedure is the same. Several data sets were first segmented manually with the help of medical experts. The resulting structures were represented by unorganized point clouds. Using level set methods, surfaces were fitted to these point clouds. The iterative closest point (ICP) algorithm was then applied to establish correspondences between the different surfaces. Surfaces and correspondences were then used to build a three dimensional statistical shape model using Principal Component Analysis (PCA). The resulting three dimensional Active Shape Models were then used to automatically segment the relevant structures in subsequent data sets for further analysis.

3.1 Introduction

Hospitals today produce large amounts of diagnostic images such as Magnetic Resonance Imaging (MRI), Single Photon Emission Computed Tomography (SPECT), Computed Tomography (CT) and digital mammography. These technologies are a crucial tool in diagnosis and treatment planning. The growing number of images in the medical field have increased the need for computerized tools as a help to find anatomical structures and to make decision support systems. Today almost all analysis of images is still done by manual inspection by the doctors even though most images are digital. Even for an experienced doctor the diagnosis can be hard to state and the process is often time consuming, especially since most images are three dimensional images, which are difficult to view. Model based approaches like Active Shape Models are often used for segmentation since they can handle noisy data, which is often the case with medical images.

Here a system for this will be presented.

3.2 Shape Reconstruction from Point Clouds

In the images in the training set the shapes are manually marked producing a large number of points located at the surface of the object. A typical point cloud for a shape in the femur data set can be seen in Figure 3.1.

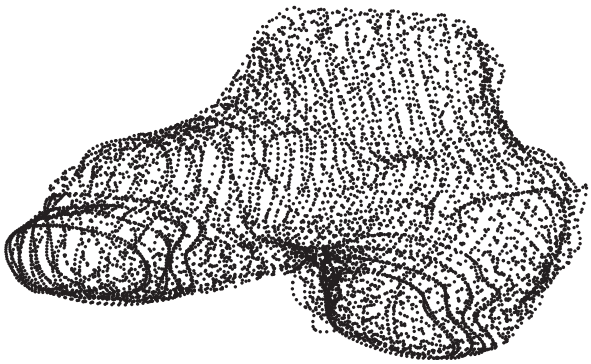


Figure 3.1: Unorganized point cloud of the femur.

From this unorganized point cloud the aim is to get a triangulation of the surface that the points are (approximately) located at. To handle the noisy representation of the shape at this point, a level set approach is used for this surface fitting problem in order to reconstruct the shape. Since this thesis is not about level set methods this is not described in detail here. For descriptions of these methods, see [1, 118, 123, 150, 151]. The idea of the level set method is to represent the surface as an implicit distance function which is zero at the surface and negative inside. The function is then updated to solve a PDE describing the solution of an optimization problem that is constructed in such a way that it will hopefully have a minimum at the true surface. By updating the function iteratively it will fit better and better to the surface of the object.

An example of the result can be seen in Figure 3.2.

3.3 Establishing Correspondences

As we saw in the previous chapter, it is of great importance that during training of an Active Shape Model a dense correspondence is established across the training set. This is the most difficult part and often the most important for a good result of the upcoming segmentation.

The traditional approach to solve this is to manually place landmarks of correspondence at the surface. Another approach for finding correspondence between shapes is

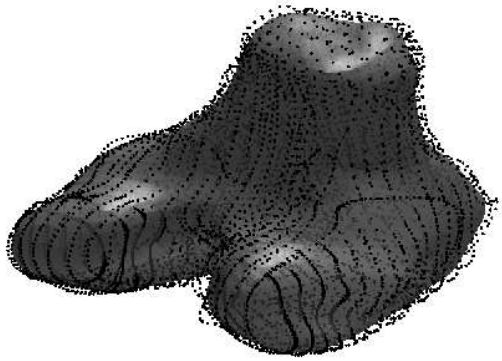


Figure 3.2: The surface fitted to an unorganized point cloud. It is hard to see the points inside the femur and this makes the surface look like it lies inside the points.

to achieve corresponding parameterization of the shapes through optimization of some objective function.

In this chapter the correspondence of points across the training set is established by the Iterative Closest Point (ICP) algorithm [12]. Using the ICP algorithm corresponding triangulations of the structures are achieved.

The ICP algorithm matches two surfaces. It uses one as source surface and one as target. The triangulation of the source is kept fixed and the aim is to get an optimal corresponding triangulation on the target surface. To do this an iterative process is applied with the steps described in algorithm 4.

When convergence is reached the closest points on the target surface are calculated one last time and these points give the new vertices in the triangulation of the target surface.

This algorithm gives two surfaces with corresponding triangulation and each point on the surface can be looked at as a landmark with a corresponding landmark at the other surface. In practice the vertexes can be picked as corresponding sample points. If the source surface is fixed (typically letting the source be the mean shape) and the target surface is interchanged it is possible to find corresponding landmarks in a larger training set. Then a new mean-shape can be calculated and then new correspondences can be calculated, and so on till convergence.

3.4 Building the Shape Model

Next, the shapes in the training set are aligned as discussed in Section 2.3.2. Now the model can be built.

input : The training set of surfaces
output: The corresponding point sets $\mathbf{x}_i$, i = 1,...,n_s on the surfaces

1 **repeat**
2 For each vertex at the source surface find the closest point at the target surface, typically using Euclidean distance.
3 Compute the transformation from the source to the new points, located in the previous step, that minimizes the mean square error between the two point clouds. Typically the set of allowed transformations are similarity transformations (translation, scale and rotation) or rigid (translation and rotation) transformations.
4 Apply the transformation
5 **until** *the improvement of the error between two iterations is less than a threshold value* $\tau > 0$.

Algorithm 4: Iterative Closest Point

Here an active shape model is used, as explained in Section 2.5, so that new shapes can be expressed by the linear model $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{\Phi b}$. With this approach it is possible to constrain the parameters in $\mathbf{b}$ so the new shape always will be accurate.

The model is generated by using Principal Component Analysis (PCA) on the covariance matrix of the training shapes, see Section 2.4.

Cootes et al. propose in [28] a constraint of the b_i parameters of $\pm 3\lambda_i$, where λ_i are the square roots of the eigenvalues of the covariance matrix, to ensure that any new shape is similar to the shapes in the training set. This method will be used here.

Another way to constrain the parameters in the shape model would be to look at the probability that the new shape is from the training set and constrain the shape into a reasonable interval.

3.5 Experiments

The algorithm was used on two data sets, MR images of the knee and SPECT images of the brain.

3.5.1 Modeling Comments

If for example the left knees were mirrored in the left-right direction of the patient it was possible to use both left and right knees at the same time to build the model of knees.

A source to make the result worse was that the images did not cover exactly the same area of the patient in all images. Sometimes that made the surfaces cover a larger part of the femur in the knee images and in the brain images the whole brain was not always

covered in the image. This resulted in some loss of shape information quality at the top of the femur and at the bottom of the brain.

3.5.2 Segmentation

Results for MR Images of the Knee

Generally, the femur was well segmented. When the initial guess was not good enough the model was not able to find the way to the femur. Instead other edges were located that were of no interest (sometimes the edge of the image).

If the initial guess was good enough the search algorithm found the right edges almost every time, in the presence of many local structures, but in some parts of the images the result was not as good. During the segmentation only the sagittal images were used and if the result was visually examined the result looked better in the sagittal view. In Figure 3.3 the result from a segmentation is viewed.

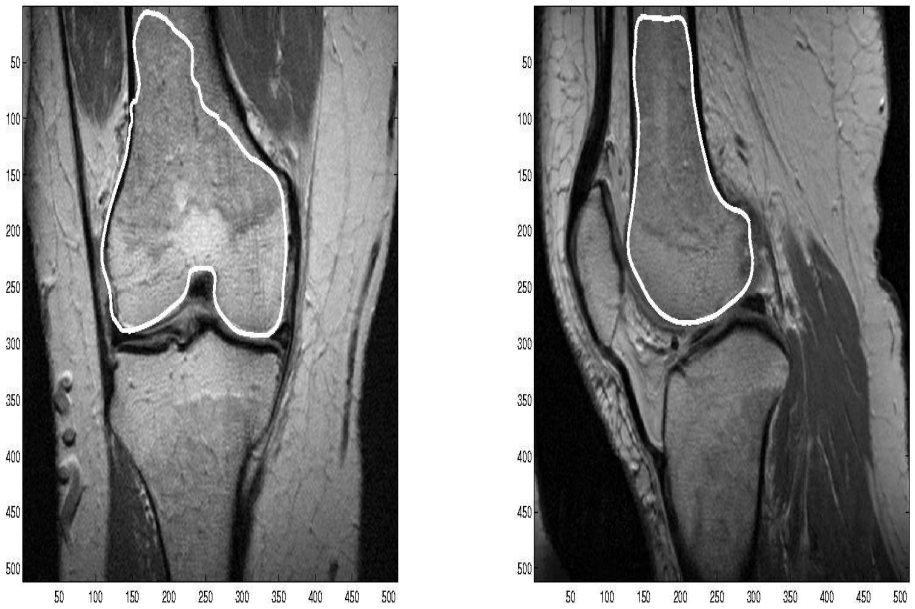


Figure 3.3: An example of the result of the segmentation of the femur in the knee. Note that the image is in 3D and the segmentation was of a surface in 3D, but for visualization 2D slices in two different directions are shown. The shape information in the model is not as good in the upper part as in the lower part, since the coverage of the training images differ. Since it is the shape and segmentation of the lower part that is of most interest anyway, this is not a big problem.

The continuation of this project includes several elements. More parts of the knee are planned to be modeled and linked to the femur model. Strategies for improving the quality of the segmentations further, for example by incorporating level set techniques, should be examined. Finally, the models are to be used both for analyzing shape differences and their link to the risk of later injury and also for planning surgery before opening up the knee.

Results for SPECT Images of the Brain

When the segmentation was done on the SPECT images a very good result was usually obtained. When the algorithm was used on a number of brains which were also manually segmented by an expert and the result was compared it was hard to tell which segmentation was the choice of the computer and which was the experts. In Figure 3.4 the result from a segmentation is viewed. In this example the whole brain is visible, but the algorithm was robust in the sense that it looks the same also when for example the bottom of the brain is cut off. The noise outside the brain does not show up very well in the image, but there are still edges there, only with smaller intensity values.

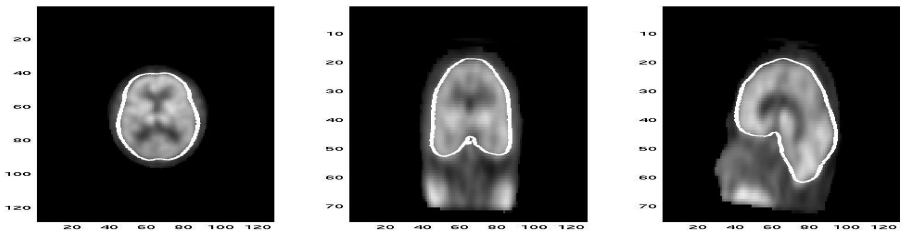


Figure 3.4: The result of the segmentation on a SPECT image of the brain. Note that the image is in 3D and the segmentation was of a surface in 3D, but for visualization 2D slices in three different directions are shown.

This project was continued in the following way. SPECT measures relative blood flow in the whole brain, i.e. we do not get absolute values of blood flow. Since global reduction of blood flow is also an interesting parameter in dementia assessment absolute values of blood flow would be valuable. There is another technique for measuring blood flow in the cortex, the brain surface layer, based on inhalation of Xenon, which measures absolute blood flow using detectors placed on the skull. In this project the long term plan is to link these two examinations to get the best of both worlds, absolute blood flow in the whole brain. In order to do this a number of patients underwent both examinations. After the Xe examination radioactive markers for the position of the Xe detectors were placed on the skull. During the SPECT examination these markers then show up in the images along with the brain. After segmentation of the brain the position of these markers, i.e. the position of the Xe detectors, in relation to the brain model were then registered.

After doing this for a set of patients it is then possible to get a model of where on the brain these detectors statistically speaking measure the blood flow. On later patients the time and work demanding process with the markers are then no longer necessary since after segmenting the brain in the SPECT image the positions of the Xe detectors can be estimated fairly well using the model. This part of the project worked well. Next the relationship between the measured values using the Xe detectors and the intensities in the SPECT images is to be analyzed (on the set of patients where the marker examinations have been performed so that we have the true detector positions) in order to complete the link. Unfortunately, during the time for this thesis work, not enough patient were examined successfully with the special marker procedure to get statistical significance for this link between the different values of blood flow. This remains to be done when more data is available.

3.6 Summary and Conclusions

In this chapter a complete system for segmentation of three dimensional medical images with active shape models is presented. The algorithm has been tested at MR images of knees and SPECT images of the brain. The results are promising, especially in the SPECT images. In the MR images it is harder to find a good initial guess which makes the result less robust and not as good as in the SPECT images. But if the initial guess is good the segmentation algorithm usually gives a good result.

Chapter 4

Segmentation using Integral Approximation

An error doesn't become a mistake until you refuse to correct it.

Orlando A. Battista

Active shape segmentation is a promising technique, but the standard algorithm is sensitive to poor image conditions with many local minima. An alternative approach is to use a generic optimization of the parameters for translation, rotation, scale and shape, using some cost-function to evaluate the segmentation. This has been done by calculating a cost for each landmark and summing up the results. This leads to un-proportional concern for areas with many landmarks. By seeing the cost-function as an integral of a cost measure over the surface in 3D or curve in 2D and then approximating this integral it becomes clear how to deal with this problem. In this chapter it is shown how this is done and also experiments are presented to confirm the value of the addition to the algorithm.

4.1 Introduction

Image analysis is becoming common in many fields, for example in analyzing medical data when diagnosing patients. In these contexts it is often necessary to first find the interesting area in the image, image segmentation. Then the segmented area of the image is analyzed. In both these steps the shape of the area is often important. It is easier to perform the segmentation if you know what shape the area usually has, and once segmented the shape of the area can be analyzed along with the data within the shape.

Using image analysis techniques it is for example possible to give support to medical staff in analyzing the huge amounts of data that modern scanning techniques produce. Improvements both in speed and accuracy can be achieved, especially in the case of three dimensional data that are becoming common and that are hard to view and analyze manually in a natural way (although research and progress are being made here too). Automatic analysis of medical data can also be used as a backup. All available data can be checked and a warning can be given if there are indications that the physician may have missed something.

4.1.1 Active Shape Segmentation

One way of segmenting and analyzing shapes is to use Active Shape Models as described earlier. The standard algorithm for segmenting the shape from an image is as follows.

The training images are sampled at each landmark along lines orthogonal to the surfaces (or curves in 2D). The profiles differentiated and normalized to get a model of the appearance of the image at each landmark. The means and variances of these profiles across the training set is taken as a model of the appearance of the image around each landmark. When segmenting an image, a search is made along lines orthogonal to the surface (or curve in 2D) in its current position and shape. Along the line profiles are sampled, differentiated, normalized and compared to the modeled appearance from the training set. Then the landmark is moved to the point along the line with the best fit. When this has been done for all the landmarks they are probably not in positions that fit the model. Therefore the closest position of the model to the points is calculated and the landmarks are adjusted to fit the model. Typically a maximum of three standard deviations from the mean is allowed for the shape parameters. This finishes one iteration and the procedure is iterated till convergence.

4.1.2 Problems

If all is well this works fine and fast. But all is not always well. If the image quality is poor, as it often is in for example medical image analysis, or if there are many local structures in the image giving rise to many locally optimal positions that according to the cost function are almost as good as the global optimum but in reality are far from correct, there can be trouble. Images of this type are not uncommon. One of the key problems is that the cost function does not really evaluate the position that we end up moving the surface to. This is because of the extra step after the optimization of individual points which we need in order to adapt to the model. If neighboring landmarks have moved irregularly before this adapting takes place we don't really know what we will get when adapting to the model. The new position might not be as good a step towards the optimum as it appeared when viewing one landmark at a time. And neighboring landmarks do move irregularly if the images are of the poor type described above. Also, the algorithm requires a reasonably good initial guess for the landmarks, otherwise the searches along lines orthogonal to the surface will not give good positions for the landmarks. To some extent these problems can be lessened by using a multi scale approach but it does not solve the problems.

4.1.3 Solutions

To overcome these difficulties alternative search strategies can be adopted. A general optimization algorithm can be used to optimize parameters for scale, rotation, translation and shape. The parameters for shape are just the coefficients for the modes of variation in the linear combination of the shape model. We evaluate a set of parameters by applying

them to the shape, sampling the image along the surface, and calculating a cost. To calculate the cost we can sample profiles as before but without the search along the lines. The costs for each landmark are summed up to make the total cost function. This is not optimal since undue weight is given to areas with many sample points on the shape. In this chapter instead of simply summing up the values for each point, an integral over the shape is approximated which gives better robustness to the algorithm. Other things have also been shown to increase robustness. This concerns for example variations of the norm used when calculating the cost [105] and optimizing which sampled features around the landmarks to use (instead of just sampling at even distances along a orthogonal profile line) [54]. There are other strategies, some of which are discussed in [54].

4.2 Modification of Segmentation Search Algorithm

4.2.1 Point of View

We first look at an idealized problem. By this we mean that the surface is simply a regular continuous surface in 3D or a regular continuous curve in 2D, not a set of landmarks or sample points. We want to measure the cost for a position and shape of the surface in the image. This then means integrating a cost function c over the shape S. For the 3Dcase we get the following formula.

$$\text{cost} = \int_S c(s)\, dA(s) \ , \tag{4.1}$$

where dA is an area-measure. Now, to compute this numerically we have to approximate it as the sum

$$\text{cost} = \sum_S c(A_i)\, \Delta A_i \ , \tag{4.2}$$

where A_i is a small part of the surface with area ΔA_i. For the 2Dcase we get the following corresponding formulas:

$$\text{cost} = \int_S c(t)\, dL(t) \ , \tag{4.3}$$

$$\text{cost} = \sum_S c(L_i)\, \Delta L_i \ , \tag{4.4}$$

where dL is a length-measure and L_i is a small part of the curve with length ΔL_i. In practice, the approximation procedure means that we sample the surface or curve at some sample points or landmarks. We then calculate the cost function c at those points. The value of the cost function at a landmark is then seen as representative of the cost function in the vicinity of that landmark, A_i above. From the above equations it is now obvious

that we should not only sum up the costs for each landmark since this only constitutes the c in the equation. We also have to calculate the size of the surface (or curve) element and use this as a weight for c at each landmark.

4.2.2 Computing the Area of the Surface Element

Experiments show that adding this correction is effective even if the calculation of the size of the surface element is not very exact. It can of course be approximated though. In this work it was done by using the approximation of the surface that we get by taking a well defined triangulation between the landmarks. The area around each landmark is then a function of the areas of the neighboring triangles, which are of course easily calculated. To calculate the area of the surface segment corresponding to a certain landmark, each triangle is divided into three parts, where the border of each part is a polygon with one corner at a landmark, two corners at the midpoints of the triangle edges leaving that landmark and one corner in the middle of the triangle. The area associated with a landmark is then a sum of such triangle-parts. This approach is illustrated in Figure 4.1.

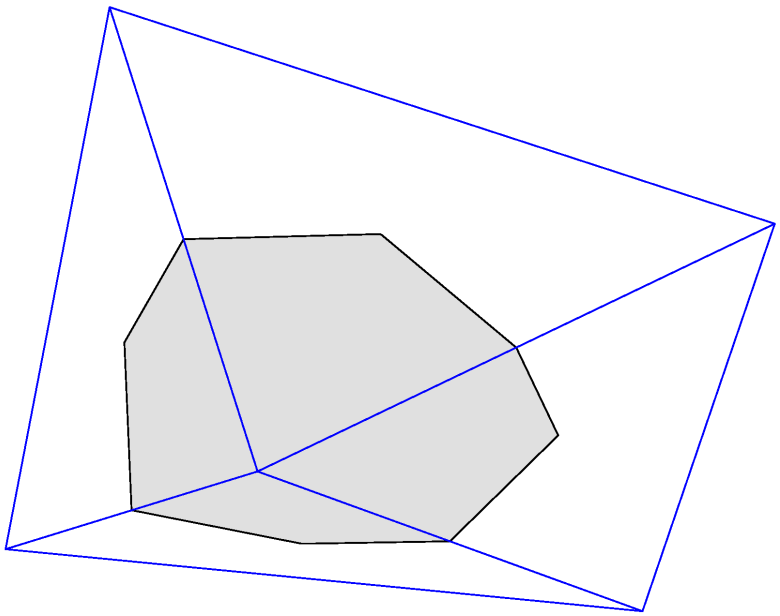


Figure 4.1: Illustration of area of surface segment around a landmark.

A simpler approach, giving the same relative weights, is to just use the sum of the areas of the triangles around the landmark.

An even simpler approximation (not used here) is to use the mean distance to neighboring landmarks. In 2D we want a weight proportional to the length of the curve segment around the landmark, which can be calculated as the sum of the distances to the two neighboring landmarks.

An alternative is to not calculate these weights from the current shape, but instead use weights calculated from the model mean-shape.

Note also that sometimes it is possible to calculate the weights from the parameterization function for the object, since this function determines the density of sample points generated by equidistant parameter values.

4.2.3 Impact

This correction leads to improved performance. This is because the density of landmarks can and often does vary considerably across the surface. This means that without the correction, areas with many landmarks will get an un-proportional influence on the cost function. The cost function is then no longer a good measure of global cost for the entire surface and so the search performance is effected negatively. The experiments show that this effect is clearly visible in real applications. This effect is actually equally true for the standard algorithm searching for new positions for each landmark individually along the normal directions. It does not sum up the costs, but the effect of giving undue weight to areas with many landmarks is still true. It is then coming into effect when calculating how best to adapt the model to the moved landmarks.

4.3 Experiments

The experiments concerned finding the surface of the brain in SPECT images. The algorithm was tested both on healthy patients and on patients with Alzheimers disease. Since the purpose was to study the importance of the correction to the cost function we did not put much effort into optimizing the performance using other factors. One such factor is the active shape model. The training set was small, only six images with marked surfaces. Also, the correspondences of the landmarks across the training set were not optimized very carefully. Another factor is the search optimization. It was not very advanced or run for very long. The experiments show a clear general improvement when using the modified cost function with the integral approximation using the area weights. The improvement generally gets clearer the longer the optimization is run.

4.3.1 The Active Shape Model

Six SPECT images with the surfaces manually marked by an expert were used to build the active shape model and correspondences were optimized using minimum description length. Figure 4.2 is a view of the variation according to one of the modes. The amount of variation is exaggerated for illustration purposes.

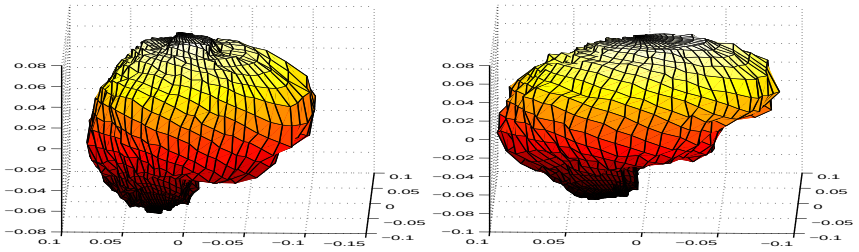


Figure 4.2: Variation according to one mode of the model.

4.3.2 Qualitative Comparison

A typical example of the segmentation result with and without the area weight modification to the cost function is presented in Figure 4.3. This all happens in 3D. The shape is a closed surface and the image is a volume of intensities. To visualize the segmentation three orthogonal slices of the image are shown. The surface is visualized by the curves that are the intersections of the surface with the planes along which the slices of the image are taken. Figure 4.3 shows the result without and with the area weight modification. The improvement is consistent (i.e. other tests looked similar) and clear (see figures). The segmentation is not perfect even with the correction. This is due to the simple active shape model used and the limited optimization procedure.

4.4 Extensions

Not only the function determining the quality of fit can be improved in this way. There are other steps in the algorithm that are computed as sums over landmarks but should really be integrals over the surface and these calculations could also be improved using weights for the landmarks as in this chapter. This includes parameters controlling rotation, translation and scale in the alignment procedure and the calculation of shape parameters. For example, remember that shape parameters are calculated as

$$\mathbf{b} = \mathbf{\Phi}^T(\mathbf{x}' - \bar{\mathbf{x}}) \ , \tag{4.5}$$

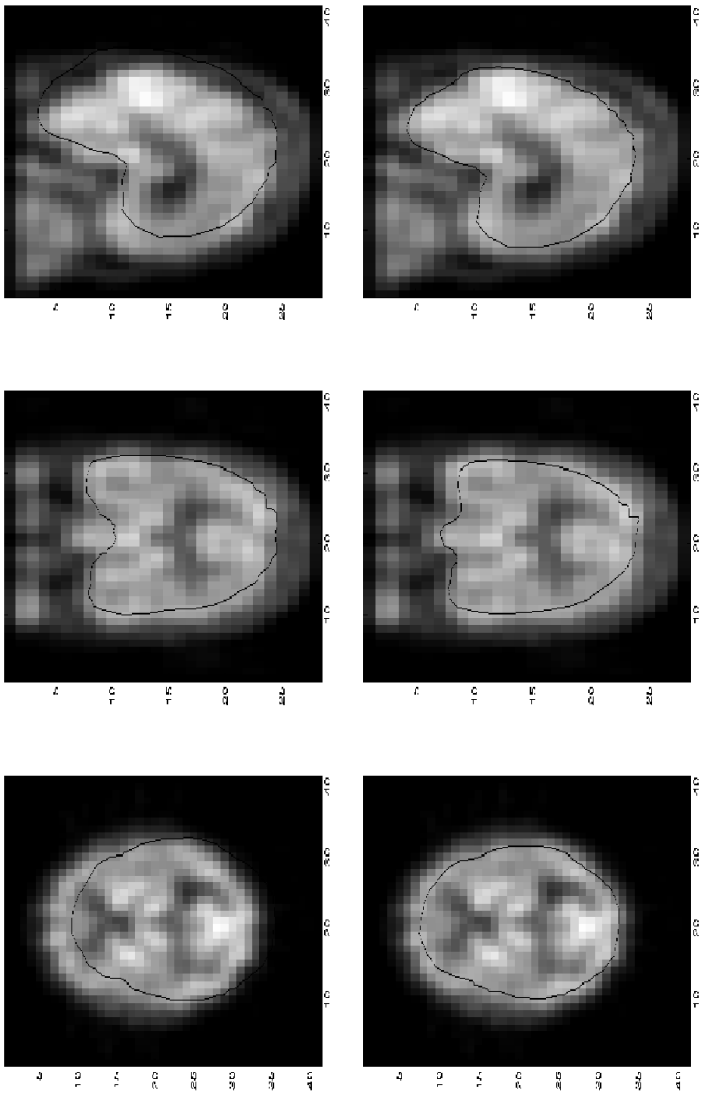


Figure 4.3: Segmentation result with standard formulation to the left and with modified formulation with area weights to the right.

i.e. the parameter for mode i is

$$\mathbf{b}_i = \mathbf{\Phi}_i^T(\mathbf{x}' - \bar{\mathbf{x}}) \ , \tag{4.6}$$

where $\mathbf{\Phi}_i$ is shape mode i which is column i of $\mathbf{\Phi}$. If this sum over points on the shape is interpreted as an approximation of an integral over the shape we can improve the approximation as above by adding weights to the terms. Similarly weights can be added to the appropriate calculations in the alignment step giving parameters for translation, rotation and scale. It is also possible to use these weights for the PCA, by modifying the matrix to be factored using the singular value decomposition. Either each landmark can be multiplied with its area-weight, or alternatively each row of corresponding landmarks can be multiplied by the mean area-weigh.

4.5 Summary and Conclusions

Measuring the cost of a position and shape of the surface should be seen as approximating the integral of a cost function over the surface. This means adding the size of the area element around each landmark as a weight for the cost function at that landmark. This improves the segmentation.

Part II

Automatic and Optimal Construction of Shape Models

Chapter 5
Establishing Correspondences on Object Outlines Using MDL

The path is as sharp as a razor.

The Upanishads

As explained previously, when modeling shape in statistical shape modeling a dense correspondence of good quality between the shapes in the training set needs to be established. Traditionally this has been done by hand, but this is both time consuming and difficult, especially in 3D. Automatic and semi-automatic algorithms for establishing the correspondences have been developed in the last two decades. One successful approach is based on optimizing the minimum description length (MDL) of the training set as described by the model built from the correspondences. As this approach will be used for several things in this thesis it will be briefly described here. More details can be found for example in [31, 41].

5.1 Introduction

Algorithms for automatic shape modeling usually establish dense correspondences by optimizing the parameterizations of the shapes with some suitable objective function for evaluating the model. There have been many different approaches, using different key concepts. Examples include: registration of images or of distance functions from object outlines [30, 108, 138], key landmarks located using features such as curvature extrema and equidistant landmarks in-between [11, 74, 130], geodesics [146], "shape context" [10], polygonal approximation [67], dynamic programming to match features such as curve length and curvature [113], point clustering and nonrigid transformation [22], local neighborhood structure [152], Iterative Closest Point (ICP) [12, 62], determinant of covariance matrix [83], and Minimum Description Length (MDL) [35, 41]. It may be interesting to note that when using registration based methods there is an alternative to using the resulting corresponding coordinates. Instead the deformation field parameters can be analyzed with for example a PCA. The resulting models are then called deformation models instead of shape models [107].

There are a number of different methods for selecting which model is to be considered optimal. Well established are for example AIC (An Information Criterion) and BIC (Bayes Information Criterion) [51, 88]. Another alternative is Ljungs Final Prediction

Error (FPE) [91]. In shape modeling, Minimum Description Length (MDL) has proved to be a successful criterion, and this thesis focuses on MDL.

Minimum Description Length was introduced by Rissanen in [104]. Since then it has been used in different fields, for example in model selection/optimization. Davies et al. introduced MDL as a way of establishing dense correspondences for shape modeling [31] and since then it has been successfully used in different versions in this field.

The minimum description length measure originated in information theory and is, simply put, the minimum amount of data that is needed to code the training set with the aid of the model, using an optimal coding strategy. A good model will be able to code the training set with less data and hence will have a lower description length.

Part of the idea behind using MDL as evaluation function is that it is thought to make a trade-off between a model that is general (can represent any instance from the shape class), a model that is specific (can only represent instances from the correct shape class) and a model that is compact (can represent the shapes with as few parameters as possible and with low variance in the parameter values).

When using MDL for establishing correspondences for shape modeling, some type of optimizer will typically search for an optimal set of parameterization functions γ_i $(i = 1, \ldots, n_s)$ for the shapes in the training set. The function used to evaluate a set of parameterization functions then consists of building a model from the correspondences produced by the parameterizations and calculating the description length for the training set using the model.

5.2 The Minimum Description Length

5.2.1 Preliminaries

In this section the minimum description length for a training set using a shape model will be presented. The formulas to be used will be stated and briefly explained. For more detailed derivation and analysis see [31, 41].

What is given is a training set of n_s shapes, each sampled in n_p points in n_d dimensions, giving the shape vectors $\mathbf{x}_i (i = 1, \ldots, n_s)$, gathered as columns in $\mathbf{X}$ efter the mean-shape is subtracted. A linear model for approximating shapes as

$$\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{\Phi}\mathbf{b} = \bar{\mathbf{x}} + \sum_{j=1}^{n_m} \mathbf{\Phi}_j b^j \ , \tag{5.1}$$

where $\mathbf{\Phi}_j$ are the shape modes (eigenvectors of $\mathbf{X}\mathbf{X}^T$), b^j are the shape space coordinates and n_m is the number of shape modes, is to be built.

Since MDL is to be used for finding the optimal parameterizations, the parts of the description length that are constant for different sets of parameterization functions can be ignored during optimization. This includes things like the description length of the mean

shape $\bar{\mathbf{x}}$. Some parts of the total description length do not depend directly on the parameterization functions, but do depend on the number of modes included in the model. This includes things like the description length of the eigenvectors $\mathbf{\Phi}_j$. In this section, all modes will be included in the total description length so the description length of $\mathbf{\Phi}_j$ is constant and can be ignored during optimization of the parameterization functions. Later in the thesis the number of modes will not be constant, and the description length of $\mathbf{\Phi}_j$ will be included in the total description length during optimization. When using all modes, the shapes in the training set can be described exactly, but when not using all modes they will only be approximated and then also the residuals need to be coded and included in the total description length. This will be done later, when not all modes are necessarily used during optimization. For now, using all modes and ignoring the constant parts, what is left is the description length of the shape parameters b_i^j.

The coordinates b_i^j of the training shapes $\mathbf{x}_i$ are

$$b_i^j = (\mathbf{\Phi}_j)^T(\mathbf{x}_i - \bar{\mathbf{x}}), (i = 1, \ldots, n_s,\ j = 1, \ldots, n_m) \ . \tag{5.2}$$

Since the shape modes $\mathbf{\Phi}_j$ are orthogonal, the total description length $\mathcal{L}$ of the set of all b_i^j, can be calculated as a sum

$$\mathcal{L} = \sum_{j=1}^{n_m} \mathcal{L}_j \ , \tag{5.3}$$

where $\mathcal{L}_j$ is the description length of the 1D-dataset $\mathcal{Y} = \{b_1^j, \ldots, b_{n_s}^j\}$.

Shannon Minimum Description Length

The data is modeled as a set of one-parameter Gaussians (mean zero). After Procrustes alignment this is usually a reasonable model of the data. Using Shannon's minimal codeword length [115, 116], given a probability density $\mathcal{P}$, the description length of a given value, $\hat{\alpha}$, is

$$-\log\left(\mathcal{P}(\hat{\alpha})\right) \ . \tag{5.4}$$

In order to calculate the description length in bits the logarithms should really be calculated in base 2. To simplify calculations the natural logarithm can be used instead, which gives the description length in so called nats. If desired for some reason, this can of course easily be converted to bits.

Quantization

It requires an infinite amount of information (bits) to describe a real number to arbitrary accuracy. So, before considering the description length, the data must be discretized. A

real number y, described with an accuracy δ (a positive real number), is represented using the integer N, that solves

$$\min_{N\in\mathbb{Z}} |y - \delta N| \ , \tag{5.5}$$

giving the approximation $y \approx \hat{y} = \delta N$.

The constant δ should be chosen in relation to the estimated noise of the training data.

Data Range

Besides the discretization accuracy δ, the range of the data must also be determined. Assume that the shape vectors

$$\mathbf{x}_i = \left[x_i^1, \ldots, x_i^{n_d}, x_i^{n_d+1}, \ldots, x_i^{n_d n_p}\right]^T \tag{5.6}$$

in the original space, $\mathbb{R}^{n_d n_p}$, are bounded so that

$$-r \leq x_i^j - \bar{\mathbf{x}}^j \leq r, \qquad (j = 1, \ldots, n_p n_d,\ i = 1, \ldots, n_s) \ . \tag{5.7}$$

In shape space, where each shape is a vector, this corresponds to the range $R^2 = r^2 + r^2 + \ldots + r^2 = n_p n_d r^2$ so that $||\mathbf{x}_i - \bar{\mathbf{x}}|| \leq R = r\sqrt{n_p n_d}$, which in turn gives,

$$|b_i^j| = |(\mathbf{\Phi}_j)^T(\mathbf{x}_i - \bar{\mathbf{x}})| \leq |\mathbf{\Phi}_j||\mathbf{x}_i - \bar{\mathbf{x}}| = 1|\mathbf{x}_i - \bar{\mathbf{x}}| \leq R \tag{5.8}$$

for all $i = 1, \ldots, n_s,\ j = 1, \ldots, n_m$.

5.2.2 MDL of a 1D-dataset

In this section the minimum description length of the discretized 1D-datasets $\mathcal{Y} = \{\hat{y}_1, \ldots, \hat{y}_{n_s}\}$ (containing the b_i^j), where $\hat{y}_i$ is Gaussianly distributed with mean zero, will be described. The distribution is described by the density function

$$f(x;\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{x^2}{2\sigma^2}\right)} \ . \tag{5.9}$$

An estimation s of σ from $\mathcal{Y} = \{\hat{y}_1, \ldots, \hat{y}_{n_s}\}$ is,

$$s = \sqrt{\frac{1}{n_s}\sum_{i=1}^{n_s} \hat{y_i}^2} \ , \tag{5.10}$$

or after discretization to accuracy δ, $\hat{s} = n\delta, n \in \mathbb{N}$. The probability of x being in the interval $\hat{y} \pm \delta/2$ is then

$$\mathcal{P}(\hat{y};\hat{s}) = \int_{\hat{y}-\frac{\delta}{2}}^{\hat{y}+\frac{\delta}{2}} f(x;\hat{s})dx \ , \tag{5.11}$$

which, if $\hat{s}$ is not to small [31], can be approximated well by

$$\mathcal{P}(\hat{y};\hat{s}) \approx \frac{\delta}{\hat{s}\sqrt{2\pi}} e^{\left(-\frac{\hat{y}^2}{2\hat{s}^2}\right)} \quad . \tag{5.12}$$

Using a two-part coding scheme the minimum description length is decomposed as

$$\mathcal{L} = \mathcal{L}_{param} + \mathcal{L}_{data} \quad , \tag{5.13}$$

where $\mathcal{L}_{param}$ is the description length of the parameters to be used for the Gaussian encoding and $\mathcal{L}_{data}$ is the description length of the b_i^j using the Gaussians.

Coding the Parameters

The variance of the Gaussian needs to be coded. It must be discretized using the accuracy δ. Without prior knowledge, the standard deviation $\hat{s}$, is assumed to have the same range as the data, $s_{max} = R$. If s is further assumed to be uniformly distributed in this range, the description length is

$$\mathcal{L}_{param} = \mathcal{L}_{\hat{s}} = -\log \mathcal{P}(\hat{s}) = \log(\frac{R}{\delta}) \quad . \tag{5.14}$$

The accuracy δ must also be coded but this will give a constant since it does not depend on the correspondences so this is ignored. Actually, as seen above, $\mathcal{L}_{\hat{s}}$ will also be independent of optimization choices for a given training set (assuming a constant δ), so $\mathcal{L}_{param}$ will not have to be considered during optimization.

It is possible to try to choose an optimal δ for the minimum description length calculations to follow. However, one may argue that δ should be chosen as a constant based on the desired accuracy for the application. Also, the description-length-optimal δ can have negative consequences such as loosing continuity of the derivative of the description length. In this thesis δ will be chosen as a suitable constant. For more details see [41].

Coding the Data

Depending on the actual range and standard deviation s of the current dataset there are three different encoding strategies. If $\hat{s}$ is not to small, a simple approximation of the Shannon description length can be used. Define some small s_{cut} to be the cutoff for when to use the approximation. A standard choice is to put $s_{cut} = 2\delta$. If $\hat{s} \leq s_{cut}$, the approximation formula is not valid and $\hat{s}$ is replaced with s_{cut}. If the range of $\mathcal{Y} \leq \delta$ the data can simply be coded as zeros and the description length is zero.

Case 1 ($\hat{s} > s_{cut}$) The minimum description length of the data set consisting of the parameters for the whole training set in one of the modes is

$$\mathcal{L}_{data} = -\sum_{i=1}^{n_s} \log(\mathcal{P}(\hat{y_i})) = -\sum_{i=1}^{n_s} \log(\frac{\delta}{\hat{s}\sqrt{2\pi}} e^{\left(-\frac{\hat{y}^2}{2\hat{s}^2}\right)}) =$$
$$-n_s \log\delta + \frac{n_s}{2}\log(2\pi\hat{s}^2) + \frac{1}{2\hat{s}^2}\sum_{i=1}^{n_s} {\hat{y_i}}^2 = -n_s \log\delta + \frac{n_s}{2}\log(2\pi\hat{s}^2) + \frac{n_s s^2}{2\hat{s}^2} \ .$$

Case 2 ($\hat{s} \leq s_{cut}$) Replacing $\hat{s}$ with s_{cut} gives

$$\mathcal{L}_{data} = -n_s \log\delta + \frac{n_s}{2}\log(2\pi s_{cut}^2) + \frac{n_s s^2}{2{s_{cut}}^2} \ .$$

Case 3 When the range of $\mathcal{Y} \leq \delta$ we get

$$\mathcal{L}_{data} = 0 \ .$$

Before putting together the total minimum description length we make one more simplification. Assuming δ is small, we can replace $\hat{s}$ with s, giving the following descriptions lengths (the indexes (1) and (2) referring to case 1 and case 2 respectively).

$$\begin{aligned} \mathcal{L}_{data(1)} &= -n_s \log(\delta) + \frac{n_s}{2}\log(2\pi) + n_s \log(s) + \frac{n_s}{2} \ , \\ \mathcal{L}_{data(2)} &= -n_s \log(\delta) + \frac{n_s}{2}\log(2\pi) + n_s \log(s_{cut}) + \frac{n_s}{2}(\frac{s}{s_{cut}})^2 \ . \end{aligned} \tag{5.15}$$

5.2.3 The Total Description Length

Putting the description lengths $\mathcal{L}_j$ in the different shape modes together as shown in Equation (5.3) and adding the description lengths of the parameters and the data as shown in Equation (5.13) gives the total description length

$$\mathcal{L}_{total} = \sum_{j=1}^{n_m} \mathcal{L}_j = F_0(n_s, R, \delta) + \sum_{s_j > s_{cut}} \left(n_s \log(s_j) + \frac{n_s}{2}\right) +$$
$$\sum_{s_j \leq s_{cut},\ range(\mathcal{Y}_j) \geq \delta} \left(n_s \log(s_{cut}) + \frac{n_s}{2}(\frac{s_j}{s_{cut}})^2\right) \ .$$

Since δ is normally chosen quite small, in many real applications case 3 will be rare and even when it does occur, usually the corresponding s_j will be small indeed so incorporating case 3 into case 2 will usually not change the value of the objective function noticeably. This results in the simpler objective function

$$F_0(n_s, R, \delta) + \sum_{s_j > s_{cut}} \left(n_s \log(s_j) + \frac{n_s}{2} \right) + \sum_{s_j \leq s_{cut}} \left(n_s \log(s_{cut}) + \frac{n_s}{2} (\frac{s_j}{s_{cut}})^2 \right) =$$

$$F(n_s, R, \delta) + \frac{n_s}{2} \left(\sum_{s_j > s_{cut}} \left(\log(\frac{s_j}{s_{cut}})^2 + 1 \right) + \sum_{s_j \leq s_{cut}} (\frac{s_j}{s_{cut}})^2 \right) =$$

$$F(n_s, R, \delta) + \frac{n_s}{2} \left(\sum_{\lambda_j > \lambda_c} \left(\log(\frac{\lambda_j}{\lambda_c}) + 1 \right) + \sum_{\lambda_j \leq \lambda_c} \frac{\lambda_j}{\lambda_c} \right) \ ,$$

where $F_0(n_s, R, \delta)$ and $F(n_s, R, \delta)$ are functions of n_s, R and δ that are constant with respect to parameterizations. Also, the eigenvalues λ_i of the covariance matrix (2.24) are equal to s_j^2 , so the substitutions $s_i^2 = \lambda_i$ and $s_{cut}^2 = \lambda_c$ have been made.

While optimizing the parameterizations to find a minimum of this function, the term $F(n_s, R, \delta)$ is constant and can be ignored. It is also clear that skipping the factor $\frac{n_s}{2}$ will not change anything so this can also be ignored, resulting in the objective function

$$\mathcal{L} = \sum_{\lambda_j > \lambda_c} \left(\log(\frac{\lambda_j}{\lambda_c}) + 1 \right) + \sum_{\lambda_j \leq \lambda_c} \frac{\lambda_j}{\lambda_c} \ . \tag{5.16}$$

The original version of MDL is not continuous. Therefore an additional step involving numerical integration is introduced in [31]. However the simplified version above is continuous so integration is not necessary.

5.3 Correspondence Optimization

5.3.1 Parameterization Optimization

As mentioned earlier, correspondences are usually optimized by optimizing the parameterizations of the shapes. To illustrate the optimization of the parameterizations, optimization of the parameterizations of shapes defined by planar curves will be described. The basic principles are the same for 3D-curves, surfaces, or other parameterized shapes, but the details can be different depending on the type of shapes to be modeled.

A set of parameterized geometric objects, represented by continuous curves with associated parameterization functions makes up the training set:

$$\mathbf{c}_i : [0,1] \ni s \rightarrow \mathbf{c}_i(s) \in \mathbb{R}^2 \ \ , i = 1, \ldots, n_s \ . \tag{5.17}$$

For simplicity, assume that every object is parameterized by arc length parameterization to start with. A model such as in (2.29) of the shape of the curves in this training set is to be built. As described earlier it is first necessary to solve the correspondence problem. This means determining a set of reparameterization functions $\{\gamma_i\}_{i=1}^{n_s}$ (where $\gamma_i : [0,1] \rightarrow [0,1]$ are strictly increasing bijective functions), such that $\mathbf{c}_i(\gamma_i(s))$ corresponds to $\mathbf{c}_j(\gamma_j(s))$ for all pairs (i,j) and all parameter values $s \in [0,1]$. Remember that correspondence means that if, for example, the shapes describe an anatomical object and $\mathbf{c}_i(\gamma_i(s))$ is at a certain anatomical point on shape i then $\mathbf{c}_j(\gamma_j(s))$ should be at the corresponding anatomical point on shape j. To shorten notation, correspondence between the shapes $\mathbf{c}_i(\gamma_i(s))$ and $\mathbf{c}_j(\gamma_j(s))$ can be denoted $\mathbf{c}_i(\gamma_i(s)) :=: \mathbf{c}_j(\gamma_j(s))$.

Now, some optimizer will try to find optimal reparameterization functions $\{\gamma_i\}_{i=1}^{n_s}$. At each iteration in the optimization process, in order to evaluate the MDL-objective-function, the shapes will be sampled in corresponding points according to the $\{\gamma_i\}_{i=1}^{n_s}$, a model will be built and the description length will be calculated. Note that it is not the case that a set of corresponding landmarks are placed on the curves and the optimizer moves them around to optimize minimum description length. Working with a fixed set of landmarks that are moved around leads more easily to problems with undesired clustering of these landmarks. See Figures 5.1 and 5.2 for a simple illustration.

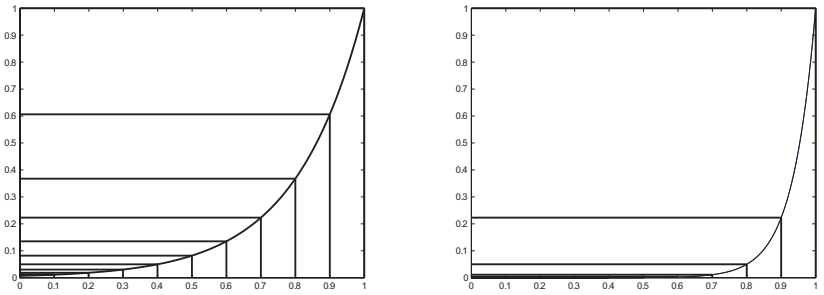


Figure 5.1: In the two plots it can be seen how the sampling points, defined by equidistant parameter values in [0,1], are mapped by two parameterization functions. The plots show how the sampling points can cluster during optimization of the parameterisation functions. In this example they cluster at the beginning of the curve.

This problem will be described and handled in detail later in the thesis. The sampling that is performed is only done in order to perform numerical computations and the set of sample points at one instance does not correspond to the set of sample points at another instance. With continuous objects and parameterization functions all points on the objects are landmarks but for numerical computations this infinite set must be sampled.

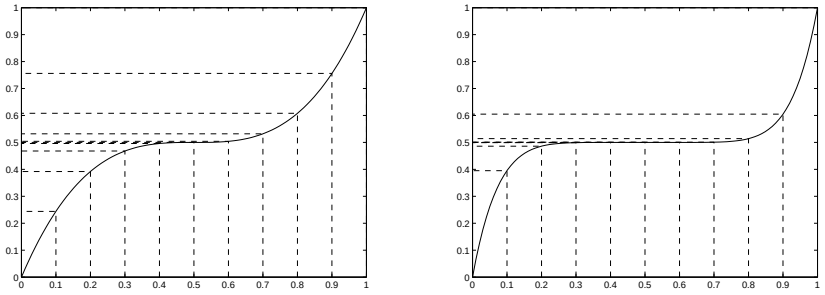


Figure 5.2: In the two plots it can be seen how the sampling points, defined by equidistant parameter values in [0,1], are mapped by two parameterization functions. The plots show how the sampling points can cluster during optimization of the parameterisation functions. In this example they cluster at the middle part of the curve.

The reparameterization functions $\{\gamma_i\}_{i=1}^{n_s}$ are controlled by a set of variables (typically controlling some basis functions) and it is these variables that the optimizer modifies.

5.3.2 The Parameterization Functions

The parameterization functions can be built up by different types of basis functions, typically on different levels of detail. Let $\gamma_i(t)$ be built up by the basis functions $p_{kl}(t)$ as

$$\gamma_i(t) = t + \sum_{k=1}^{n_l} \sum_{l=1}^{2^{(k-1)}} n_{kl} p_{kl}(t) \ , \tag{5.18}$$

where the scalars n_{kl} are the variables to be optimized. Different alternatives of the type of basis functions have different advantages. In this thesis the choice of basis functions is not the main focus so a simple set of basis functions sometimes called tent-functions, giving piecewise linear parameterization functions, will typically be used.

The basic tent-function is

$$g(t) = \begin{cases} t, \ 0 \leq t < 1/2 \\ 1-t, \ 1/2 \leq t \leq 1 \end{cases} \ . \tag{5.19}$$

The basis functions will be versions of this function that are translated, scaled and of different width. The width of the basis function at level k is $w = 1/2^{(k-1)}$ and the basis

functions are

$$p_{kl}(t) = \begin{cases} 0, & 0 \leq t \leq w(l-1) \\ w \cdot g((t - w(l-1))/w), & w(l-1) \leq t \leq wl \\ 0, & wl \leq t \leq 1 \end{cases} \tag{5.20}$$

To simplify notation, the double index kl is replaced by the single index l (i.e. not the same l) and instead an index i for shape object id is introduced to obtain

$$\gamma_i(t) = t + \sum_{l=1}^{n_n} n_{il} p_{il}(t) \ . \tag{5.21}$$

Note that the strictly increasing property of the parameterization functions must be preserved during optimization, usually be restricting the possible values for the parameterization function control variables n_{il}.

5.4 The Gradient of the Minimum Description Length

Many optimizers use the gradient of the objective function. How to do this for MDL optimization will be briefly explained here. For details, see [4, 41, 44, 101].

The key result needed is the derivative of the singular values in $\mathbf{S}$ with respect to the elements of the matrix $\mathbf{A}$ where $\mathbf{A}$ has the singular value decomposition $\mathbf{A} = \mathbf{USV}^T$. These are given by

$$\frac{\partial s_k}{\partial a_{ij}} = u_{ik} v_{jk} \ , \tag{5.22}$$

where s_k is the k-th singular value, a_{ij} is the element on row i and column j of $\mathbf{A}$, u_{ik} is the element on row i and column k of $\mathbf{U}$ and v_{jk} is the element on row j and column k of $\mathbf{V}$.

Let each parameterization function γ_j ($j = 1, \ldots, n_s$) have n_n control variables, with control variable l for curve j being denoted n_{jl}. Assume the objective function is

$$\mathcal{L} = \sum_{\lambda_k > \lambda_c} \left(\log(\frac{\lambda_k}{\lambda_c}) + 1 \right) + \sum_{\lambda_k \leq \lambda_c} \frac{\lambda_k}{\lambda_c} \ . \tag{5.23}$$

Differentiation gives

$$\frac{\partial \mathcal{L}}{\partial n_{jl}} = \sum_{\lambda_k > \lambda_c} \frac{1}{\lambda_k} \frac{\partial \lambda_k}{\partial n_{jl}} + \sum_{\lambda_k \leq \lambda_c} \frac{1}{\lambda_c} \frac{\partial \lambda_k}{\partial n_{jl}} = \sum_{\lambda_k > \lambda_c} \frac{1}{\lambda_k} \frac{\partial s_k^2}{\partial n_{jl}} + \sum_{\lambda_k \leq \lambda_c} \frac{1}{\lambda_c} \frac{\partial s_k^2}{\partial n_{jl}} = \sum_{\lambda_k > \lambda_c} \frac{2 s_k}{\lambda_k} \frac{\partial s_k}{\partial n_{jl}} + \sum_{\lambda_k \leq \lambda_c} \frac{2 s_k}{\lambda_c} \frac{\partial s_k}{\partial n_{jl}} \tag{5.24}$$

Let x_{ij} be the i-th sample point coordinate on shape j and $\frac{\partial x_{ij}}{\partial n_{jl}}$ be the derivative with respect to parameterization function control variable n_{jl}. Using Equation (5.22) this gives

$$\frac{\partial s_k}{\partial n_{jl}} = \sum_i \frac{\partial s_k}{\partial x_{ij}} \frac{\partial x_{ij}}{\partial n_{jl}} = \sum_i u_{ik} v_{jk} \frac{\partial x_{ij}}{\partial n_{jl}} \quad . \tag{5.25}$$

Finally, even though $\frac{\partial x_{ij}}{\partial n_{jl}}$ can somewhat be approached analytically, it is typically estimated using differential approximation.

Chapter 6
Aligning Shapes by Optimizing MDL

> *It is assumed that the phenomena which we now accept without surprise do not excite our astonishment because they are understood, but this is not the case. If they do not surprise us, it is not because they are understood. It is because they are familiar.*
>
> *Charles Robert Richet*

As described earlier, shape is defined to be invariant to similarity transformations, so before modeling shape as described earlier it is first necessary to normalize the objects with respect to similarity transformations. This is normally done using Procrustes analysis as described in Section 2.3.

Instead of Procrustes, the Minimum Description Length (MDL) from Chapter 5 can be used also for the alignment, i.e. align the shapes using similarity transformations in order to optimize the minimum description length of the set of shapes using the model constructed giving the current alignment. This is used already in [31], but here the effects of the different alignment strategies are more carefully analyzed and also the gradient of the description length to be used in Gauss-Newton alignment optimization is stated and derived.

The MDL based algorithm is compared to Procrustes on a number of data sets. Apart from the obvious advantage of using the same measure for alignment and reparameterization, it is also concluded that there is improvement in generalization when using Minimum Description Length to align the shapes. Often the difference is small, but a synthetic example shows that the Procrustes alignment can fail significantly where the proposed MDL method does not. This example shows that the alignment using minimum description length can give more intuitive results and give much better generality than when using Procrustes.

In [86] shapes are aligned by minimizing L_1 distances instead of squared distances. This results in positive effects similar to the ones presented here but the problem of alternatingly optimizing two different measures of course remains.

6.1 Introduction

The common way to align shapes before building shape models is Procrustes analysis [38, 56], i.e. minimizing the sum of squared distances between corresponding sample points from the mean shape to the shapes in the training set over similarity transformations. Of

course, similarity transformations can be filtered out in many other ways. For example, in [16, 38, 79, 122] so called Bookstein and Kendall coordinates are used. These alignment strategies are not as intuitively optimal in any sense as Procrustes analysis.

In the previous chapter it was shown how MDL can be used to locate correspondences. However, this means that different objective functions are used in different stages of the optimization (sum of squared distances during alignment and MDL during reparameterization), which can cause problems for the optimizer. It makes more sense to do the alignment using the same goal function that is being used for the parameterization evaluation step, so that the goal function is optimized with respect to parameterization and alignment.

When the gradient of an objective function is known for a specific optimization problem, it generally pays off to use optimization techniques using the gradient. In this chapter MDL will be used to align shapes, using gradient information. The Description Length is here minimized using Gauss-Newton. In order to do this the derivative of the description length with respect to rotation is derived.

In principle, optimization over similarity transformation includes translations, scaling and rotation. In practice however, translation is well dealt with simply by translating the mean to the origin and scale is usually fairly well dealt with by normalizing to size one, so the main issue is typically rotation. Moreover, it has been observed that when it comes to scale and translation, Procrustes analysis and MDL alignment give very similar results, but for rotation the alignments are different. For these reasons this chapter focuses on rotation. Of course, adding small adjustments in translation and scale according to MDL may improve the result further.

6.2 Optimizing MDL using the Gradient

Let n_s shapes $\mathbf{x}_1, \ldots, \mathbf{x}_{n_s}$ in complex coordinates be centered at the origin and with the scale normalized so that the Euclidean norm is one. In complex coordinates a centered shape $\mathbf{x}_j$ is rotated by θ_j to $\mathbf{x}_j e^{i\theta_j}$. Differentiating the objective function

$$\mathcal{L} = \sum_{\lambda_k > \lambda_c} \left(\log(\frac{\lambda_k}{\lambda_c}) + 1 \right) + \sum_{\lambda_k \leq \lambda_c} \frac{\lambda_k}{\lambda_c} \tag{6.1}$$

with respect to θ_j, we get

$$\frac{\partial \mathcal{L}}{\partial \theta_j} = \sum_{\lambda_k > \lambda_c} \frac{1}{\lambda_k} \frac{\partial \lambda_k}{\partial \theta_j} + \sum_{\lambda_k \leq \lambda_c} \frac{1}{\lambda_c} \frac{\partial \lambda_k}{\partial \theta_j} \quad , \tag{6.2}$$

so what is needed is $\frac{\partial \lambda_k}{\partial \theta_j}$. Let the j-th column of $\mathbf{X}_0$ be the sampling points on shape j in complex coordinates after rotation by θ_j,

$$\mathbf{X}_0 = \left[\begin{array}{cccc} \mathbf{X}_1, & \ldots, & \mathbf{X}_{n_s} \end{array} \right] = \left[\begin{array}{cccc} \mathbf{x}_1 e^{i\theta_1}, & \ldots, & \mathbf{x}_{n_s} e^{i\theta_{n_s}} \end{array} \right] . \tag{6.3}$$

Let $\mathbf{Y}$ be the matrix holding the deviations from the mean shape,

$$\mathbf{Y} = \mathbf{X}_0 - \overline{\mathbf{X}} \quad , \tag{6.4}$$

where each column in $\overline{\mathbf{X}}$ is the mean shape $\bar{\mathbf{x}}$,

$$\bar{\mathbf{x}} = \frac{1}{n_s}\sum_{j=1}^{n_s}\mathbf{X}_j = \frac{1}{n_s}\sum_{j=1}^{n_s}\mathbf{x}_j e^{i\theta_j} \quad . \tag{6.5}$$

A singular value decomposition of $\mathbf{Y}$ gives us $\mathbf{Y} = \mathbf{USV}^T$. Here $\mathbf{U}$ corresponds to $\mathbf{\Phi}$ in Equation (2.29) and the diagonal matrix $\mathbf{S}^T\mathbf{S}$ holds the eigenvalues λ_k.

Now, if y_{pq} is the p-th sample point on shape q and $\frac{\partial y_{pq}}{\partial \theta_j}$ is the derivative with respect to the rotation θ_j then

$$\frac{\partial \lambda_k}{\partial \theta_j} = \frac{\partial s_k^2}{\partial \theta_j} = 2s_k\frac{\partial s_k}{\partial \theta_j} = 2s_k\sum_{pq}\frac{\partial s_k}{\partial y_{pq}}\frac{\partial y_{pq}}{\partial \theta_j} = 2s_k\sum_{pq}u_{pk}v_{qk}\frac{\partial y_{pq}}{\partial \theta_j} \quad . \tag{6.6}$$

Here it is used that $\frac{\partial s_k}{\partial y_{pq}} = u_{pk}v_{qk}$, where u_{pk} and v_{qk} are elements in $\mathbf{U}$ and $\mathbf{V}$, see Section 5.4. Moreover,

$$\frac{\partial y_{pq}}{\partial \theta_j} = \frac{\partial \mathbf{x}_q(p)e^{i\theta_q}}{\partial \theta_j} - \frac{1}{n_s}\sum_{l=1}^{n_s}\frac{\partial \mathbf{x}_l(p)e^{i\theta_l}}{\partial \theta_j} = \begin{cases} i(1-\frac{1}{n_s})\mathbf{x}_j(p)e^{i\theta_j} & q = j \\ -i\frac{1}{n_s}\mathbf{x}_j(p)e^{i\theta_j} & q \neq j \end{cases} , \tag{6.7}$$

$$\text{since} \quad \frac{\partial \mathbf{x}_q(p)e^{i\theta_q}}{\partial \theta_j} = \begin{cases} i\mathbf{x}_j(p)e^{i\theta_j} & q = j \\ 0 & q \neq j \end{cases} \quad . \tag{6.8}$$

To simplify the process, if n_s is relatively large, this can be approximated by

$$\frac{\partial y_{pq}}{\partial \theta_j} = \begin{cases} i\mathbf{x}_j(p)e^{i\theta_j} & q = j \\ 0 & q \neq j \end{cases} \quad . \tag{6.9}$$

Then $\frac{\partial \lambda_k}{\partial \theta_j}$ can be written as

$$\frac{\partial \lambda_k}{\partial \theta_j} = 2s_k i v_{jk}\mathbf{U}_k^T\mathbf{x}_j e^{i\theta_j} = 2s_k i v_{jk}\mathbf{U}_k^T\mathbf{X}_j \quad , \tag{6.10}$$

where $\mathbf{U}_k$ is the k-th column in $\mathbf{U}$. Finally this gives the gradient information

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta_j} &= \sum_{\lambda_k > \lambda_c}\frac{1}{\lambda_k}2s_k i v_{jk}\mathbf{U}_k^T\mathbf{x}_j e^{i\theta_j} + \sum_{\lambda_k \leq \lambda_c}\frac{1}{\lambda_c}2s_k i v_{jk}\mathbf{U}_k^T\mathbf{x}_j e^{i\theta_j} \\ &= \sum_{\lambda_k > \lambda_c}\frac{1}{\lambda_k}2s_k i v_{jk}\mathbf{U}_k^T\mathbf{X}_j + \sum_{\lambda_k \leq \lambda_c}\frac{1}{\lambda_c}2s_k i v_{jk}\mathbf{U}_k^T\mathbf{X}_j \quad . \end{aligned} \tag{6.11}$$

6.3 Experimental Validation

The experiments are done on 2D-shapes, but the principles are the same for any dimension.

Experiments were conducted as follows. Given a dataset, the center of gravity was moved to the origin for all shapes and scale was normalized to one according to the Euclidean norm. To make the comparison between Procrustes- and MDL-alignment easier, the rotation of the shapes was initialized according to the Procrustes condition. The rotation was then optimized using Gauss-Newton by optimizing MDL.

In Figure 6.1 the typical behavior of the cost function can be seen. Here, for visualization, two rotations (x and y axis) have been optimized to align three shapes. In the top figure it can be seen that the local minimum close to the origin is a global minimum. It seems to be several minima, but this is due to the periodicity of the function (the range goes from -2π to 2π in x and y). The bottom figure zooms in on the origin (the origin corresponds to the Procrustes solution). It can be seen that the minimum is not at the origin and thus the Procrustes solution does not coincide with the MDL solution. When more shapes are aligned projections of the cost function to lower dimensions look similar to these plots. As can be seen the difference in rotation is not typically large, but in some cases it is.

Experiments were run on seven natural data sets, seen in Figure 6.2 and described in appendix A. The contours were sampled with 64-128 sampling points using arc-length parameterization. The quality of the models built from the aligned datasets was in this experiment measured as the mean square error in leave-one-out reconstructions. This is often interpreted as the models generalization ability, see Section 2.6.1, since it measures the ability of the model to represent unseen shape instances of the object class. The model is built with all but one example and then fitted to the unseen example. As will be seen later in this thesis this measure can be problematic when used to evaluate parameterization functions, but since in these experiments arc-length parameterization is always used, in this case there is no problem. The results are shown in Figure 6.3 and Figure 6.4. The plots show the mean squared approximation error against the number of modes used. For all examples we get models that give the same or lower error when using the description length criterion compared to Procrustes alignment. This means that the models generalize better. The improvements on these datasets are small but consistent. The computational cost of course increases when using this alignment compared to Procrustes, but it is often worth while spending the extra time on building a model of the highest possible quality. Once the model has been built it can be used over and over again.

One reason why the MDL alignment does not succeed in giving more improvement of the results is that when there are many shapes in the training set, the path to the minimum is often narrow and difficult for the optimizer to follow. Also, the derivatives can get numerically unstable close to the minimum.

In some cases the difference between standard Procrustes and MDL alignment is large,

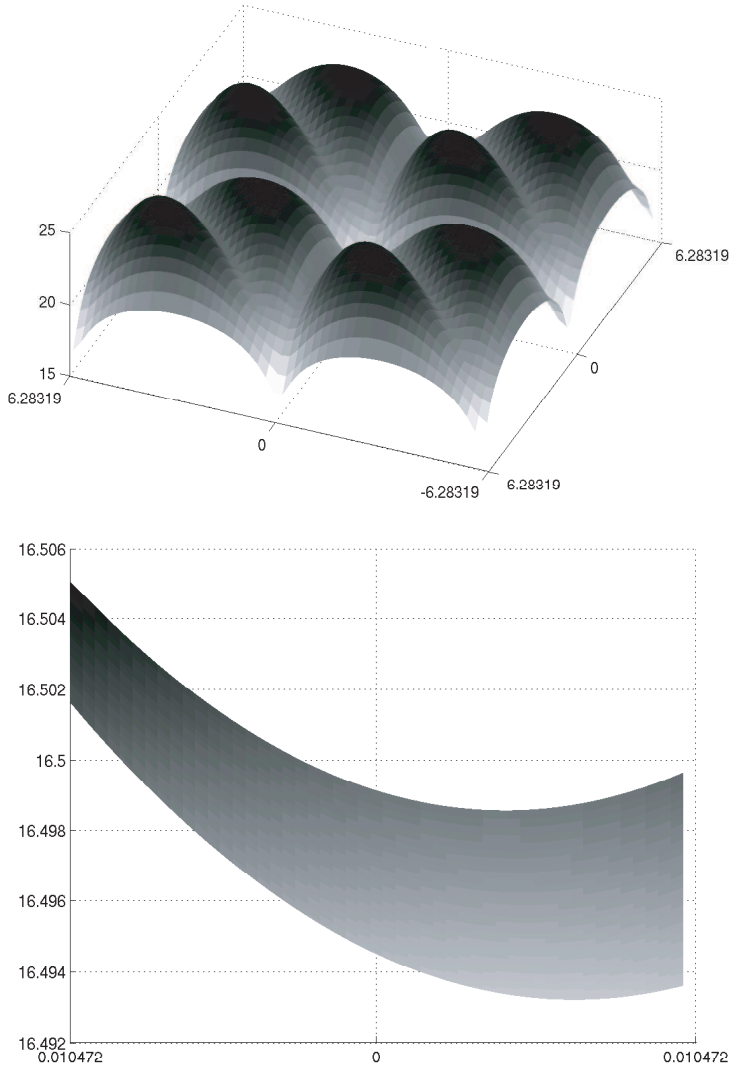


Figure 6.1: The MDL cost function. In the upper figure the range on each axis is -2π to 2π. The lower figure zooms in on the origin showing that the minimum of the 3D surface is not at the origin.

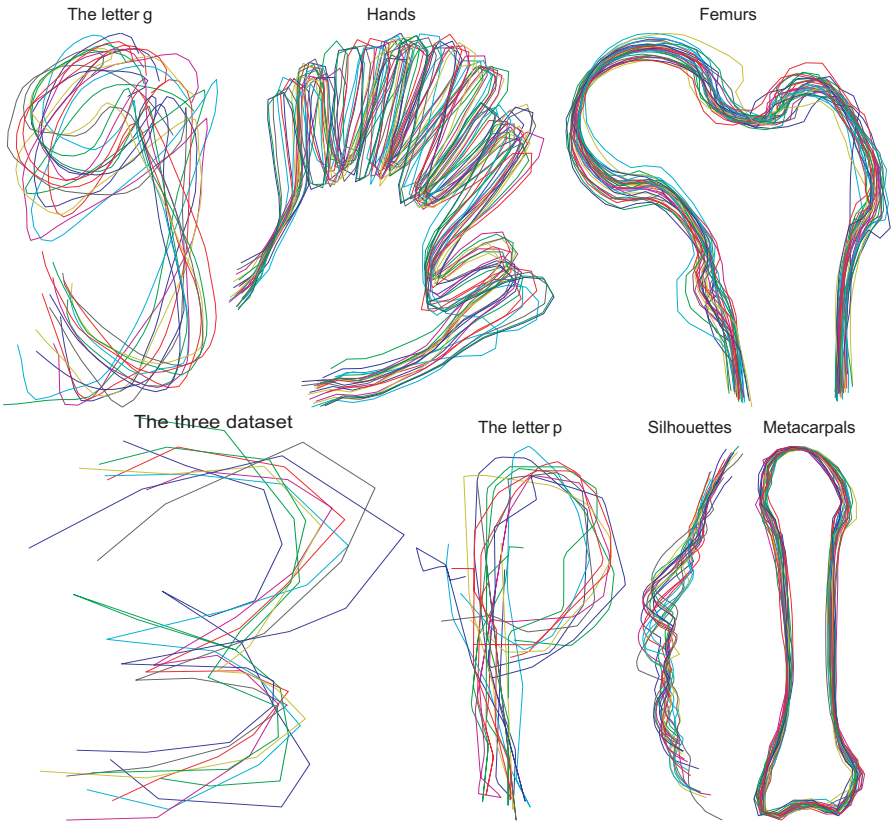


Figure 6.2: The description length aligned datasets.

in favor of MDL alignment. In Figure 6.5 is an example of when standard Procrustes goes visibly wrong. In this case the dataset is a synthetic shape class consisting of identical boxes with an identical bump in different positions on the top edge of the box. This means that this dataset has one linear shape variation mode. The 24 shapes are represented using 128 landmarks, with a higher concentration of landmarks on the bump. For a human it would be natural to align the boxes and let the bump be misaligned, as shown at the top in Figure 6.5, or possibly to align the bumps and let the boxes be misaligned, which is almost what can be seen at the bottom in the same figure (In Chapter 8 it is investigated further how well human opinion agrees with the results of computer algorithms for locating correspondences). Procrustes alignment does not give such a natural alignment and introduces nonlinearities in the shape variation. Note that, since this data only has

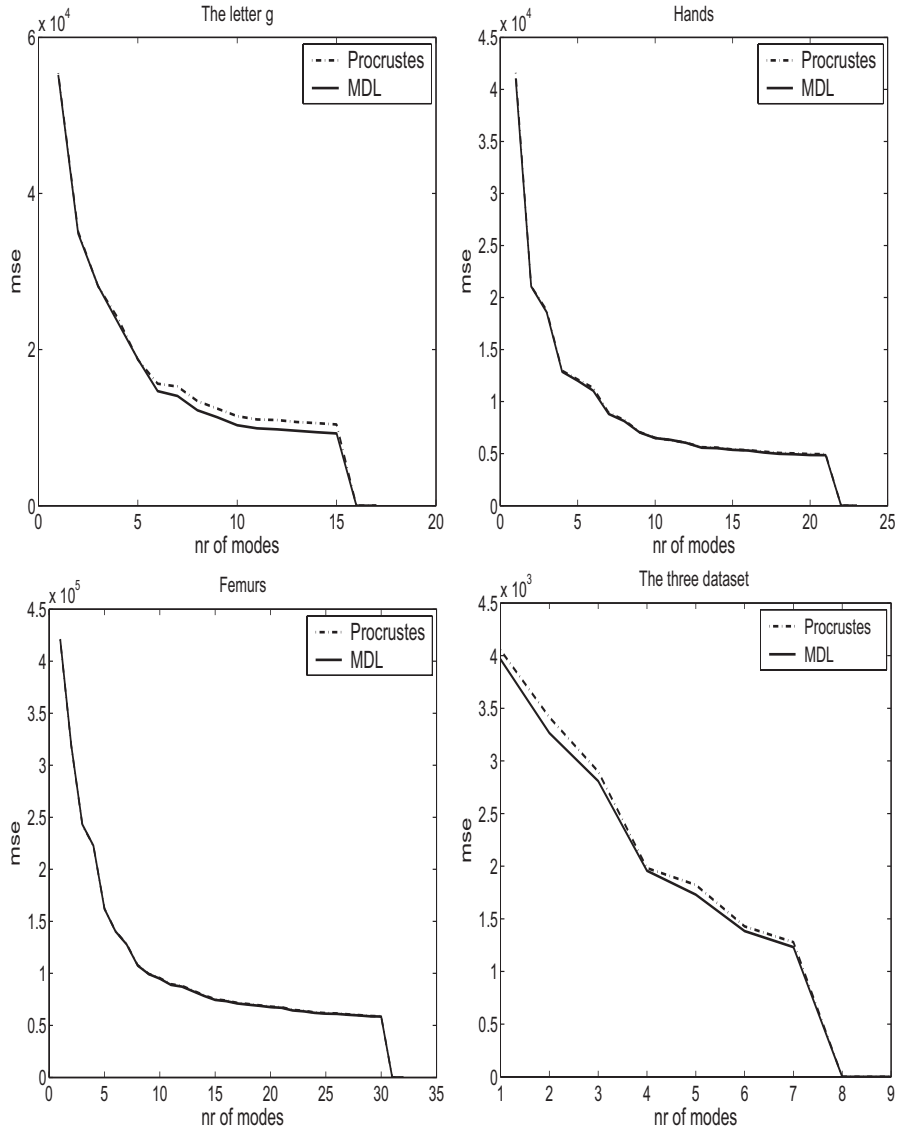


Figure 6.3: The mean squared error of "leave one out" reconstructions of the g-dataset, the hand dataset, the femur dataset and the three dataset.

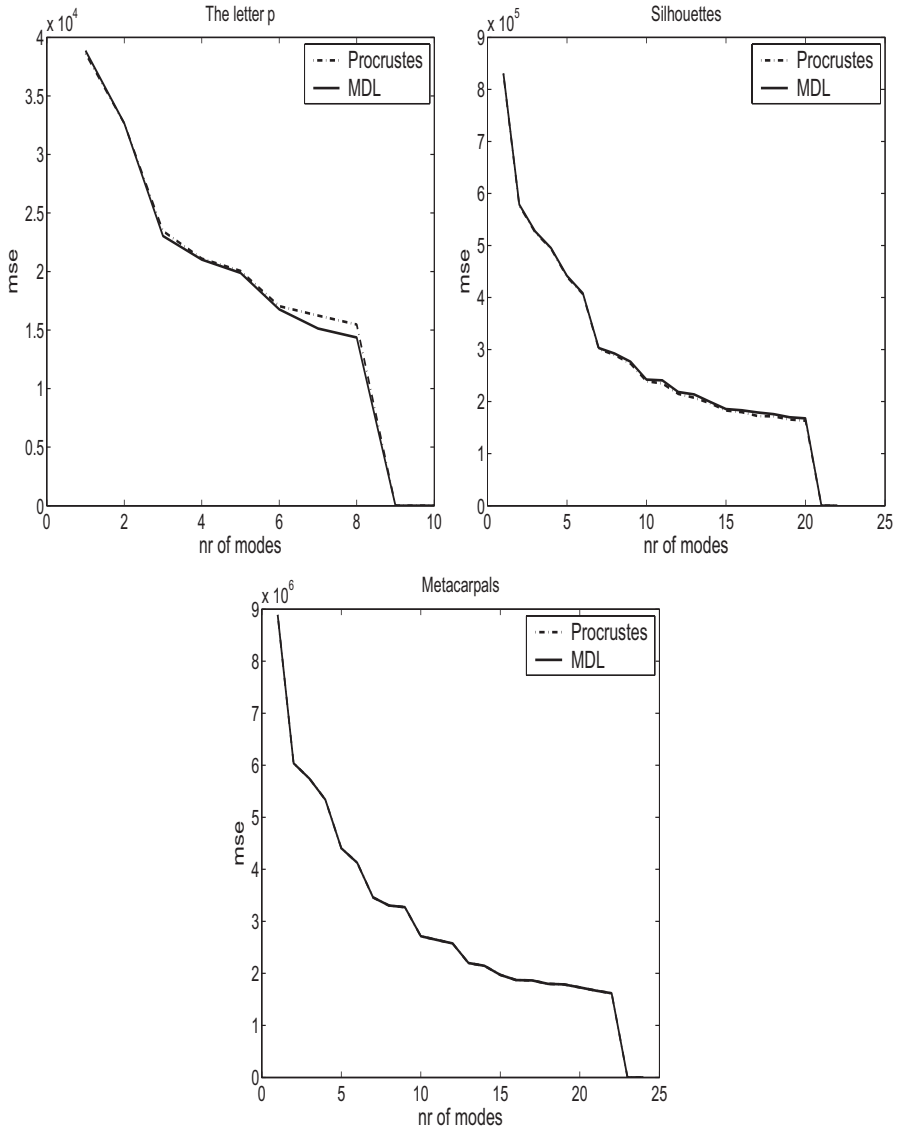


Figure 6.4: The mean squared error of "leave one out" reconstructions of the p dataset, the silhouette dataset and the metacarpal dataset.

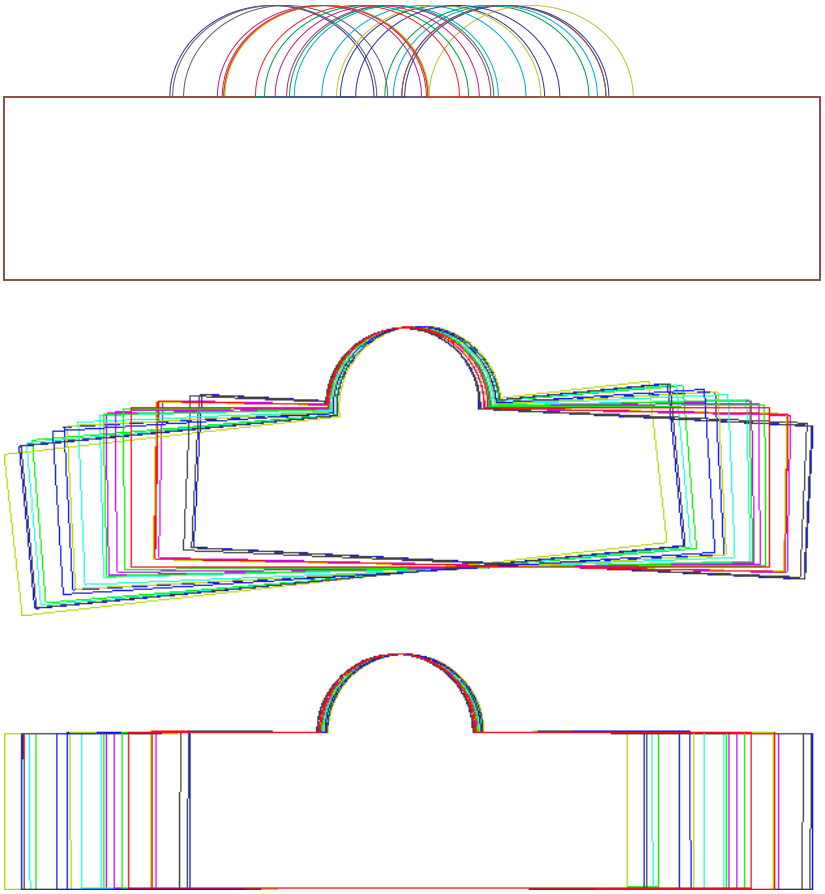


Figure 6.5: A synthetic example. At the top is the training set consisting of boxes with a bump in different positions (one linear shape mode). In the middle is the Procrustes alignment, which has failed to give a natural alignment isolating the shape mode. At the bottom is the result of the description length alignment, which is successful.

one shape mode, perfect alignment should give zero mean squared error on the "leave one out" reconstruction using just one mode. In this experiment the model built from the description length aligned shapes gets almost zero error on the first shape mode, see Figure 6.6, whereas the model built from the Procrustes aligned dataset needs two modes to get a small error. The MDL aligned shapes also corresponds well to how humans would align the shapes, whereas the Procrustes aligned shapes do not. This is a trivial example, but it points out that care must be taken when using Procrustes alignment, especially when parameterizations are determined automatically. In some cases Procrustes really does not give the desired result.

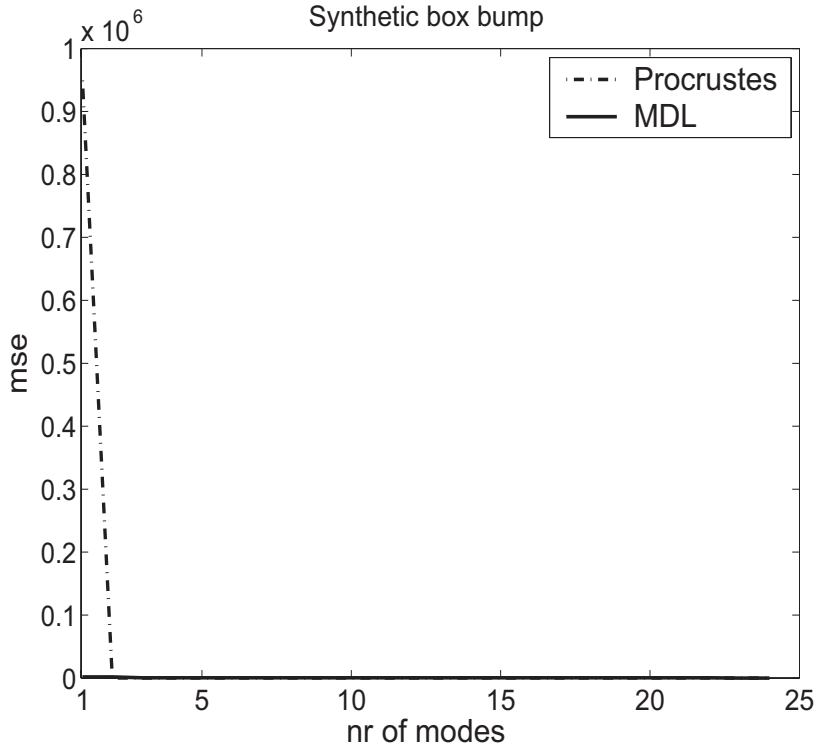


Figure 6.6: The generalization ability of the models. The minimum description length alignment succeeds in locating the only shape mode (the error is zero already with one mode so the curve is extremely close to the x-axis) but the Procrustes alignment needs two modes.

Chapter 7

Stable Correspondence Optimization using Parameterization Invariance

Believe nothing, o monks, merely because you have been told it. Do not believe what your teacher tells you merely out of respect for the teacher. But whatsoever, after due examination and analysis, you find conducive to the good, the benefit, the welfare of all beings - that doctrine believe and cling to and take as your guide.

Buddha

A major problem during correspondence optimization is that landmarks tend to cluster together on parts of the shape with small variation. This is a problem both when moving around a set of landmarks and when defining landmarks via sample points using say equidistant parameter values. Ad hoc strategies to handle this problem include keeping the parameterization of one shape in the training set fixed or adding a penalty to the objective function that penalizes reparameterizing many shapes in the training set in the same way which is how the clustering is often achieved. In this chapter novel theory to prevent clustering during correspondence optimization is presented. The theory is developed for continuous shapes, specifically curves, as opposed to working with a finite number of landmarks along the curves. Instead of representing continuous curves using landmarks, the problem is treated analytically and numerical approximations are introduced at the last stage.

The major problem here is to define shape variation in a way that is invariant to curve parameterizations. The problem is solved by observing that at the core of calculating the shape model is a set of scalar products. These are replaced by modified formulas that are invariant to mutual reparameterizations.

The suggested algorithm increases stability and the resulting models are also of higher quality both according to the generalization measure and according to visual inspection of the specificity.

We first published this solution in 2004 in [76]. In 2007, [139], Twining et.al. described a version achieving similar results by letting the measure used in the scalar product integral be the area measure on the mean shape.

7.1 Introduction

The problem is the following. Correspondences of landmarks do not necessarily imply correspondence of the entire shapes. If many sample points are in approximately one point or a small region with small variation (*clustering*), the objective function measuring the correspondence may get a low value, but this value is based on a small nonrepresentative part of the curve. Increasing the number of sample points does not solve the problem. This is because even though increasing the number of sample points may result in most parts of the shape being sampled, the weight distribution (defined by the parameterization functions) will be unchanged. Most sample points will still be in the small areas with small variation and the value of the objective function will be mostly determined by these points.

MDL along with most objective functions for correspondence optimization suffers from these problems. It has been suggested that this can be prevented by using a "master example", i.e. an example that is not reparameterized during optimization. The idea is that each curve will be reparameterized to fit the "master example". This strategy can work if there are very few shapes in the training set but breaks down if there are too many shapes in the training set. For example, if one keeps the parameterization of the first curve unchanged it is still possible to reparameterize the other $n_s - 1$ curves to put more weight to a particular part of the curves. If there are many curves and they ignore the mismatch with the one kept fixed, their clustering of sample points may lead to a lower MDL while giving worse correspondences. In [135] a penalty is suggested if the mean of a parameterization function control variable over the examples in the training set is different from zero. This penalizes that the parameterizations move in the same direction on all the shapes. This means that the mean movement of corresponding sampling points should be zero during optimization. However, even when these methods prevent clustering collapse, it is a problem that further constraints on the parameterization functions are needed in the optimization formulation, since this will effect optimality.

Here a different solution is proposed. The scalar product at the core of the calculations is replaced with a formula which is invariant to mutual reparameterizations. This formulation is then invariant to the types of reparameterizations that typically cause clustering and it also approximates equal weighting of all parts of the curves.

In this chapter it is shown that the infimum of the minimum description length (MDL) in the standard formulation is actually always zero. This global infimum for the standard formulation is reachable by reparameterizing all the shapes so that all sample points are positioned in approximately the same point (or that approximately all the weight is there in the case of increased number of sample points). This solution to the correspondence problem is of course completely useless, so the global minimum corresponds to the worst possible solution for landmark correspondence.

Using the new formula instead of the standard scalar product when calculating the covariance matrix makes the optimization well defined in the sense that the infimum of

the MDL does not correspond to moving all the sample points to approximately the same point. With this formulation the advantage of clustering (placing all sample points at one point or in a small region) is essentially removed.

7.2 Preliminaries

7.2.1 Continuous Formulation

The standard shape model (introduced in Section 2.5) describes the variation of a set of points as linear combinations of a set of basis vectors

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{\Phi}\mathbf{b} \ . \tag{7.1}$$

Here we will work with a shape theory that is intrinsically defined for continuous objects, specifically curves, and is independent of mutual reparameterizations of the shapes in the training set (such reparameterizations do not change correspondence), i.e. only depends on relative differences in parameterization (which do change correspondence). By mutual reparameterization is meant a reparameterization function that is applied to all curves in the training set. Such a mutual reparameterization function can put most of the weight at one point or a small section of the curve. As with the standard shape model, the model will be a linear model, i.e.

$$\mathbf{c}(s) = \bar{\mathbf{c}}(s) + \sum_{k=1}^{n_m} \mathbf{b}_k \mathbf{\Phi}_k(s) = \bar{\mathbf{c}}(s) + \mathbf{\Phi}(s)\mathbf{b}, s \in [0,1] \ , \tag{7.2}$$

where in the last step we allow ourselves a matrix-type notation even though the $\mathbf{\Phi}_k(s)$ are really infinite-dimensional. Think of $\mathbf{\Phi}$ as consisting of the orthogonal principal components $\mathbf{\Phi}_k(s)$ of the formally infinite dimensional covariance matrix of the shapes in the training set. n_m is the number of modes of shape variation and $\mathbf{c}(s)$, $\bar{\mathbf{c}}(s)$ and $\mathbf{\Phi}_k(s)$ are functions of the continuous parameter s.

The standard way of determining $\mathbf{\Phi}$ from experimental data, where the shapes $\{\mathbf{c}_i\}_{i=1}^{n_s}$ have been aligned e.g. according to the Procrustes condition (similarity transformations), is to find the eigenvectors of the covariance matrix

$$\mathbf{C}_0 = \frac{1}{n_s - 1}\mathbf{X}\mathbf{X}^T \ , \tag{7.3}$$

where each column in $\mathbf{X}$ represents one shape with the mean-shape subtracted.

This is straightforward in the case of finite point configurations, but for curves it is more difficult since the "column vectors" are infinite dimensional. However, (with slight abuse of notation) the covariance matrix $\mathbf{C} = \frac{1}{n_s - 1}\mathbf{X}^T\mathbf{X}$ (where with infinite columns in $\mathbf{X}$, each element is really an integral) is finite dimensional and has the same non zero singular values as $\mathbf{C}_0$. For illustration, let $\mathbf{X}$ be finite dimensional for a moment.

Using singular value decomposition $\mathbf{X}$ can be written $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$, which means that $\mathbf{C} = \frac{1}{n_s-1}\mathbf{V}\mathbf{S}^2\mathbf{V}^T$ and $\mathbf{C}_0 = \frac{1}{n_s-1}\mathbf{U}\mathbf{S}^2\mathbf{U}^T$. We see that, as in Chapter 2, $\mathbf{\Phi} = \mathbf{V}$ and $\mathbf{\Phi}_0 = \mathbf{U}$. The basis $\mathbf{\Phi}_0$ can thus easily be calculated from the singular value decomposition of $\mathbf{C}$ in the following way,

$$\mathbf{\Phi}_0 = \mathbf{U} = \mathbf{U}\mathbf{S}\mathbf{V}^T\mathbf{V}\mathbf{S}^{-1} = \mathbf{X}\mathbf{\Phi}\mathbf{S}^{-1} \ . \tag{7.4}$$

This can now be generalized to an infinite dimensional $\mathbf{X}$ since $\mathbf{C}$ will still be finite. In order to do this (7.4) is interpreted in the following way. $\mathbf{M} = \mathbf{\Phi}\mathbf{S}^{-1}$ is a matrix determining what combinations of the functions $\mathbf{X}_j$ (that with slight abuse of notation are columns in $\mathbf{X}$) make up the modes in $\mathbf{\Phi}_0$,

$$\mathbf{\Phi}_{0i} = \sum_j \mathbf{M}_{ji}\mathbf{X}_j \ . \tag{7.5}$$

In the standard case as described earlier in the thesis, each column in $\mathbf{X}$ is built up as follows. First comes the coordinates for the first landmark or sample point, then the coordinates for the second point etc. This means that the coordinates alternate. In the 2D case we may say that x-coordinates and y-coordinates alternate. This ordering is arbitrary and if we instead build up the columns of $\mathbf{X}$ by first taking the x-coordinates for all sample points and then the y-coordinates this is better suited for the current problem. With this ordering we can think of the columns as first having a continuous function describing the x-coordinate and then another continuous function for the y-coordinate,

$$\mathbf{X} = \begin{bmatrix} \mathbf{c}_1^x \circ \gamma_1 - \bar{\mathbf{c}}^x, \ldots, \mathbf{c}_{n_s}^x \circ \gamma_{n_s} - \bar{\mathbf{c}}^x \\ \mathbf{c}_1^y \circ \gamma_1 - \bar{\mathbf{c}}^y, \ldots, \mathbf{c}_{n_s}^y \circ \gamma_{n_s} - \bar{\mathbf{c}}^y \end{bmatrix} \ , \tag{7.6}$$

$$\bar{\mathbf{c}}^x(s) = \frac{1}{n_s}\sum_{i=1}^{n_s} \mathbf{c}_i^x \circ \gamma_i(s) \ , \tag{7.7}$$

$$\bar{\mathbf{c}}^y(s) = \frac{1}{n_s}\sum_{i=1}^{n_s} \mathbf{c}_i^y \circ \gamma_i(s) \ . \tag{7.8}$$

7.2.2 The Scalar Product and MDL

The element c_{ij} on row i and column j of $\mathbf{C}$ can be seen as a scalar product of shape i with shape j defined by

$$\begin{aligned} c_{ij} = \frac{1}{n_s-1}\Big(&\int_0^1 (\mathbf{c}_i^x(\gamma_i(s)) - \bar{\mathbf{c}}^x(s)) \cdot (\mathbf{c}_j^x(\gamma_j(s)) - \bar{\mathbf{c}}^x(s))ds + \\ &\int_0^1 (\mathbf{c}_i^y(\gamma_i(s)) - \bar{\mathbf{c}}^y(s)) \cdot (\mathbf{c}_j^y(\gamma_j(s)) - \bar{\mathbf{c}}^y(s))ds\Big) \end{aligned} \tag{7.9}$$

Dividing the time interval into $s_0, \ldots, s_N$, where $s_n = n/N$, the scalar product can be evaluated numerically using the set $\{s_n : s_n = n/N, n = 0, \ldots, N\}$.

A useful objective function for solving the correspondence problem

$$\mathbf{c}_i(\gamma_i(s)) :=: \mathbf{c}_j(\gamma_j(s)), \ \ \forall \ i, j \tag{7.10}$$

is the minimum description length of the shape model. The reparameterization functions $\{\gamma_i\}_{i=1}^{n_s}$ are located by minimizing MDL. In Chapter 5 the description length of a training set using the shape model $\mathcal{M}$ was formulated as

$$MDL(\mathcal{M}) = \sum_{\lambda_i > \lambda_c} (\log \frac{\lambda_i}{\lambda_c} + 1) + \sum_{\lambda_i \leq \lambda_c} \frac{\lambda_i}{\lambda_c} + K \ , \tag{7.11}$$

where the scalars λ_i are the eigenvalues of the covariance matrix $\mathbf{C}$, the scalar λ_c is a cutoff constant and K is a scalar, which is independent of the parameterizations. The constant K can be ignored during optimization and the following cost function is used,

$$F(\mathbf{C}) = \sum_{\lambda_i > \lambda_c} (\log \frac{\lambda_i}{\lambda_c} + 1) + \sum_{\lambda_i \leq \lambda_c} \frac{\lambda_i}{\lambda_c} \ , \tag{7.12}$$

where $\mathbf{C}$ is the covariance matrix defined above and the elements of $\mathbf{C}$ can be interpreted as scalar products between the shapes in the training set, as in Equation (7.9).

7.2.3 Formulation Using Vector Scalar Product

Instead of thinking of the shape as one function for each parameter (x and y in 2D), it is possible to use a formulation where the shape is a vector valued function. Then you get the shorter notation

$$c_{ij} = \frac{1}{n_s - 1} \int_0^1 (\mathbf{c}_i(\gamma_i(s)) - \bar{\mathbf{c}}(s)) \cdot (\mathbf{c}_j(\gamma_j(s)) - \bar{\mathbf{c}}(s)) ds \ , \tag{7.13}$$

where the $\cdot$ stands for the vector scalar product, which of course means that this is actually the same as the formula in Equation (7.9) above. Because of the clearer connection with the discrete case and with implementation we will use the earlier formulation in Equation (7.9) without the vector valued functions and without the vector scalar product.

7.3 The Problem with Parameterization Dependence

One problem with optimizing MDL as presented above is that it is not independent of mutual reparameterizations. The scalar product depends on the reparameterization functions γ_i and γ_j. How a parameterization function can distribute the weights on different parts of a curve is illustrated in Figures 5.1. In this figure it can be seen how

two parameterization functions put more weight on the first part of the curve. Using the standard scalar product in this case, it is possible that $\gamma_i(s_n) \leq \epsilon$ for all $s_n < s_N$. This means that $\mathbf{c}_i(s), s \in (\epsilon, 1)$ is not taken into account in the calculation of the scalar product. If the number of sample points is increased it is possible to get some points $\mathbf{c}_i(s), s \in (\epsilon, 1)$ but γ_i may still be such that the *fraction* of points $\mathbf{c}_i(s), s \in (\epsilon, 1)$ will be neglectable. Another example is given in Figure 5.2. Here two parameterization functions put more weight on the middle part of the curve so that $|\gamma_i(s_n) - 0.5| \leq \epsilon$ for all sample points $0 < s_n < s_N$ so $\mathbf{c}_i(s), s \in (0, 0.5 - \epsilon) \cup (0.5 + \epsilon, 1)$ is not taken into account. If the number of sample points is increased it is possible to get some points $\mathbf{c}_i(s), s \in (0, 0.5 - \epsilon) \cup (0.5 + \epsilon, 1)$ but γ_i may still be such that the *fraction* of points $\mathbf{c}_i(s), s \in (0, 0.5 - \epsilon) \cup (0.5 + \epsilon, 1)$ will be neglectable. Even if the entire shape is considered to some extent, the result is not independent of curve parameterizations. By changing all parameterizations γ_i with a mutual parameterization γ, so that $\tilde{\gamma}_i = \gamma_i \circ \gamma$, one effectively puts different weights at different parts of the curves without changing the correspondences. The same problem occurs when a discrete set of landmarks is used as in [32, 34]. The most weight is put on that part of the curve which has the most landmarks. The problem with this is that the optimization scheme for locating correspondences becomes ill-posed as illustrated by the following theorem. In this chapter the standard scalar product is replaced by a formula that does not suffer from this problem.

Theorem 7.3.1. *For $F(\mathbf{C})$ defined by (7.12) via (7.9),*

$$\inf_{\gamma_1,\ldots,\gamma_{n_s}} F(\mathbf{C}) = 0 \ . \tag{7.14}$$

The infimum is attained when the reparameterization functions approach a measure with all mass at one point.

Proof. Assume that a training set of continuous curves $\mathbf{c}_i(t), i = 1, \ldots, n_s, t \in [0, 1]$, is given and that the reparameterization functions $\gamma_i, i = 1, \ldots, n_s$ put the shapes in correspondence,

$$\mathbf{c}_i(\gamma_i(s)) :=: \mathbf{c}_j(\gamma_j(s)) \ . \tag{7.15}$$

Now, construct a reparameterization function γ such that,

$$\gamma(t) = \begin{cases} \frac{\epsilon}{1-\epsilon} t, & 0 \leq t \leq 1 - \epsilon \\ \epsilon + \frac{1-\epsilon}{\epsilon}(t - (1 - \epsilon)), & 1 - \epsilon \leq t \leq 1 \end{cases} \ . \tag{7.16}$$

This function is illustrated in Figure 7.1. The correspondence of the shapes is retained, if all curves are reparameterised with the parameterisation function γ. If now ϵ is sufficiently small all weight will be situated close to the start of the curves. Performing Procrustes on these shapes will result in, that in the limit $\epsilon \to 0$, all shapes coincide with the mean-shape. This means that the covariance matrix $\mathbf{C}$ will be the zero matrix. Hence all eigenvalues of $\mathbf{C}$ are zero and $F(\mathbf{C}) = 0$, if $\lambda_c > 0$. ∎

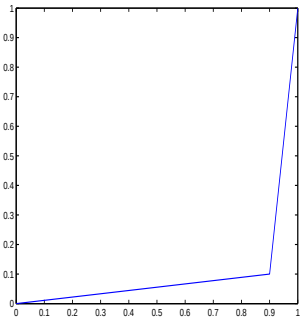


Figure 7.1: Mutual reparameterization function. Here $\epsilon = 0.1$.

This is not only a theoretical problem. This minimum can easily be found. Assume that the curves are in approximate correspondence. The optimizer can now start to concentrate points on parts of the curve with small variation, while keeping correspondence, by performing mutual reparameterizations, i.e reparameterizing all curves by the same reparameterization function. So this is a major problem also in practice and the minimization criterion is not well defined. A good global optimizer would find this meaningless global minimum. Even if the global minimum is not found it is still possible to reduce $F(\mathbf{C})$ by concentrating the sampling points on parts of the curves with low variance. This means that it is possible to reduce the MDL in two ways, both by finding correspondences and by performing mutual reparameterization which actually give worse solutions. Therefore algorithms for automatic correspondence localization on shapes tend to gather sample points together (concentrate the weight).

7.4 A Parameterization Invariant Method

To improve the algorithm discussed above, the standard scalar product is replaced with a formula which is invariant to mutual reparameterizations. We will think of the new formula as a new scalar product. This removes the undesired way to reduce MDL and the optimizer focuses on establishing correspondences.

7.4.1 Defining the New Formula

Definition 7.4.1. Let $\{\mathbf{c}_i\}_{i=1}^{n_s}$ be a training set of curves, parameterized with $\{\gamma_i\}_{i=1}^{n_s}$ and centered at the origin. A new scalar product type formula between $\mathbf{c}_i$ and $\mathbf{c}_j$ is

defined by

$$\mathbf{c}_i \cdot \mathbf{c}_j = \frac{1}{n_s - 1}\frac{1}{n_s}\sum_{k=1}^{n_s}[\int_0^1 \left(\mathbf{c}_i^x\left(\gamma_i \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_k^{-1}(s))\right) \cdot \\ \cdot \left(\mathbf{c}_j^x\left(\gamma_j \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_k^{-1}(s))\right) ds + \\ \int_0^1 \left(\mathbf{c}_i^y\left(\gamma_i \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_k^{-1}(s))\right) \cdot \\ \cdot \left(\mathbf{c}_j^y\left(\gamma_j \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_k^{-1}(s))\right) ds] \quad (7.17)$$

Intuitively, what happens is that if the parameterization functions γ_i all put more weight into one part of the shapes, the composition of γ_i with the inverse parameterization functions γ_k^{-1} neutralizes this so that the curves are sampled more equally. The result of the inverse functions for the case in Figure 7.2 is illustrated in Figure 7.3.

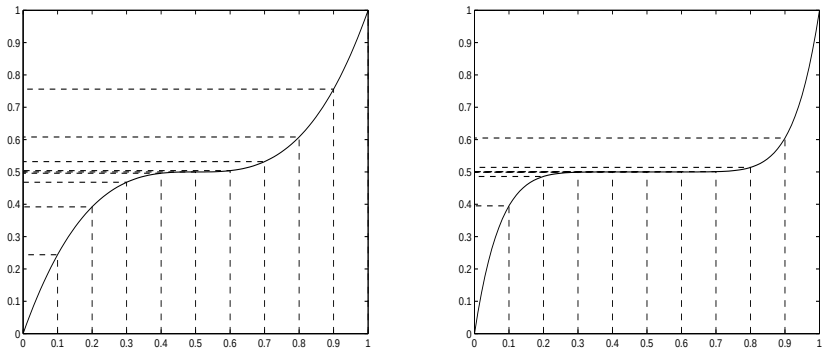


Figure 7.2: In the two plots it can be seen how the sampling points, defined by equidistant parameter values in [0,1], are mapped by two parameterization functions. The plots show how the sampling points can cluster during optimization of the parameterisation functions. In this example they cluster at the middle part of the curve.

This also means that it is not possible to neglect any part of the curves, since it can be seen that there is one term that gives arc-length parameterization for $\mathbf{c}_i$ and one term that gives arc-length parameterization for $\mathbf{c}_j$.

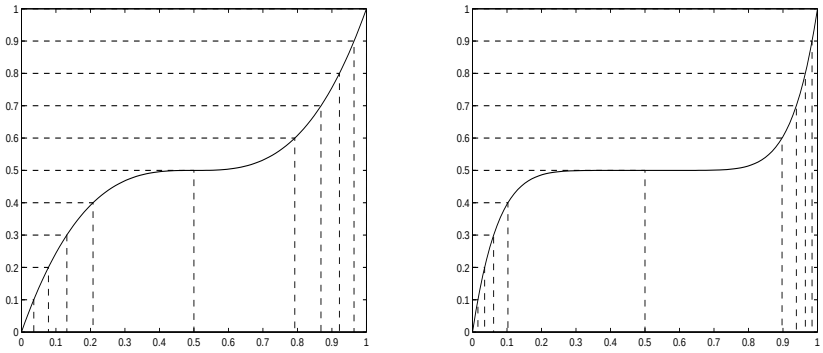


Figure 7.3: In the two plots it can be seen how the sampling points, defined by the set of equidistant points, are mapped from the y-axis to the x-axis by two inverse parameterization functions.

7.4.2 Proving Invariance

Theorem 7.4.1. *The formula presented in Definition 7.4.1 is invariant to mutual reparameterizations.*

Proof. Let $\{\gamma_i\}_{i=1}^{n_s}$ be a set of parameterization functions such that

$$\mathbf{c}_i(\gamma_i(s)) :=: \mathbf{c}_j(\gamma_j(s)), \;\; i = 1 \dots n_s, j = 1 \dots n_s \;\; . \tag{7.18}$$

Let γ be an arbitrary reparameterization function and let $\tilde{\gamma_i}(s) = \gamma_i \circ \gamma$ and $\tilde{\gamma_j}(s) = \gamma_j \circ \gamma$. Then the correspondence $\mathbf{c}_i(\tilde{\gamma_i}(s)) :=: \mathbf{c}_j(\tilde{\gamma_j}(s))$ still holds.

From the definition of the proposed formula we get,

$$\begin{aligned}
&\mathbf{c}_i(\tilde{\gamma_i}(s)) \cdot \mathbf{c}_j(\tilde{\gamma_j}(s)) = \\
&\frac{1}{n_s - 1}\frac{1}{n_s}\sum_{k=1}^{n_s}[\int_0^1 \left(\mathbf{c}_i^x\left(\tilde{\gamma_i} \circ \tilde{\gamma_k}^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\tilde{\gamma_m} \circ \tilde{\gamma_k}^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^x\left(\tilde{\gamma_j} \circ \tilde{\gamma_k}^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\tilde{\gamma_m} \circ \tilde{\gamma_k}^{-1}(s))\right) ds + \\
&\int_0^1 \left(\mathbf{c}_i^y\left(\tilde{\gamma_i} \circ \tilde{\gamma_k}^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\tilde{\gamma_m} \circ \tilde{\gamma_k}^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^y\left(\tilde{\gamma_j} \circ \tilde{\gamma_k}^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\tilde{\gamma_m} \circ \tilde{\gamma_k}^{-1}(s))\right) ds] =
\end{aligned}$$

[and since $\tilde{\gamma_i}(s) = \gamma_i \circ \gamma$ and $\tilde{\gamma_j}(s) = \gamma_j \circ \gamma$]

$$
\begin{aligned}
&= \frac{1}{n_s - 1}\frac{1}{n_s}\sum_{k=1}^{n_s}[\\
&\int_0^1 \left(\mathbf{c}_i^x\left(\gamma_i \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^x\left(\gamma_j \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s))\right) ds + \\
&\int_0^1 \left(\mathbf{c}_i^y\left(\gamma_i \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^y\left(\gamma_j \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma \circ \gamma^{-1} \circ \gamma_k^{-1}(s))\right) ds] =
\end{aligned}
$$

[and using that $\gamma \circ \gamma^{-1}$ is the identity function]

$$
\begin{aligned}
&= \frac{1}{n_s - 1}\frac{1}{n_s}\sum_{k=1}^{n_s}[\int_0^1 \left(\mathbf{c}_i^x\left(\gamma_i \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_k^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^x\left(\gamma_j \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_k^{-1}(s))\right) ds + \\
&\int_0^1 \left(\mathbf{c}_i^y\left(\gamma_i \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_k^{-1}(s))\right) \cdot \\
&\cdot \left(\mathbf{c}_j^y\left(\gamma_j \circ \gamma_k^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_k^{-1}(s))\right) ds] = \\
&\mathbf{c}_i(\gamma_i(s)) \cdot \mathbf{c}_j(\gamma_j(s)) \ .
\end{aligned}
$$

■

The new formula would actually be invariant to mutual reparameterizations using only one term in the sum in Definition 7.4.1. Two terms may also seem like a natural choice and using two terms it is natural to choose the terms that parameterize $\mathbf{c}_i$ and $\mathbf{c}_j$

to arc-length respectively.

$$\begin{aligned}
\mathbf{c}_i \cdot \mathbf{c}_j = \frac{1}{n_s - 1}\frac{1}{2}[\int_0^1 & \left(\mathbf{c}_i^x(s) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_i^{-1}(s))\right) \cdot \\
& \cdot \left(\mathbf{c}_j^x\left(\gamma_j \circ \gamma_i^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_i^{-1}(s))\right) ds + \\
\int_0^1 & \left(\mathbf{c}_i^y(s) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_i^{-1}(s))\right) \cdot \\
& \cdot \left(\mathbf{c}_j^y\left(\gamma_j \circ \gamma_i^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_i^{-1}(s))\right) ds + \\
\int_0^1 & \left(\mathbf{c}_i^x\left(\gamma_i \circ \gamma_j^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_j^{-1}(s))\right) \cdot \\
& \cdot \left(\mathbf{c}_j^x(s) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^x(\gamma_m \circ \gamma_j^{-1}(s))\right) ds + \\
\int_0^1 & \left(\mathbf{c}_i^y\left(\gamma_i \circ \gamma_j^{-1}(s)\right) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_j^{-1}(s))\right) \cdot \\
& \cdot \left(\mathbf{c}_j^y(s) - \frac{1}{n_s}\sum_{m=1}^{n_s}\mathbf{c}_m^y(\gamma_m \circ \gamma_j^{-1}(s))\right) ds] \ . \qquad (7.19)
\end{aligned}$$

However, even though invariance to truly mutual reparameterizations is achieved like this, or even by using only one term, it is still possible to find parameterizations that decrease MDL by gathering sample points together in different ways on different curves. For example this can be done if the curves are grouped so that sample points are clustered in different ways in the groups.

Theoretically, clustering at some small level is still possible even when all the terms are used as in Definition 7.4.1, but using all the terms makes it hard for the optimization algorithm to locate such parameterizations and also, the effect would be small. Therefore it is better to use Definition 7.4.1. However, if time is an issue, using only one or two terms may be desirable.

To evaluate the scalar product numerically it is of course necessary to sample the curves. These sample points are just a step in the evaluation and should not be confused with landmarks in the traditional sense. In practice this is numerically achieved as follows. From the set $\mathcal{S}_0 = \{s_n : s_n = n/(n_p - 1), n = 0, \ldots, n_p - 1\}$, which is the set that defines the sample points, the set $\mathcal{T}_k = \gamma_k^{-1}(\mathcal{S}_0)$ is retrieved. This is shown schematically in Figure 7.3. $\mathcal{T}_k$ decides what parameter values are used in term k in Definition 7.4.1. In practice, one set of parameter values $\{\mathcal{T}_k, (k = 1, \ldots, n_s)\}$ can be defined and the scalar

product can be evaluated as one term. Since all the sets $\mathcal{T}_i (i = 1, \ldots, n_s)$ are used in Definition 7.4.1, as opposed to when using only one or two terms, corresponding points across the entire training set are in this case used to calculate the different c_{ij}. This is another advantage of using all the terms.

7.5 Experimental Validation

In this section the algorithm is validated on the hand-, femur- and g- data sets described in the appendix.

Models built using the proposed formula are compared with models built using the standard scalar product. For the standard scalar product the algorithm uses the parameterization function control variable cost described above and in [135] to penalize mutual reparameterizations (control vaiable mean different from zero) in order to prevent clustering of sample points.

Thodberg's implementation of MDL [135] has been used for the comparison in these experiments.

Seven parameterization function control variables controlling seven basis functions have been used for each reparameterization function. To evaluate the minimum description length for a given parameterization the curves must be sampled. For the proposed formula the sampling of each parameterization function $\gamma_i (i = 1, \ldots, n_s)$ results in n_p sample points. n_p was set to 65. This means that for the proposed formula the number of sample points are $n_p n_s$. This results in 1105 to 2080 sample points for the three different datasets. To give a fair comparison, the number of sample points were set to 2048 for the standard scalar product. Other initializations and parameters for the optimization algorithms were identical in all cases.

The resulting model quality is evaluated by looking at specificity and generality. Remember that by specificity is meant that shapes generated by the model using reasonable parameter values (not more than ± 3 std.) should be representative for the shape class. The definition of specificity is given in Section 2.6.2. In Chapter 8 it is shown that the quantitative measure of specificity is problematic. A better way to evaluate the specificity is to simply plot shapes produced by using equal parameter values in the different models that are to be compared. By increasing the parameter value up to 3 std. it can often be seen which model is more specific.

The specificity evaluation is presented for the most difficult dataset, the g-shapes. In Figure 7.4, the mean-shape and the mean-shape plus and minus three standard deviations of the first shape mode is plotted for each model. In the figure, it can be seen that the shapes generated by the proposed algorithm are somewhat more representative. Look at the left plot (standard scalar product used) and note, for example, the sharp bend in middle of the dashed curve and the not sharp enough corner at the end of the first arc of the mean shape (the solid curve). That this corner is not sharp is a sign that the correspondences at this point are matching badly which results in a smoothing effect. Also, when

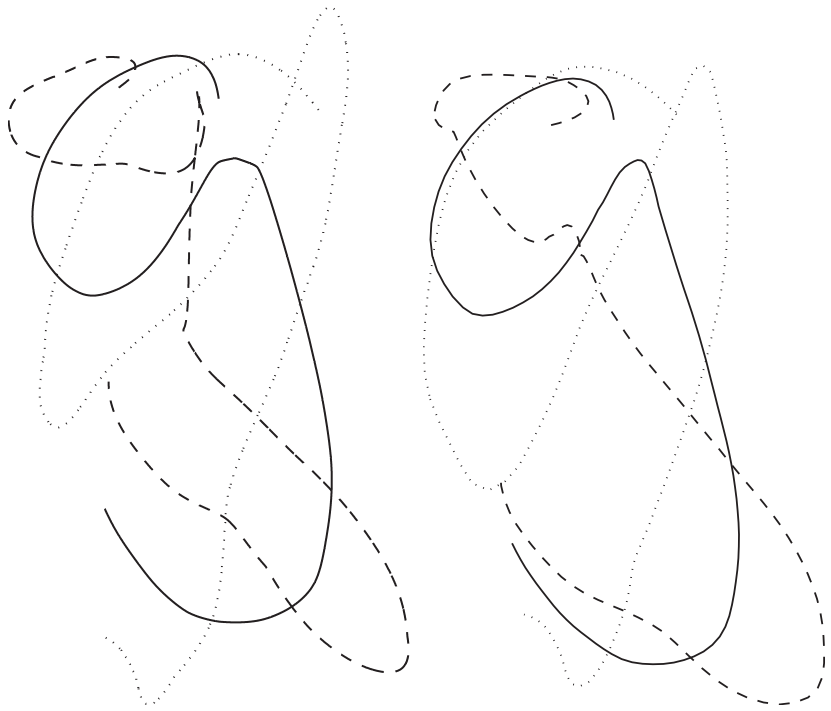


Figure 7.4: The mean (solid line) and the first mode of variation (dashed and dotted lines) of the optimized models. The model on the left is optimized and built using the standard scalar product and the model on the right is optimized and built using the proposed new formula.

comparing the two dotted g:s it can be seen that the one to the right (proposed formula used) looks more natural, with less artificial bends, corners where there should be corners and smoothness where it should be smooth. For the other datasets in this experiment the differences in specificity of the two models are somewhat harder to distinguish.

The generalization ability of the models is measured as the mean square error in "leave-one-out" reconstructions. This measures the ability of the model to represent unseen shape samples or, in other words, the models ability to *generalize*. The definition using a discrete set of landmarks is given in Section 2.6.1. In this experiment, the error between the original curve and the model approximation curve is calculated by integrating the squared distance between the curves by arc-length. The result is shown in Figure 7.5. The plots show the mean square approximation error against the number of modes used. It can be seen that the new formula gives models that generalize better than models

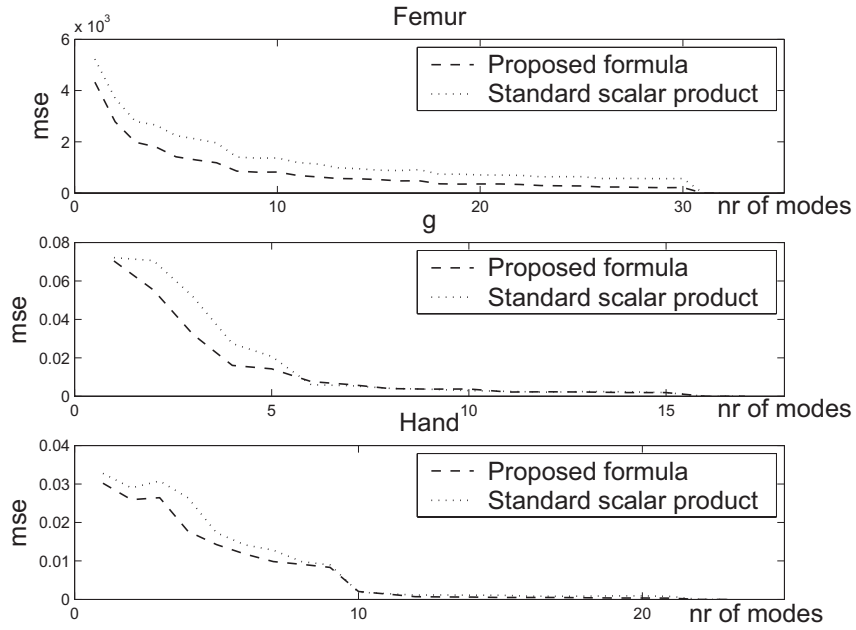


Figure 7.5: The mean square approximation error of the six models is plotted against the number of modes used. The top figure shows the models of femurs, the middle figure shows the models of g:s and the bottom figure shows the models of hands.

built using the standard scalar product, even if node cost penalties are used to prevent clustering with the standard scalar product.

7.6 Summary and Conclusions

In this chapter the standard scalar product is replaced with a new formula that is invariant to mutual reparameterizations. The optimization of the shape model becomes invariant to mutual reparameterizations and is therefore better suited for finding correspondences, since clustering is prevented.

The global infimum of the optimization problem using the standard scalar product was shown to correspond to reparameterizing all the shapes to put all the weight in a single point, a useless solution. Using the proposed formula this problem is avoided and a more robust and stable algorithm is achieved. Even when the algorithm using the standard scalar product does not collapse, an improvement in quality is achieved using

the proposed formula.

The algorithm was compared to an algorithm, which uses ad hoc solutions to prevent clustering of sample points. The comparison showed that the achieved models using the proposed formula are more general for all the three tested datasets. The model of the most difficult dataset turns out to be more specific. It was harder to distinguish the differences in specificity of models of the two other datasets.

A similar approach replacing the standard scalar product using inverse parameterization functions should be able to extend to 3D models of surfaces.

Chapter 8
Benchmarking of Algorithms for Correspondence Optimization

From a persons personal point of view, artifacts don't make us smarter or make us have better memories; they change the task. From the system point of view, the person+artifact is more powerful than either alone.

Donald A. Norman

There are many algorithms available for automatic determination of correspondences for the construction of Statistical Shape Models from examples. Benchmarking is needed to rank the different algorithms. Prior work has argued that the quality of the models produced by the algorithms can be evaluated by a specific set of measures designed to measure compactness, generality and specificity. In this chapter severe problems with these standard measures are discussed and analyzed both theoretically and experimentally, both on natural and on synthetic datasets. Instead of the standard measures, a ground truth correspondence measure (GCM) can be used for benchmarking and in this chapter such a measure is defined and used for benchmarking of several algorithms. Minimum Description Length (MDL) with an added curvature cost comes out as the winner of the automatic methods. Hand marked correspondences turn out to be best when this is feasible and a semi-automatic method is shown to lie in between the best automatic method and the hand built models in quality.

8.1 Introduction

As described earlier in this thesis, there are many algorithms available for automatic shape modeling. The field has matured and there is a need for benchmarking of these algorithms. Similar developments have taken place in different fields, e.g. the field of stereo [110].

In order to evaluate these algorithms, different measures of the quality of the parameterizations and the resulting models have been used. What should be considered as a shape model of high quality is highly dependent on the application. For example the model that performs best on segmentation might not be the model that performs best on classification. If the model is to be used for a specific purpose, such as segmentation of the brain in SPECT images, the choice of algorithm should be made using a criterion

based on the application. For a more general evaluation of shape model quality the standard measures have been compactness, specificity and generality [31]. It is also common to evaluate correspondences subjectively by plotting the shapes with some corresponding points marked. These methods have become very popular since they do not require any ground truth data, which is very time consuming to collect. As this chapter will show however, these measures are problematic and ground truth is necessary after all.

In [111] the quality of registrations of images is evaluated both by measuring the overlap with ground truth and by measuring model quality measures such as specificity and generality of models constructed from the registered images. There it is claimed that ground truth correlates with generality and specificity. This is shown by doing random perturbations of ground truth and noting that all the measures increase monotonically as the level of perturbation increases. However, this does not show that the measures are minimal at the ground truth. The path to the minimum is not likely to be found by a random perturbation. Measuring the sensitivity to perturbation of ground truth it is also claimed that specificity is more sensitive than the other two measures. Instead of measuring sensitivity to random perturbation it would be more interesting to examine which measure is most suitable for choosing between two training sets produced by different non-random strategies. This might be done by letting human experts choose which is the best of the two compared training sets.

In this chapter, problems with the standard general model quality measures, namely compactness, generality and specificity, are analyzed. It is shown that they do not succeed in measuring what they attempt to measure. With practical experiments it is also shown that the standard measures do not correlate well with correspondence quality.

The qualities that the standard measures attempt to measure are often, but not always, important. However, even when they are important, as we will see, it is problematic to actually measure them.

For most applications, high quality of the correspondences is both desirable and highly important. A shape model built from correct correspondences is a model that correctly describes the shape variation within the class, whereas, as will be shown, a simplified model can get excellent measure of specificity, generality and compactness, but relevant shape information may have been lost. To cope with this, here it is proposed that a ground truth correspondence measure (GCM), measuring the quality of the correspondences at important locations, is used for benchmarking. Verification using ground truth is an established method in computer vision, see for example the CAVIAR project [50]. Ground truth has also been used for verification of segmentation, see for example [96]. In [129] evaluation of four algorithms for 3D-correspondence is done with a simple ground truth correspondence measure used together with the standard measures to compare the algorithms. However, no evaluation of the evaluation measures is done. In this chapter a close examination of the different evaluation measures is performed.

The compactness-, specificity- and generality-measures were described in Section 2.6. In Section 8.2 theoretical weaknesses of these measures are discussed. In Section 8.3 a

measure called Ground truth Correspondence Measure (GCM) is introduced. This new measure makes the subjective evaluation of correspondences more objective and quantitative. In Section 8.4 experiments show that all the standard measures can lead to very wrong conclusions when used to evaluate algorithms. GCM however always corresponds to the subjective evaluation or in synthetic examples to the known ground truth. In Section 8.5 several algorithms for finding correspondences are bench-marked using GCM.

The three major contributions of the work in this chapter are: (i) It is shown by theoretical analysis and practical experiments on both natural (extracted from real images) and synthetic datasets that the former shape model measures compactness, specificity and generality have severe weaknesses. (ii) A correspondence quality measure (GCM) is proposed and it is shown that GCM together with a database of ground truth correspondences is well suited for benchmarking algorithms for correspondence localisation. (iii) Benchmarking of several state of the art algorithms is presented and MDL with curvature cost comes out as the winner of the automatic methods. (iv) Contrary to former results handmade models are shown to be the best and a semiautomatic algorithm is proposed that outperforms all of the automatic algorithms. One might argue that it is obvious that a measure built on ground truth data, favors other hand built models marked by other humans. One must then also keep in mind that our measure gives extremely low value also for the hand built synthetic box-bump model.

8.2 Discussion of Model Quality Measures

It is important to build the model from correct correspondences. Assume that a set of training shapes with correct correspondence, $\mathbf{c}_i(\gamma_i(s)) := \mathbf{c}_j(\gamma_j(s))$ over all the shapes at all points, is given. For medical images this means that every anatomical point in one shape is matched to the same anatomical points in all other shapes in the training set (even for non-medical data, we still use the term anatomical to refer to points such as for example the upper left corner of the right headlight on a car). A model built from these dense correspondences will describe all the shape variation that is present in the training set and no variation that is not present. If we perturb the correspondences, new artificial shape modes are likely to be introduced into the model. These shape modes describe movement of points along the shape, due to incorrect correspondences. It may also be the case that there are real shape variations that are lost e.g. by algorithms favoring closeness between corresponding points. Therefore correct correspondences are very important for model quality. Specificity and generalization ability are also important properties of a shape model, but as we will see, they are both difficult to measure. Also a common situation when measuring generality, specificity and compactness is that the curves ($G(n_m), S(n_m)$ or $C(n_m)$) for different models intersect each other and in this case it is not possible to choose which model is of higher quality. Sometimes this problem can be avoided since it may be enough to only examine the asymptotic value of the quality measures, but often this is not desirable since one often wants to have few modes.

In this section, the three standard measures will be analyzed and specific weaknesses will be described.

8.2.1 Generality

One major problem with measuring generality defined using Equation (2.33) is that the parameterization of the shape that has been left out is unknown. This is often handled by letting the shape be included together with the training set in the correspondence optimization, and removing it first before the final model is built. This produces a bias so that the generality cost for the optimized model will always be underestimated so that it looks better than it is. This is especially true when optimizing costs like MDL since there is the risk of oversimplifying the shapes by optimizing the parameterizations using MDL and similar measures. This is because incorrect correspondences that get rid of true shape variation will give a better MDL. To avoid these problems it would be necessary to run the optimization procedure once for each shape that is left out and then pick a suitable parameterization for the left out shape and try to represent this shape using the model. There is no obvious way to pick the parameterization of the left out shape. One way to decide on this parameterization function is by solving the correspondence problem. This means that evaluation of solutions to the correspondence problem in their turn require solutions to the correspondence problem for the left out shape, but then we are back to including the test shape in the optimization, resulting in the bias above.

In [111], a different version of the generalization measure is introduced using the formula

$$G(n_m) = \frac{1}{n_s} \sum_{j=1}^{n_s} ||\mathbf{y}_j - \mathbf{y}'_j(n_m)|| \ , \tag{8.1}$$

where $\mathbf{y}_j$ are the shapes in the training set and $\mathbf{y}'_j$ is the closest shape to $\mathbf{y}_j$ minimized over a set of model generated shapes and n_m is the number of modes used to create the samples. Formally,

$$\mathbf{y}'_j = \arg \inf_{\mathbf{x}_j \in \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}} ||\mathbf{y}_j - \mathbf{x}_j(n_m)|| \ . \tag{8.2}$$

This version avoids the "leave one out" problem, but it does not measure generalization ability to shapes outside the training set. Consider the shape space $\mathbf{\Omega}_{n_p}^{n_d}$ of all possible shapes with n_p points in n_d dimensions. Let Ω_M be the subspace generated by the model. Then $\arg \inf_{\mathbf{x}_j \in \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}} ||\mathbf{y}_j - \mathbf{x}_j(n_m)||$ is actually an approximation of the orthogonal projection of the shape $\mathbf{y}_j$ in the training set onto the subspace Ω_M. Therefore there is no need for approximating the distance, since the projection can be calculated using the scalar product and the vectors generating Ω_M. This is illustrated in Figure 8.1. If the model spans the whole space spanned by the training set the projection distance will of course be zero. Then it may be argued that we are instead interested in

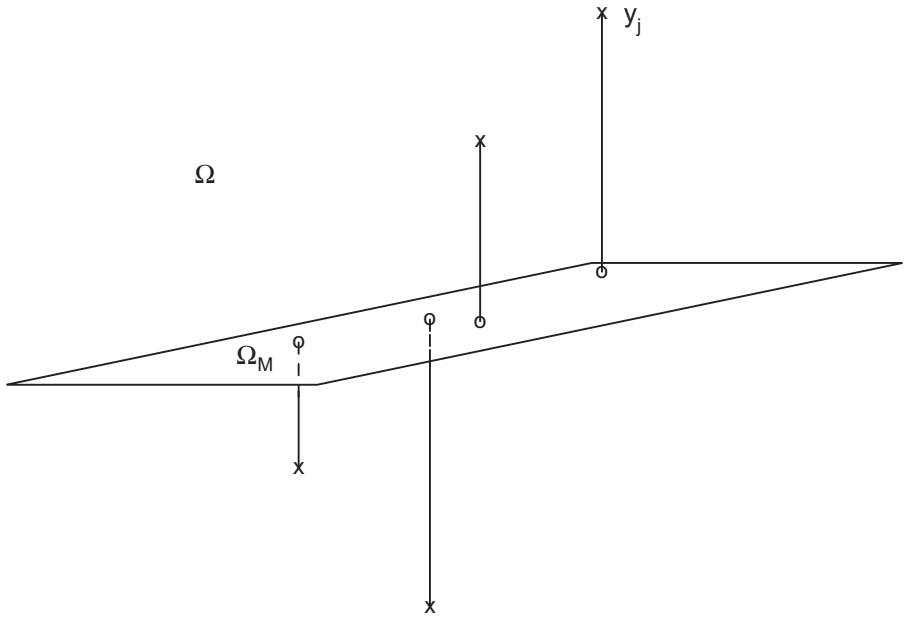


Figure 8.1: The generality measure (8.1) approximates the sum of distances from the shapes in the training set to the shape space generated by the model.

measuring the closest distance for a bounded number of shapes in Ω_M. However, this does not measure generality. Actually, a more general model will have more spread out sample shapes and therefore the distance to the closest one will be greater, so the measure will be larger when it should be smaller. This is not a measure of generality, but rather of specificity.

Another approach for handling this problem and measuring generality is the one used in Section 7.5. In this approach the model is built as in the standard approach but no optimized correspondences between the left-out shape and the model generated shape are used. Instead arclength parameterization is used for the shapes when calculating the approximation error. This makes it possible to get some measure of the generality without including the left-out shape in the optimization. However, since in some cases arclength will not be close to correct correspondence, this measure, while better than nothing, can be wrong.

8.2.2 Compactness

One problem with the compactness measure defined in Equation (2.38) is that it does not take into consideration the value of gathering the variance in as few modes as possible.

In Figure 8.2 the compactness of two models are plotted over the number of modes n_m. The total variances of the two models are equal. For model A all variance is concentrated in the first shape mode. The graph for model A therefore goes up to the total variance in the first mode. For model B the variance is distributed equally over all modes. According to the quantitative definition of the compactness measure and the strategy for comparing models in Section 2.6, model B is more compact than model A, whether we include all modes or only sum up to some other n_m or whether we look at which curve is below the other. However, from the qualitative definition, by compactness is meant as little variance as possible in as few modes as possible, so from that definition the conclusion is that model A is more compact than model B, since they have the same total variance but model A captures it in one mode.

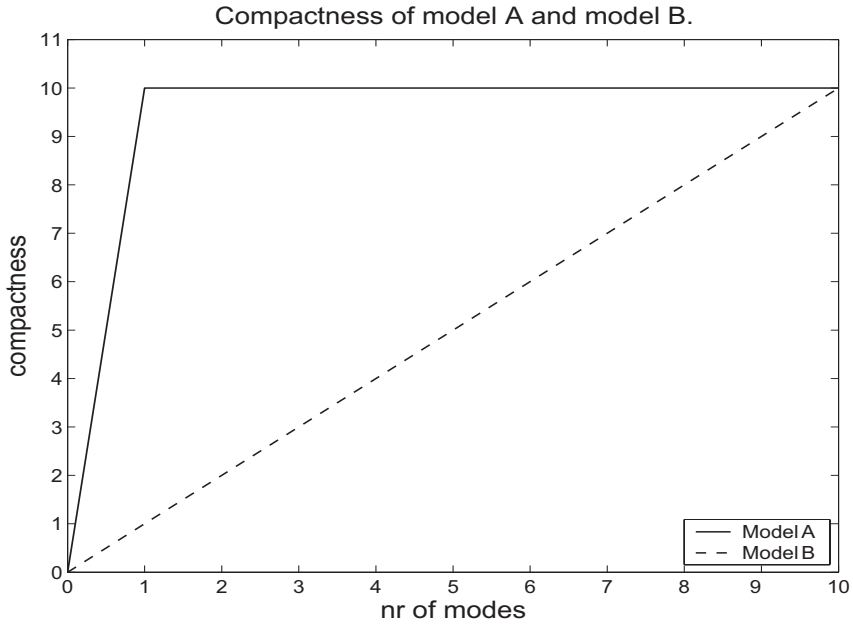


Figure 8.2: The compactness curve of model A is above that of model B. So model B is therefore more compact judging from the measured quantity. However from the qualitative definition model A is more compact than model B, since it captures all the variation in one mode.

This example may be even clearer if we let the total variance be very slightly (ϵ) greater for model A and if we let the variance be distributed over 1000 modes instead of 10 in model B. The compactness curve for model B is then strictly below the one for model A, even when using all modes. However, this ϵ-improvement in total variance is of course not worth using a thousand modes instead of one. Therefore we can conclude that the compactness measure in Equation (2.38) does not capture the notion of compactness very well.

Another thing to consider is that the compactness measure favors shapes parameterized so that they can be described as simply as possible. This approach works well for finding approximate correspondences but there is also the risk that these simple descriptions come at the cost of loosing real information about shape variation. This should then ideally lead to a bad generalization measure, since the model should not be able to accurately describe new shapes, but as described in the generality section, the left out shape is often included in the parameterization optimization together with the training set in order to decide a parameterization and then this shape is also described in a simplified manner and the loss of shape information is not detected. Actually, this can also be a problem in general when optimizing MDL which can be understood when noticing the similarity between the formulas for compactness (2.38) and for MDL (5.16). Of course, this similarity also means that the specified compactness measure is biased towards MDL and similar methods.

8.2.3 Specificity

For simplicity of illustration, assume first that the norm used for specificity is $||f(x)|| = \int |f(x)|dx$ (or $||f(x)|| = \sum |f(x)|$) , i.e. the L^1-norm (or l^1). Assume two models built from a training set consisting of rectangles as shape A and shape B in Figure 8.3. Also assume that Model 1 generates samples as shape C and model 2 generates samples as shape D. These models will have equal specificity measure, since C and D have equal distance to the closest shape in the training set. However, shape C belongs to the shape class, but shape D does not. The conclusion is that disturbances with zero mean will not be detected by the specificity measure.

To explain the above in a more precise way, consider the training set

$$\mathcal{T} = \{\mathbf{y}_1, \ldots, \mathbf{y}_{n_s}\} \tag{8.3}$$

consisting of rectangles with equal width w and varying height $h \in \{h_1, \ldots, h_{n_s}\}$. For simplicity, assume $w = 1$. Assume that model 1 can generate all rectangles with width w and height $h \in (0, \infty)$. Call this space Ω_1 and call its probability distribution μ_1. We can identify Ω_1 with $\mathbb{R}^+$. Assume that model 2 can generate the same rectangles and also any shape that can be constructed by adding functions with mean zero and bounded amplitude to the top side of the rectangle (in the example in Figure 8.3 the added function is sin). Call this space of shapes, generated by model 2, Ω_2 with probability distribution

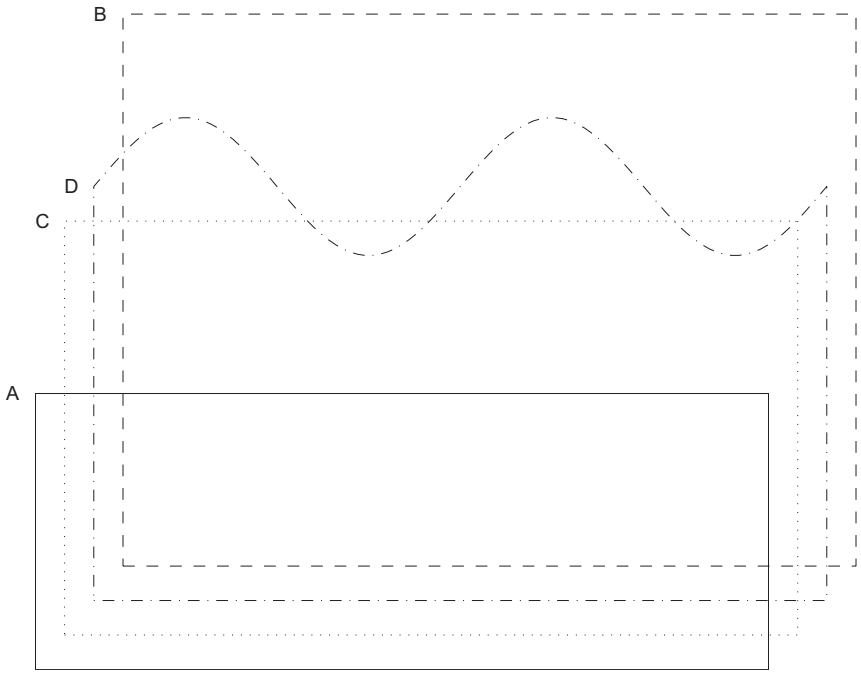


Figure 8.3: Shape A and shape B are in the training set. One model generates examples like shape C and another model generates shapes like D. These models are equal with regard to the given formula for specificity.

μ_2. In order to describe top side functions for model 2, let $F = C_0([0,1], \mathbb{R})$, i.e. the space of continuous functions from $[0,1]$ to $\mathbb{R}$. Let suitable functions be

$$B_{m,a} = \{f \in F : \sup_{x \in [0,1]} |f(x) - m| \leq a, \int_0^1 f dx = m\}. \tag{8.4}$$

In order to keep track of the distance to the closest rectangle, define

$$a(h) = \inf_{h_i \in \{h_1, \ldots, h_{n_s}\}} |h - h_i|. \tag{8.5}$$

We denote $\Omega_h = B_{h,a(h)}$ and note that

$$\Omega_{h_1} \bigcap \Omega_{h_2} = \emptyset \tag{8.6}$$

for all $h_1, h_2 \in \Omega_1, h_1 \neq h_2$. Now, let

$$\Omega_2 = \bigcup_{h \in \Omega_1} \Omega_h. \tag{8.7}$$

The shape space generated by model 2 is an extension of the shape space generated by model 1, $\Omega_1 \subset \Omega_2$. Model 2 therefore clearly is less specific than model 1.

Let $\pi : \Omega_2 \to \Omega_1$ be determined by the mapping defined on the topside functions as

$$\pi : f \mapsto \int_0^1 f dx \ . \tag{8.8}$$

These definitions are illustrated in Figure 8.4.

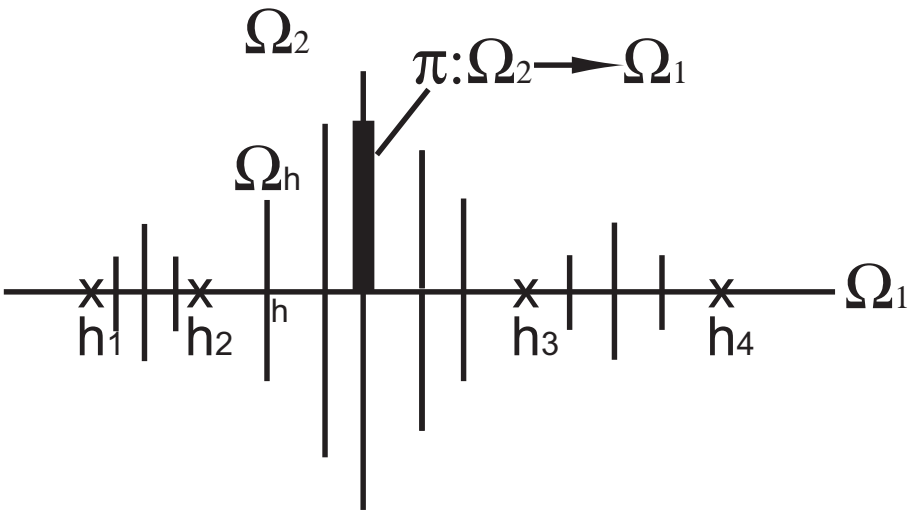


Figure 8.4: The space Ω_2.

Now, assume that, loosely speaking (thinking for example of Ω_2 as a subset of $\mathbb{R}^n$ and assuming probability density functions p_1 and p_2 corresponding to μ_1 and μ_2), we want

$$\int_{\Omega_h} p_2(\mathbf{z}) d\mu_0(\mathbf{z}) = p_1(\mathbf{h}) \ , \tag{8.9}$$

where Ω_h is the space consisting of shapes generated from the shape $\mathbf{h}$, which is a rectangle with height h, i.e. Ω_h are rectangles where the top side has been replaced with some

continuous function so that the mean is still at h. This can be stated as demanding that μ_2 is normalized such that

$$\int_{\Omega_2} \tilde{\phi} \circ \pi(\mathbf{z}) d\mu_2(\mathbf{z}) = \int_{\Omega_1} \tilde{\phi}(\mathbf{h}) d\mu_1(\mathbf{h}) \ , \tag{8.10}$$

for all functions $\tilde{\phi} \in C_0(\Omega_1, \mathbb{R})$.

Let's now evaluate the specificity measure for model 2 using the general definition (2.37). This gives the specificity measure as

$$\int_{\Omega_2} \inf_{\mathbf{y}_j \in \mathcal{T}} ||\mathbf{y}_j - \mathbf{z}|| d\mu_2(\mathbf{z}) \ . \tag{8.11}$$

Define $\phi(\mathbf{z}) = \inf_{\mathbf{y}_j \in \mathcal{T}} ||\mathbf{y}_j - \mathbf{z}||$ and note that because of how Ω_2 is constructed $\phi(\mathbf{z}) = \tilde{\phi} \circ \pi(\mathbf{z})$ where $\tilde{\phi}(\mathbf{h}) = \inf_{\mathbf{h}_j} |\mathbf{h}_j - \mathbf{h}|$.

Then use (8.10) to show that (8.11) is equal to

$$\int_{\Omega_1} \inf_{\mathbf{h}_j} |\mathbf{h}_j - \mathbf{h}| d\mu_1(\mathbf{h}) \ . \tag{8.12}$$

and this is the specificity measure for model 1.

So the specificity measures for model 1 and model 2 are equal but as explained above, model 2 is clearly less specific than model 1. In the calculations above the shapes clearly do not have to be rectangles. Actually Ω_1 can be any shape space and Ω_2 can be any extension formed by adding integrable functions with mean zero and suitably bounded amplitude to some part of the shape.

Using the standard L^2-norm the above proof is not quite as intuitive as above and the space Ω_h gets more complicated, but still we get the same specificity for the space Ω_1 as for Ω_2.

To illustrate the problem with the specificity measure further another example is given in Figure 8.5, where a number of shapes are shown. For shape A, B and C landmark 40 is at the left corner of the bump and landmark 50 is at the right corner of the bump. Landmark 10, 20, 30 and 60 are at the corners of the rectangle in clockwise order. Between these landmarks the landmarks are placed by arc length parameterization. Shape D is constructed by taking shape B and projecting down the bump onto the rectangle. Assume now that two different models of box-bumps are built in some way from shape A and shape B. Shape C also belongs to the class of box-bumps but shape D does not. Say that model 1 generates shapes like Shape C and model 2 generates shapes like shape D. Look at the terms in the specificity definition (2.34). The term for model 2 that comes from shape D will be less than the term for model 1 that comes from shape C. This implies according to (2.34) that model 2 is more specific than model 1, but this is clearly wrong.

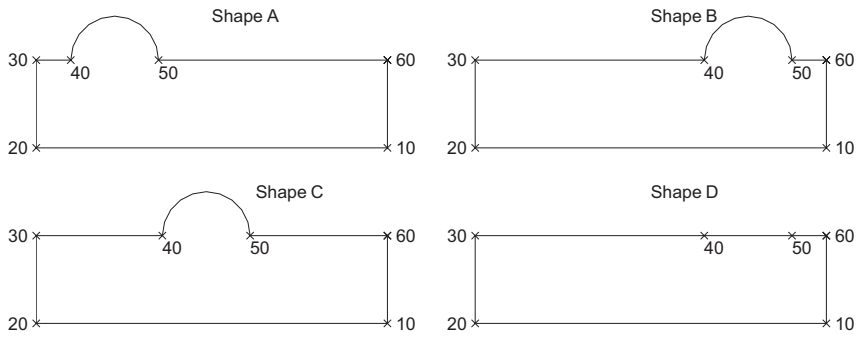


Figure 8.5: Two models are built from shape A and shape B. It turns out that the model that generates shapes like shape D has a better specificity than the model that generates shapes like shape C, but shape D does not belong to the shape class whereas shape C does.

So we can conclude that it is difficult to design a specificity measure that takes into consideration in what way shapes are similar to each other and that the quantitative measure defined in Equation (2.34) does not succeed. The measure (2.34) is based on the idea that the model should only be able to represent shapes that look like the ones in the training set. This might be a reasonable approach if the training set is very large but in practice it is often rather small. The model must then be able to represent not only shapes that look like the ones in the training set but also shapes that deviate from the training shapes, but do so in a way that means that they still belong to the shape class. The standard specificity measure can easily give worse error for shapes that belong to the shape class than it does for shapes that do not belong to the shape class.

8.2.4 Conclusions

We have illustrated serious problems with the standard measures. As an extreme example it is possible to get perfect compactness, specificity and generality (zero) by sampling in approximately one point on all shapes, but this of course is a useless set of correspondences. Some problems with compactness were noted already in [31]. The commonly used measures for specificity, compactness and generality can give a rough idea about the quality of a model in some cases, but in other cases, the conclusions drawn from these measures can be wrong. Later in this chapter, this will be further illustrated with experiments.

8.3 Ground Truth Correspondence Measure (GCM)

If ground truth correspondences are known for a training set it is possible to measure the quality of the automatically established correspondences (used for building the model) by measuring how well the produced correspondences agree with the known ground truth. Since ground truth is required, this measure can of course not be used to *locate* correspondences, but instead to *evaluate algorithms* for establishing such correspondences.

Note that it is often necessary to use some apriori ground truth information to evaluate correspondences since the correspondence problem in general is not solvable without it. The geometric information in the shape object may not be enough, other types of knowledge about the objects are often needed. Then automatic algorithms based only on geometric information may be differently good at estimating true correspondences from the geometric information alone. Benchmarking with ground truth correspondences is then necessary in order to evaluate the algorithms. The following example illustrates this.

In Figure 8.6 three shapes are shown. Assume that shape A and Shape B belong to the training set. A reasonable assumption would be that there is one shape variation mode corresponding to the position of the bump. Now, assume shape C is also in the training set. It is now reasonable to assume two shape modes, one for each bump telling the height of each bump. The conclusion is that for the training set consisting of shape A and shape B the correspondence problem is not solvable without ground truth information. Therefore in order to be able to measure the correctness of correspondences located automatically a database of ground truth correspondences is necessary.

8.3.1 Basic Definition

In order to measure the quality of the correspondences produced by an algorithm for automatic correspondence determination, datasets with manually located landmarks and synthetic datasets with known corresponding points is used. For synthetic examples these marks are exact but for manually placed marks there is of course a subjective element and also the introduction of a small error is inevitable.

Let the parameterizations γ_i of the shapes $\mathbf{x}_i$ $(i = 1, \ldots, n_s)$ be optimized by the algorithm that is to be evaluated. Then, for every shape $\mathbf{x}_i$ together with its ground truth points $\mathbf{g}_{ij}$ $(j = 1, \ldots, n_g)$, find t_{ij} so that

$$\mathbf{x}_i(\gamma_i(t_{ij})) = \mathbf{g}_{ij}, \;\; i = 1, \ldots, n_s, \; j = 1, \ldots, n_g \;\; . \tag{8.13}$$

This means that the parameter values that correspond to the ground truth points on the shape are calculated. Formally $t_{ij} = \gamma_i^{-1}(\mathbf{x}_i^{-1}(\mathbf{g}_{ij}))$. Now, keeping i fixed for a moment, for every shape $\mathbf{x}_k$ $(1 \leq k \leq n_s, k \neq i)$ use the same parameter values t_{ij} computed for $\mathbf{x}_i$. The points produced should be close to the corresponding ground truth points of $\mathbf{x}_k$, if the parameterization functions represent correspondences of high

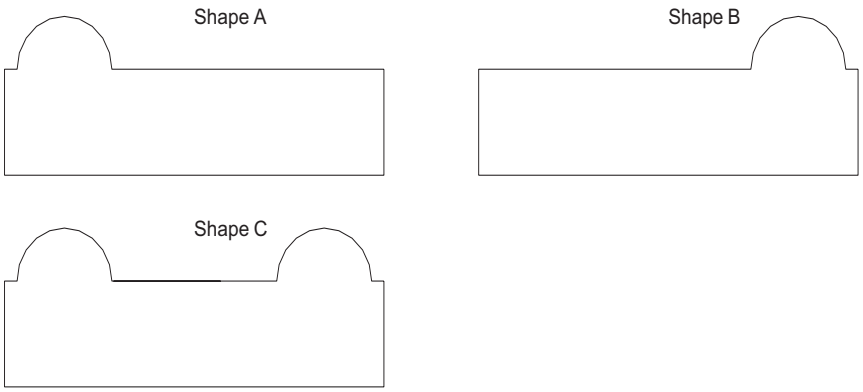


Figure 8.6: An illustration of one difficulty with the correspondence problem showing that ground truth is needed to evaluate the correctness of correspondences. If only shape A and shape B are taken into account, it is reasonable to assume one shape mode, namely the bump position. If also shape C is in the training set, it is natural to assume two shape modes, namely the height of the bumps.

quality. That is, $\mathbf{x}_k(\gamma_k(t_{ij}))$ should be close to $\mathbf{g}_{kj}$. The deviation is measured by

$$e_{ijk} = d\left(\mathbf{x}_k(\gamma_k(\gamma_i^{-1}(\mathbf{x}_i^{-1}(\mathbf{g}_{ij})))) \ , \ \mathbf{g}_{kj}\right) \ , \tag{8.14}$$

where d is some suitable metric. Finally we do this starting from every shape in the set, using each of its ground truth points and measure correspondence to each other shape in the set and take the mean error so we get

$$GCM = \frac{1}{n_s(n_s-1)n_g} \sum_{i=1}^{n_s} \sum_{j=1}^{n_g} \sum_{k \in K} d\left(\mathbf{x}_k(\gamma_k(t_{ij})) \ , \ \mathbf{g}_{kj}\right) \ , \tag{8.15}$$

$$K = \{1, \ldots, i-1, i+1, \ldots, n_s\} \ .$$

where t_{ij} is the parameterization parameter value for the ground truth point j on shape i, i.e.

$$t_{ij} = \gamma_i^{-1}(\mathbf{x}_i^{-1}(\mathbf{g}_{ij})) \ . \tag{8.16}$$

The constant n_s is the number of shapes and n_g is the number of ground truth points on each shape. Any metric d could be used, but here $d(a, b)$ is the length of the shortest path between the point a and the point b along the shape, which has been normalized to have area one. Locally (usually also globally) the shortest path is a geodesic. Apart

from the advantage of measuring the error along the shape, this approach also gives scale invariance. On curves, the metric d corresponds to the arc-length distance on curves normalized to length one. The procedure described above is illustrated in Figure 8.7.

In [94] a somewhat similar measure is used to evaluate correspondences on registered surfaces. However, there the distance between the corresponding points is measured as the Euclidean distance. Also, the distance is measured between points on a deformed template and points on the target surface whereas here we focus on group-wise correspondence.

8.3.2 GCM Measure with Mahalanobis Distance

For synthetic examples ground truth marks are exact but for manually placed ground truth points on natural shapes there is of course a subjective element and also the introduction of a small error is inevitable.

Therefore, for natural shapes, statistics about the ground truth points could be taken into account.

Let a number of people mark corresponding ground truth points on the same dataset. One example, which has been marked by 18 people, from the database can be seen in Figure 8.8.

Means and variances can then be calculated and the norm used to calculate GCM can then be the Mahalanobis normalized distance,

$$GCM = \frac{1}{n_s(n_s-1)n_g} \sum_{i=1}^{n_s} \sum_{j=1}^{n_g} \sum_{k \in K} \frac{d\left(\mathbf{x}_k(\gamma_k(t_{ij}))\ ,\ \overline{\mathbf{g}}_{kj}\right)}{\sigma_{kj}} \quad , \tag{8.17}$$

$$K = \{1, \ldots, i-1, i+1, \ldots, n_s\} \quad ,$$

where $\overline{\mathbf{g}}_{kj}$ is the mean and σ_{kj} is the standard deviation for landmark j on shape k.

8.3.3 Comments

Note that it is not the case that we optimize the positions of a set of landmarks like the centers of the eyes of a face that then can be compared to ground truth. There are no such landmarks during optimization. The optimization works directly on the parameterization functions to establish dense correspondence. The ground truth points are used to evaluate the established parameterization functions but the optimization does not move around landmarks trying to place them at the ground truth points. The correspondence is completely described by the parameterization functions. By mapping the ground truth points from one shape via the parameterization functions to another shape we can measure the quality of the correspondence. Note that there are infinitely many sets of parameterization functions describing the same correspondences. This is the reason why it is necessary to map ground truth points between shapes.

Although the GCM measure is based on a finite set of points on the shape, if these points are chosen carefully they will represent most of the valuable information of the

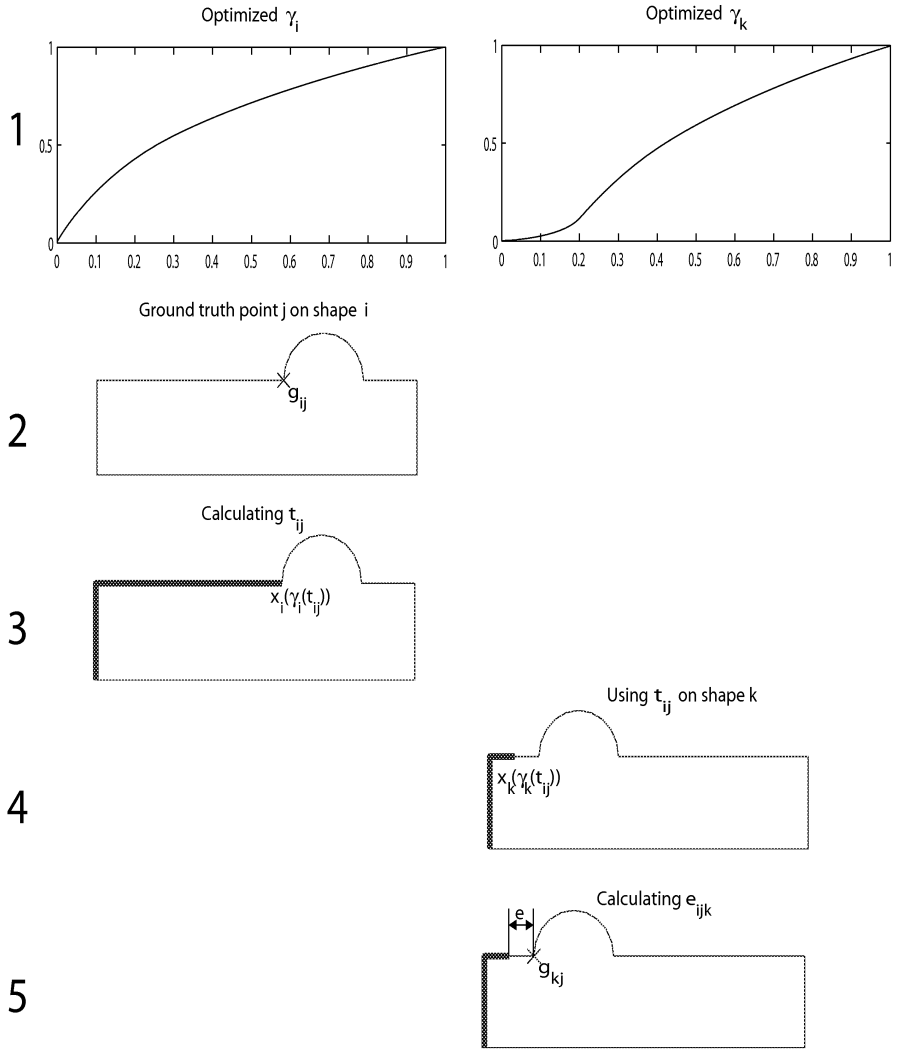


Figure 8.7: Illustration of GCM calculation: (1) The optimiser determines the parameterisation functions $\gamma_i, i = 1 \ldots n_s$, (2) Given ground truth point j on shape i, (3) calculate the corresponding parameter t_{ij}. (4) Use t_{ij} on shape k. (5) Calculate the distance e_{ijk} from $\mathbf{x}_k(\gamma_k(t_{ij}))$ to ground truth point j on shape k.

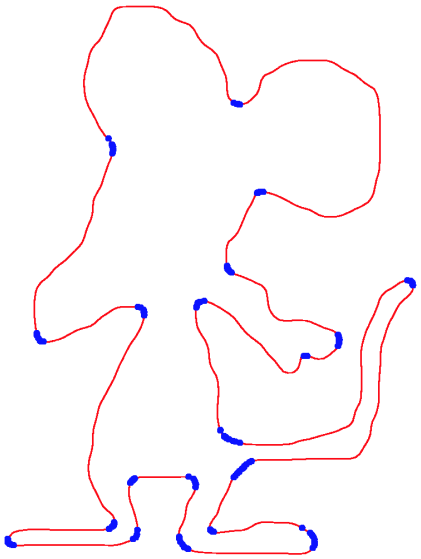


Figure 8.8: One example of the rat shapes with ground truth points set by 18 people. Note that there is different variance for different ground truth points.

shape. In handwriting recognition such points are called core points, see [127]. Typically the ground truth points would be chosen at points where the shape is characteristic for all shapes in the shape class.

This ground truth correspondence measure measures the precision of the established correspondences, and thereby the extent to which the shape model represents the true shapes. Using similar techniques other ground truth measures can be defined. For example, using ground truth segmented images a ground truth segmentation measure can be defined. In medical applications the diagnose could be used to define a ground truth classification measure.

8.4 Experimental Validation of GCM

8.4.1 Setup

In this section, GCM, generality, compactness and specificity are evaluated experimentally. The experiments show that GCM together with its database of ground truth landmarks is a better measure than the standard evaluation methods. The experiments in this

section treat two datasets.
Silhouettes: The silhouette dataset consists of 22 contours of silhouettes. 22 persons were photographed using a digital camera. The silhouettes were then extracted using an edge detector.
Box-bumps: A synthetic shape consisting of a box with a bump.
In Figure 8.18 one example of a silhouette shape and one example of a box-bump shape can be seen. For the optimization, seven parameterization parameters (controlling seven basis functions) and 128 sampling points were used for each shape.

8.4.2 Box-bump Experiment

The first experiment was to start from correct correspondences and then optimize the parameterizations so as to minimize MDL. Synthetic box-bump shapes, consisting of a rectangle with a bump on different positions on the top side, were used for this test, since we know the true correspondence here. The value of the minimum description length (MDL) (5.16) and the ground truth correspondence measure (GCM) (8.15) over the number of iterations is plotted in Figure 8.9.

It is interesting to note here that the GCM increases (gets worse) as MDL decreases (gets better). The minimum, when the parameterization is optimized with MDL as cost function, does not correspond to true correspondences in this case. This is illustrated in Figure 8.10, where it can be seen that optimizing MDL, starting from true correspondences has resulted in worse correspondences. The plot shows 4 box-bump shapes zoomed in on the bump. Since the plot does not show the actual curve of the shape but instead straight lines between the sample points on the shapes there appears to have been shape changes at the corners. In an attempt to ensure that the algorithm considers the original shapes properly the number of sampling points could be increased but the sample points will still not be exactly at the corners. Also, the main problem with ensuring that the entire shape is taken into consideration is not the number of sample points but instead the relative weight given to different parts of the shape depending on the parameterization function, independent of the number of sample points.

In Figure 8.11 the compactness and specificity measures indicate that the MDL optimized model has higher quality. In this case we started from ground truth and as can be seen in Figure 8.10 the correspondences are worse for the optimized model. So we can conclude that compactness as defined in Equation (2.38) and specificity as defined in Equation (2.34) do not correlate with correct correspondences.

In Figure 8.12 the specificity is evaluated visually. The upper model is built from ground truth correspondences and the lower model is built from MDL-optimized correspondences. The plots show the mean-shape plus three standard deviations of the first two shape modes. Since the data in this case only has one shape mode it is enough to examine the first two modes as in this plot. The model built from ground truth correspondences (Model 1) is clearly more specific than the MDL-optimized model (Model 2). The slight

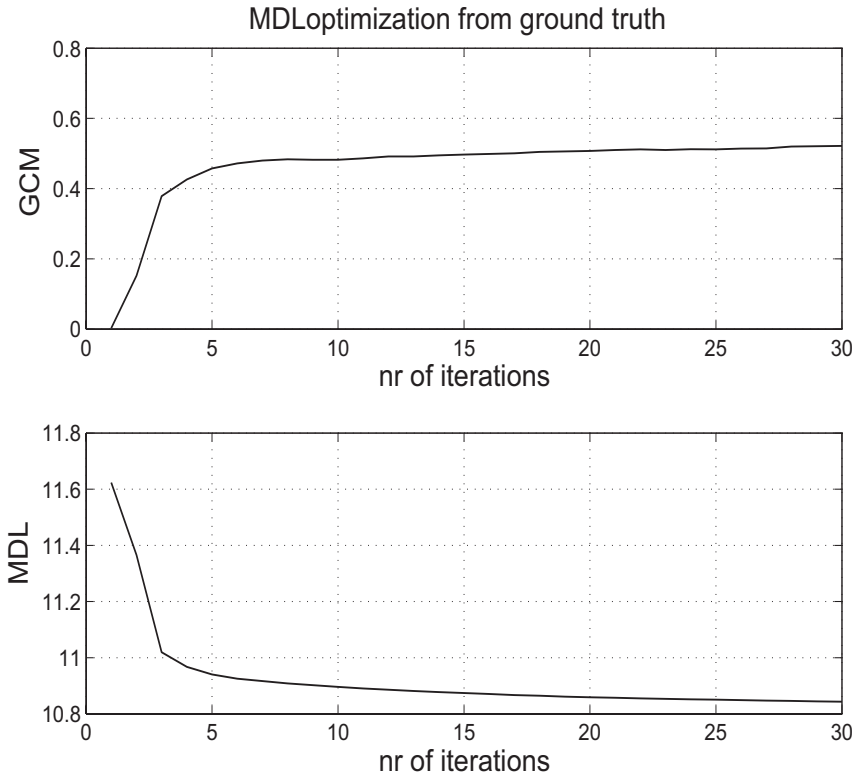


Figure 8.9: The ground truth correspondence measure (GCM) and the minimum description length (MDL) plotted over number of iterations.

distortion in model 1 is due to the alignment of the shapes. The Procrustes alignment introduces some nonlinearities in the shape modes. From Figure 8.12 it can be concluded that the model built from better correspondences has better specificity, contrary to the conclusion of the plot of (2.34) in Figure 8.11. So also in practice the specificity measure defined by (2.34) does not work. The problem with the specificity measure is that if the training set is limited (which is often the case) it can not be assumed that all shapes of a class are close to one of the shapes in the training set.

Summing up this experiment, it is concluded that the compactness and specificity measures defined in Equations (2.38) and (2.34) do not correlate well with the quality of the correspondences in general and the specificity measure in particular contradicts visual inspection of specificity and fails to measure the desired property.

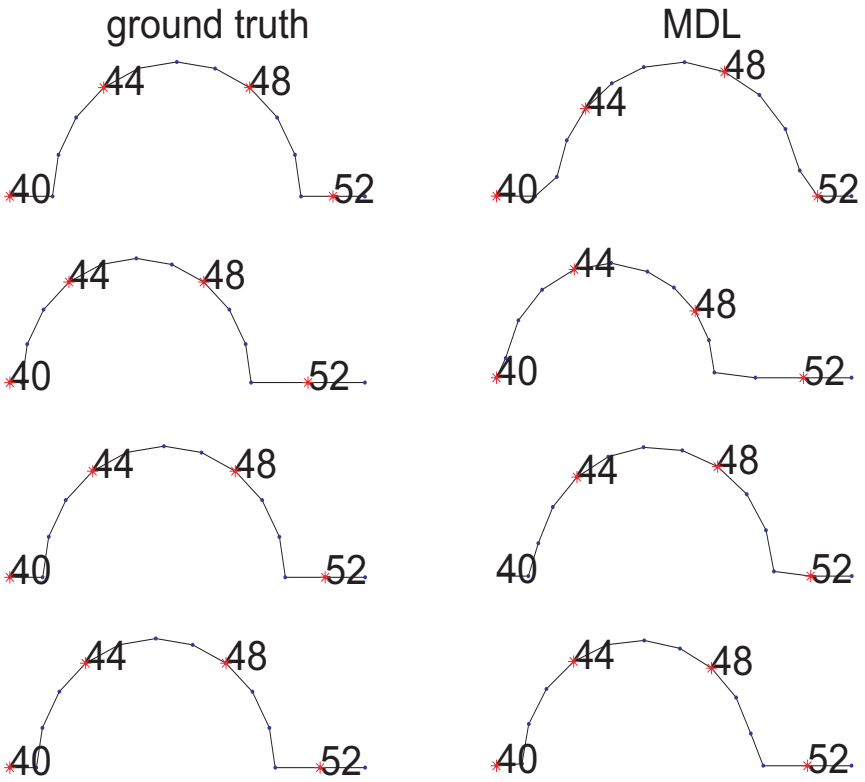


Figure 8.10: To the left the true correspondences for the box bumps can be seen. To the right the correspondences established by minimizing MDL are shown. The figure shows the bump part of the shapes. Compare for example the location of samplepoint 51.

In Chapter 10 of Davies' thesis [31] a somewhat similar experiment indicates that MDL does succeed in finding close to optimal correspondences on the synthetic box-bump data set resulting in good specific shape modes. However, in that experiment the shapes are not aligned using Procrustes to minimize Euclidean distances like they are in this one. MDL near ground truth was also examined in [31] (Figure 6.5) by showing that MDL increases monotonously in mean when applying random perturbations of Gaussian noise with increasing magnitude. However this does not show that the optimum of MDL is located at ground truth, since it does not show that there is no path that leads to lower MDL than that at ground truth correspondences. In fact, it is unlikely that the path to the optimum would be found by random perturbations. So these experiments do not

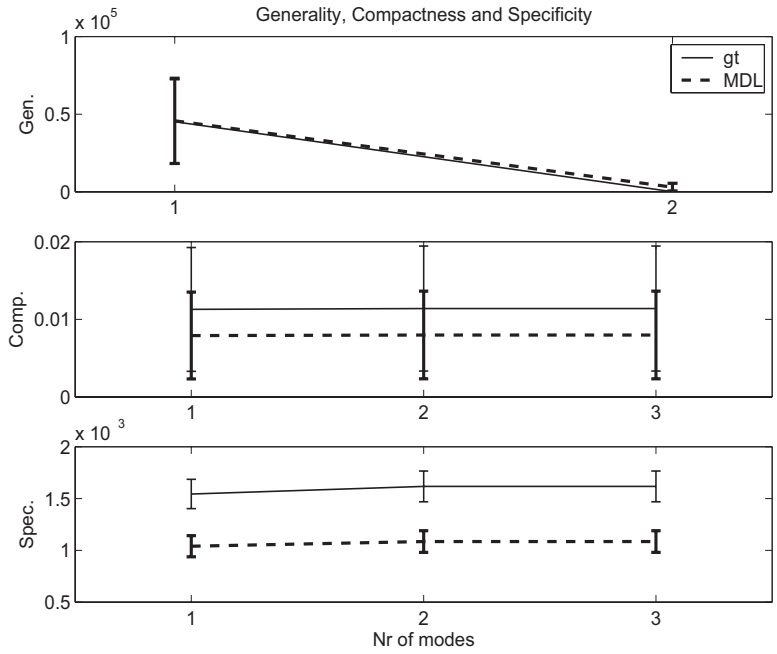


Figure 8.11: Generalization, compactness and specificity of the ground truth box-bump model and the MDL-optimized box-bump model.

contradict each other. The point with the experiment presented in this section is not to evaluate whether MDL succeeds or fails, but rather to examine the evaluation measures.

8.4.3 Silhouette Experiment

In the second experiment silhouette shapes initialized with arc-length parameterization were used. The parameterizations were optimized with respect to MDL [35] until convergence. Then the optimization was continued with respect to MDL plus a curvature cost [136] until convergence.

Figure 8.13 shows the resulting correspondences on the part of the shapes corresponding to the eye. The plots show sample point 25 to 40. Anatomically this shows the end of the forehead and the beginning of the nose of a person looking to the left. The nose begins approximately at sample point 34 in the bottom row. The correspondences are clearly better when using curvature. Other parts of the curves are similar.

The top plot in Figure 8.14 shows how GCM first goes down and converges while

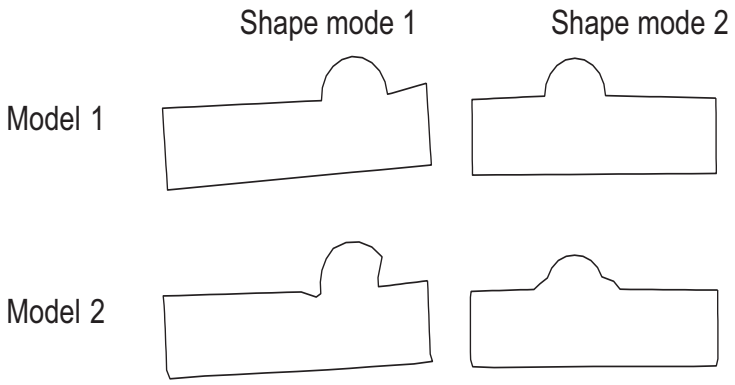


Figure 8.12: Mean-shape plus three standard deviations of the first two shape modes for models built from ground truth correspondences (Model 1) and MDL-optimized correspondences (Model 2).

optimizing MDL and then when curvature cost is added GCM clearly decreases and converges at a lower level. The middle plot shows how MDL first decreases as it is minimized, but then when MDL + curvature cost is minimized in the second part MDL increases. So GCM captures an improvement in correspondences that MDL misses. In Figure 8.15 it can be seen how the measures of generality, compactness and specificity all indicate that the model without curvature cost has higher quality. So the standard measures can not be used to measure correspondence quality. This experiment shows that GCM captures an improvement in correspondences that generality, compactness, specificity and MDL all miss.

The above experiment on silhouettes was performed on 4 shapes. A similar experiment using all the 22 silhouette shapes was also done and the results were similar, as can be seen in Figure 8.16 and Figure 8.17.

To conclude, both in theory and in practice the standard measures have severe problems.

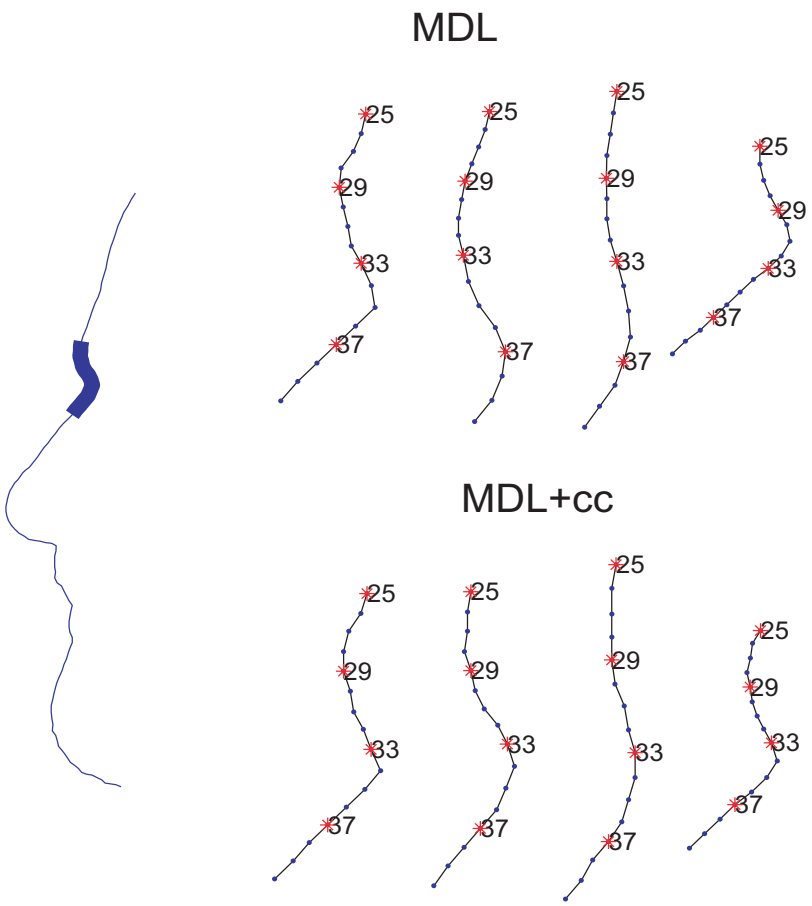


Figure 8.13: Corresponding sample points on parts of silhouettes optimized with MDL and MDL plus curvature cost. To the left the part zoomed in on is marked bold on a whole silhouette.

8.5 Benchmarking using GCM

8.5.1 Database

First a database of eight shape classes (one synthetic dataset and seven natural datasets of curves extracted from real images) was collected. The first five shape classes (sharks, birds,

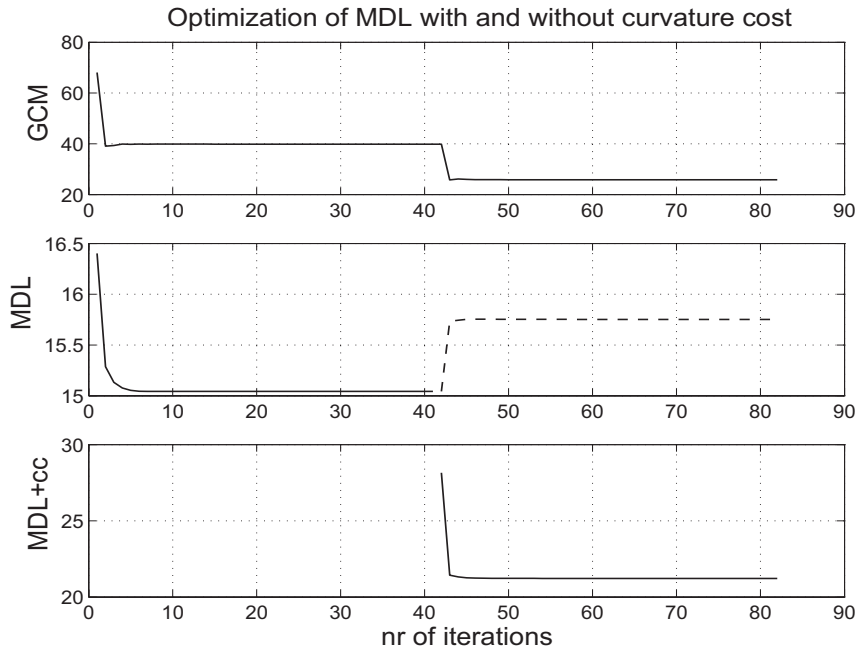


Figure 8.14: Optimization of MDL and MDL + curvature cost.

flight birds, rats, forks) are curves generated from images in the LEMS database from Brown University and each of these shape classes consists of 13-23 shapes [114, 117]. The sixth and eighth shape classes are the silhouette, and box-bump shapes from the previous section. The seventh dataset consists of 23 contours of hands.

Figure 8.18 illustrates the shape classes. All the natural examples have been marked with ground truth points by 18-25 people. In total the database consist of 28518 (not including the synthetic dataset) ground truth landmarks manually set. The box-bumps are synthetic with a total of 1464 ground truth points on 24 shapes.

8.5.2 Bench-marked Algorithms

The following algorithms have here been bench-marked using GCM:
arc-length: All the curves are sampled using arc-length parameterization.
MDL: As we have seen in previous chapters, one popular way to solve the correspondence problem in shape modeling is to apply the MDL-technique [33, 34]. See Chapter 5 and for example [35].
MDL+Cur: In [136] it was shown that a curvature cost can be added to the description

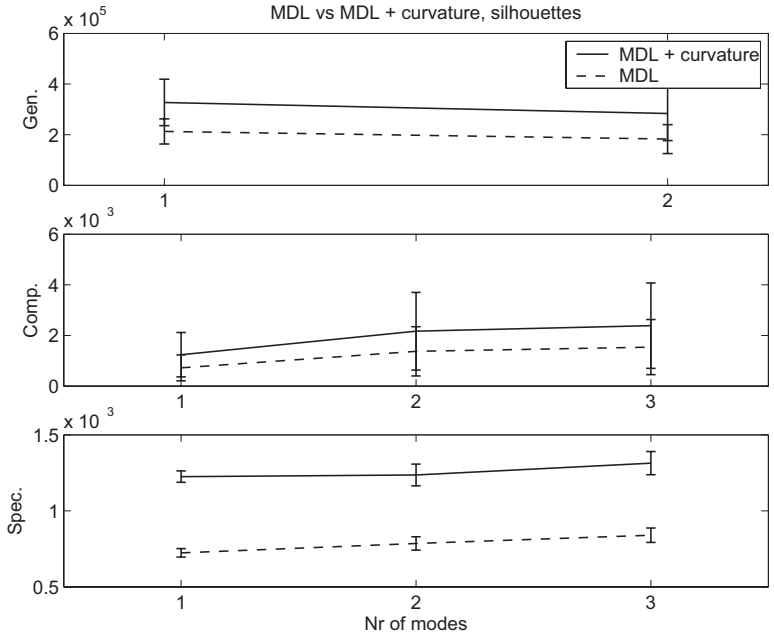


Figure 8.15: Generalization, compactness and specificity of the MDL silhouette model and the MDL + curvature cost silhouette model.

length to improve the correspondences produced by the optimization (named MDL+Cur here). The curvature cost is a measure of the variation in the set of curvature functions at corresponding points along the shapes. With good correspondences the curvature at corresponding points will be similar and so the variance and thereby the curvature cost will be low. The total cost function is then $E = E_{MDL} + KE_{cc}$, where E_{MDL} is the description length, K is a constant and E_{cc} is the curvature cost.

MDL+me: One of the shapes is used as a master example. Its parameterization function is held fixed during optimization. This is a technique aimed at avoiding clustering of the sampling points [35].

MDL+nodecost: Another technique for avoiding clustering of sample points. A node cost is introduced to ensure that parameterization node parameters should in mean be close to zero for each parameterization node. [136]

MDL+Par: In [76] and Chapter 7 MDL is used with a parameterization invariant scalar product and this gives better results (called MDL+Par in this chapter).

AIAS+MDL: In [43] a technique that formerly was used in stereo for finding correspondences is shown to work for shape models. This is called Affine Invariant Active

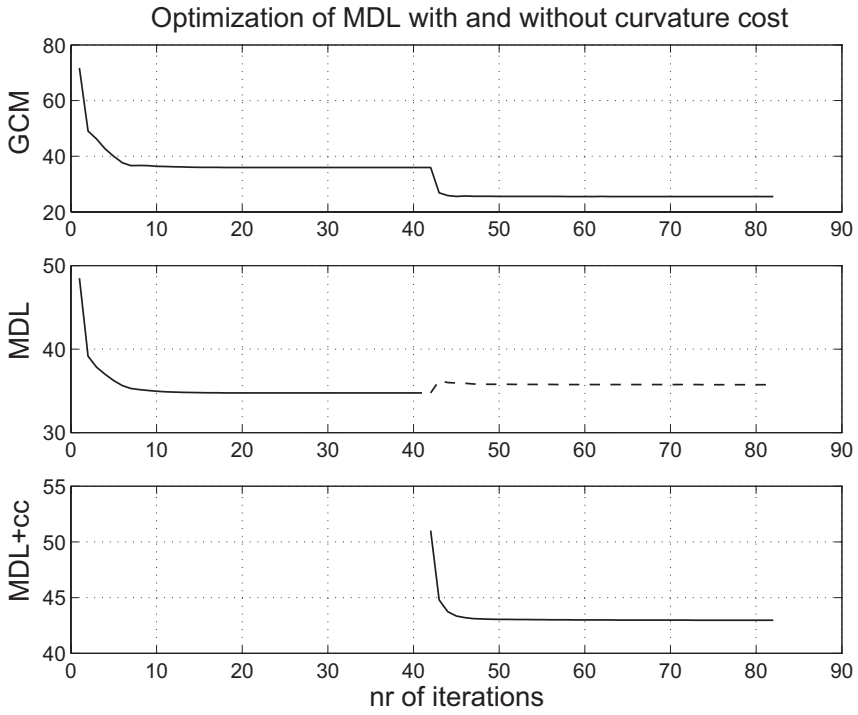


Figure 8.16: Optimization of MDL and MDL + curvature cost for 22 silhouettes.

Shape, AIAS. In this chapter an algorithm that minimizes MDL for an AIAS model (AIAS+MDL) is bench-marked.

AIAS+MDL+Cur: MDL is minimized for an Affine Invariant Active Shape model [43] with an added curvature cost.

Eucl: The parameterizations are optimized to minimize the Euclidean distance between corresponding points on all the shapes.

Eucl+Cur: The parameterizations are optimized to minimize a distance that is defined as the Euclidean distance plus a distance in curvature variation between corresponding points on all the shapes.

Cur: The curvature cost used in the algorithms above is here used by itself as a cost function to be minimized.

Handmade: Handmade models of all the datasets were built by a different person than the ones marking ground truth. This was done without knowing which anatomical points were used as ground truth. Parameterizations are determined by placing some correspond-

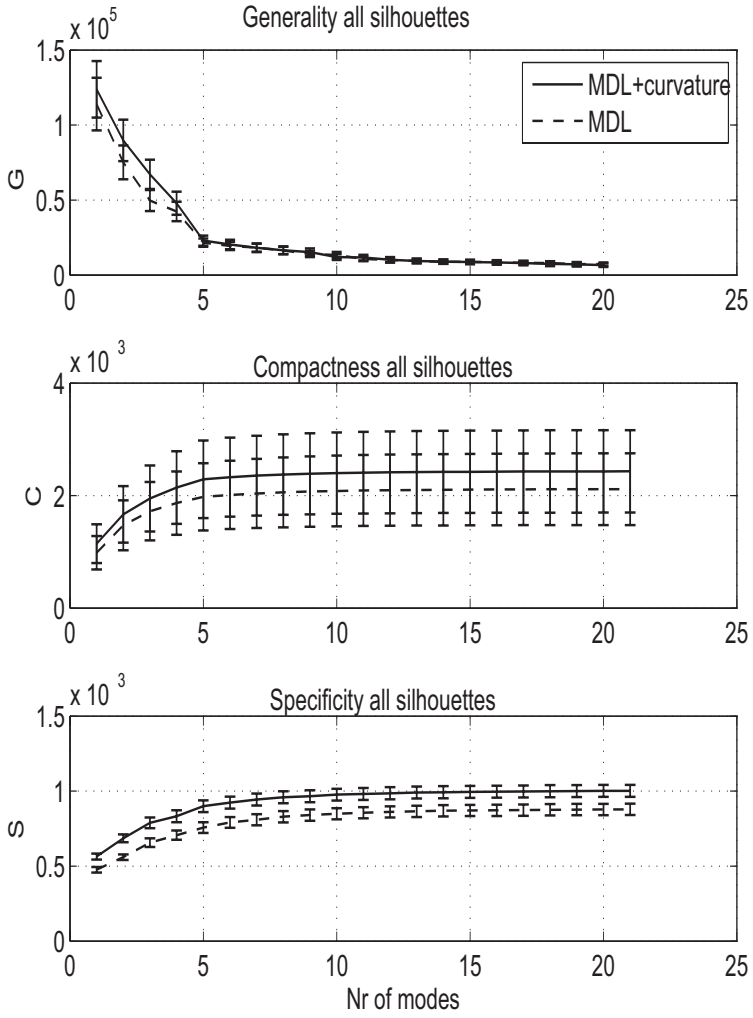


Figure 8.17: Generalization, compactness and specificity of the MDL silhouette model and the MDL + curvature cost silhouette model using 22 shapes.

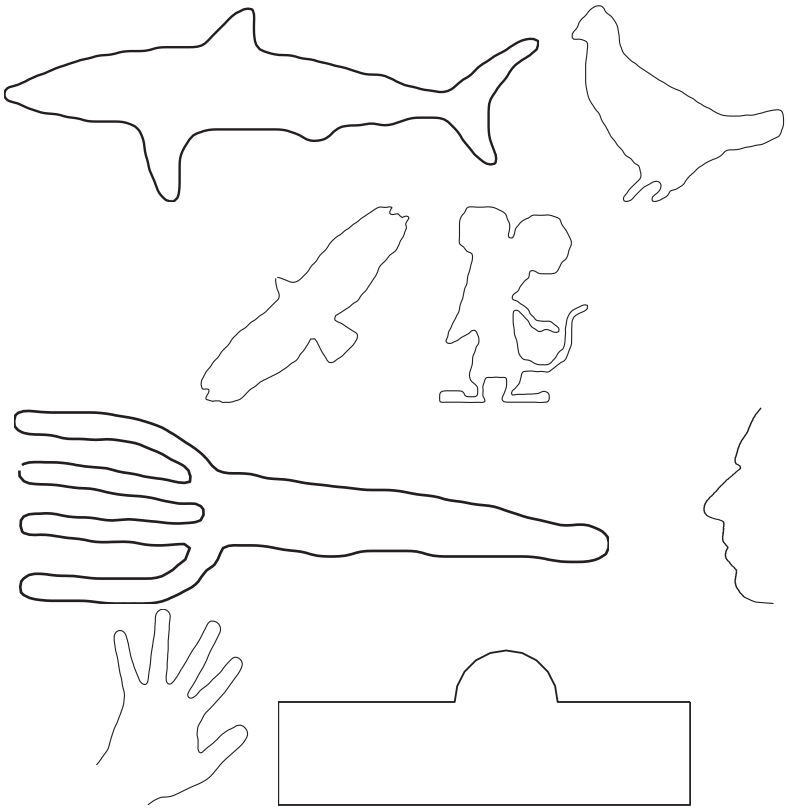


Figure 8.18: One example from each class of shapes in the database.

ing points on the shapes and using arc-length between these points.

8.5.3 Benchmarking

All tests were performed with 40 iterations in each algorithm and with 128 sampling points and 7 reparameterization function control variables for each shape.

Table 8.1 shows the remaining percentage of GCM (with Mahalanobis) after optimizing from arc-length (100% means GCM is equal to when using arc-length parameterization and 0% means perfect correspondences according to GCM).

AIAS+MDL+Cur is the algorithm that is best in mean. This algorithm succeeds especially well on the bird dataset (along with AIAS+MDL). MDL+Cur has the lowest median result. Since the median is a more stable measure and since MDL+Cur is the

Algorithm	sharks	birds	flightbirds	rats	forks	silhouettes	hands	mean	median	$boxbumps^*$
arc-length (GCM)	15.55	22.88	7.12	13.37	19.11	8.93	14.00	14.42	14.00	–
MDL (%)	27.21^{7}	65.17^{6}	56.33^{6}	29.03^{7}	19.41^{3}	46.23^{6}	17.60^{8}	37.28^{6}	29.03^{7}	29.20^{3}
MDL+Cur (%)	$\mathbf{21.51^{1}}$	79.93^{8}	$\mathbf{45.02^{1}}$	$\mathbf{27.02^{1}}$	23.26^{6}	22.54^{2}	14.80^{4}	33.44^{3}	$\mathbf{23.26^{1}}$	$\mathbf{17.42^{1}}$
MDL+me (%)	29.26^{9}	91.67^{9}	61.80^{10}	30.10^{8}	19.83^{4}	47.76^{8}	16.81^{7}	42.46^{8}	30.10^{8}	50.20^{5}
MDL+nodecost (%)	26.13^{6}	66.71^{7}	56.34^{7}	28.90^{6}	19.21^{2}	47.37^{7}	16.24^{6}	37.27^{5}	28.90^{6}	30.97^{4}
MDL+Par (%)	24.40^{5}	62.40^{5}	47.94^{4}	27.80^{3}	$\mathbf{18.28^{1}}$	39.88^{5}	14.17^{2}	33.55^{4}	27.80^{4}	85.80^{7}
AIAS+MDL	21.93^{3}	$\mathbf{23.38^{1}}$	57.69^{8}	28.37^{5}	20.37^{5}	38.81^{4}	14.57^{3}	29.30^{2}	23.38^{2}	75.93^{6}
AIAS+MDL+Cur (%)	21.61^{2}	23.73^{2}	47.73^{3}	27.42^{2}	23.92^{7}	26.12^{3}	$\mathbf{14.17^{1}}$	$\mathbf{26.38^{1}}$	23.92^{3}	111.00^{8}
Eucl (%)	43.76^{10}	60.02^{4}	58.85^{9}	36.92^{10}	27.32^{8}	129.74^{10}	26.39^{10}	54.71^{10}	43.76^{10}	116.82^{9}
Eucl+Cur (%)	28.84^{8}	55.18^{3}	53.54^{5}	35.03^{9}	28.59^{9}	103.57^{9}	18.26^{9}	46.14^{9}	35.03^{9}	118.88^{10}
Cur (%)	22.01^{4}	111.16^{10}	45.82^{2}	27.96^{4}	31.19^{10}	$\mathbf{21.12^{1}}$	15.84^{5}	39.30^{7}	27.96^{5}	17.60^{2}
Semiauto (%)	20.31	20.40	47.68	24.58	14.32	16.67	9.27	21.89	20.31	16.52
Handmade (%)	17.94	8.84	14.24	10.44	7.33	9.53	6.67	10.71	9.53	0.00

Table 8.1: The second row in the table shows the GCM (Mahalanobis normalized) for arclength parameterization of the different datasets. The following rows shows the percentage of GCM error (Mahalanobis normalized) left after optimizing from arclength parameterization. The upper index indicates the rank of this algorithm on this dataset. The winning algorithm on each dataset is bold. Since the box-bumps (*) are so easy to mark, only one person has marked this dataset and the GCM without Mahalanobis has been used.

best algorithm on three natural datasets and also performs best on the synthetic dataset, MDL+Cur is selected as the winning algorithm. There is no algorithm that is best on all datasets and no algorithm gives as good correspondences as the correspondences manually annotated.

In Figure 8.19 the correspondences on the legs of the birds (the most difficult dataset) are shown. Four birds out of fourteen were picked to illustrate the results. The differences are most easily seen on the legs. It is interesting to visually compare the results of the different algorithms on this dataset. In the columns of Figure 8.19, starting with the left column, are the results of the MDL+Cur algorithm, the Eucl+Cur algorithm, the AIAS+MDL algorithm and in the right column the handmade correspondences. Note that only a few points were placed by hand. The points in the figure are sample points based on the parameterization functions. MDL+Cur is the best overall algorithm, but as can be seen in Table 8.1, it is not good on the birds dataset. The Euclidean algorithm is a simple algorithm, but as can be seen in the table it gets decent results on this dataset. The AIAS+MDL algorithm is the best automatic algorithm on the birds dataset. It can easily be seen that MDL+Cur does not perform well on this dataset. Eucl+Cur also has problems that can be seen most easily on the first shape. AIAS+MDL is quite good and the handmade correspondences are even better. So the visual inspection confirms the results of the GCM-benchmarking.

For the winning algorithm, GCM was then used to pick optimal parameter values, such as number of sampling points, number of parameterization control variables and paramaters specific to the winning algorithm by evaluating GCM on the shark dataset. The algorithm was then run with these parameters on all datasets, which resulted in an even better algorithm.

8.5.4 Semi-Automatic Algorithm

Since handmade models turn out to be best, a semi-automatic algorithm was tested. Five shapes were manually marked to decide corresponding parameterizations and these were then kept fixed, while the rest of the shapes in the dataset were optimized one by one using MDL with curvature cost to fit the five fixed shapes. This resulted in an algorithm better than all the automatic algorithms, see Table 8.1.

8.6 Summary and Conclusions

For evaluation of the quality of shape models built from correspondences located automatically there exist a number of standard measures. These measures can give a rough idea about the quality of a model in some cases, but in other cases the conclusions drawn from these measures can be wrong. It is shown in this chapter that there is reason to question the ability of these measures to capture model quality. In this chapter these measures are analyzed both theoretically and experimentally on natural and synthetic data. Problems

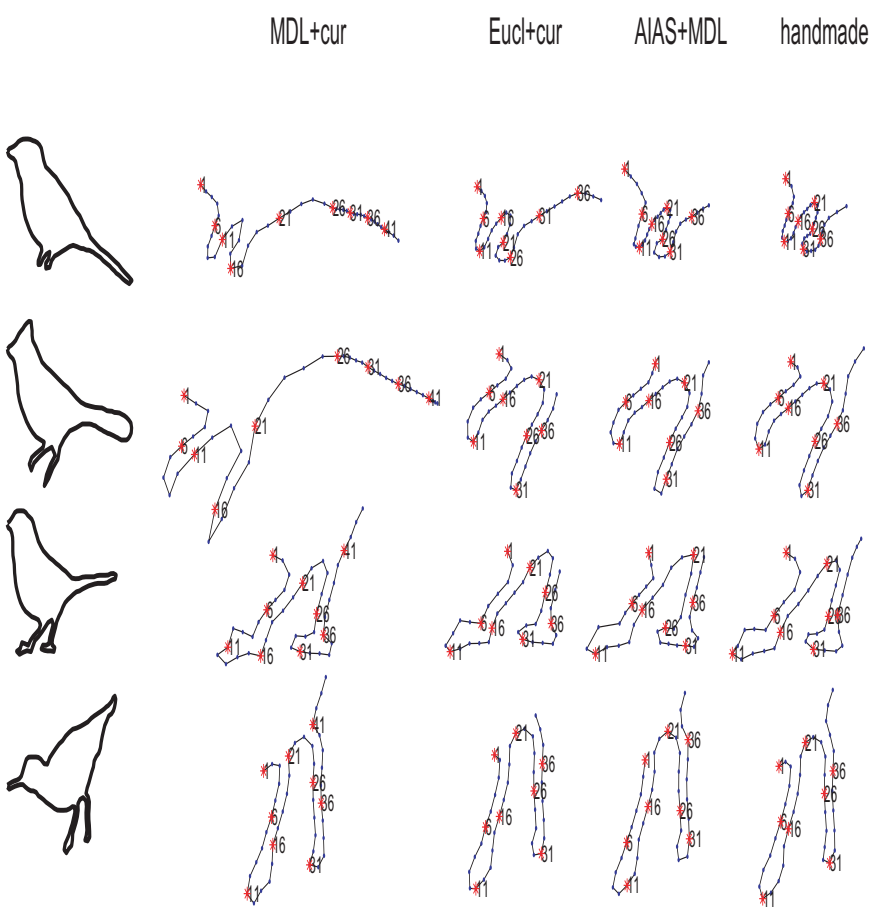


Figure 8.19: Correspondences on the bird dataset (the most difficult dataset) for four different algorithms zoomed in on the legs.

with these measures are discussed and it is shown that they do not succeed in measuring the desired properties (sometimes referred to as failing with respect to *validity*) and also they do not capture the quality of the correspondences.

Since the standard general model quality measures do not work, instead a Ground Truth Correspondence Measure (GCM) is proposed for the evaluation of correspondence quality to be used in benchmarking. It is shown in experiments on different datasets (both natural and synthetic) that this measure corresponds well to subjective evaluation

of correspondence quality, whereas the standard measures do not.

In this chapter several algorithms are bench-marked using GCM. In Table 8.1 it can be seen that MDL+Cur is the best algorithm. There is no algorithm that is best on all datasets and no algorithm gives as good correspondences as the correspondences manually marked. The semi-automatic algorithm is better than the automatic algorithms on all datasets but the flightbird dataset.

In previous work it is often claimed that automatic algorithms give better correspondences than models built by hand. These claims are often supported by measures like generality, specificity and compactness but we have shown that these measures have severe problems and should not be used for evaluation. Measuring GCM, it is concluded that models carefully built by hand are actually very good. In some cases it may not be reasonable to manually mark the full dataset but, as seen, a semi-automatic approach, where only a small number of shapes need to be manually marked, works very well.

Of course it would be desirable to have a measure that does not rely on having ground truth, but as we have shown here, the measures available all have severe problems and can not be used for benchmarking. It is our view that ground truth based measures must be used for evaluation of correspondence algorithms used for automatic model building.

The measure GCM can also be used to evaluate algorithms for finding correspondences on shapes of any dimension, e.g. surfaces. A simple examination is performed in [129] but a more thorough examination along the lines in this chapter remains to be done.

It would be very interesting to examine a ground truth segmentation measure using a database of segmented images and measuring how well the shape model is able to segment them. This could be done using for example the methods used in [96]. Also in for example medical decision support it would be interesting to evaluate a ground truth diagnose measure.

Chapter 9
Fully Automatic Construction Without Object Outlines

Everything is complicated; if that were not so, life and poetry and everything else would be a bore.

Wallace Stevens

One successful measure for establishing correspondences of high quality is minimum description length (MDL). In other approaches it has been shown that parts+geometry models which model the appearance of parts of the object and the geometric relation between the parts have been successful for automatic model building. In this chapter it is shown how to fuse the above approaches and use MDL to fully automatically build optimal parts+geometry models from unlabeled (i.e. different images in the training set can be of objects from different object classes) images. The algorithm handles object classification, outliers, missing data, model complexity selection and correspondence optimization, all as natural integrated parts of the process. The resulting MDL objective function is optimized over both discrete and continuous decision variables. The algorithm has been implemented and tested for establishing patch correspondences for model building from unlabeled images with occlusions.

9.1 Introduction

9.1.1 Background

MDL [7, 35, 104] has been applied both to establishing corresponding parameterizations of manually pre-segmented object outlines in order to build active shape models and in order to group-wise register the entire training images while building active appearance models [138]. In other approaches parts+geometry models have been successful for fully automatic model building ([10, 37, 47, 48]). In this chapter it is shown how to fuse the above approaches and use MDL to fully automatically build optimal parts+geometry models from unlabeled (i.e. different images in the training set can be of objects from different object classes) images with occlusions. Another line of research that is related to our work is the area of non-rigid factorization methods, e.g. [18]. In addition to reconstructing a model for the point set, they also try to estimate the camera motion. However, they assume that feature correspondences are given or obtained by some heuristics. Sim-

ilarly, outliers and missing data are handled with ad-hoc techniques. Our framework is well-suited to be applied to these problems as well.

In this chapter, MDL is used as a framework for a number of decisions, not only to determine correspondences. We include decisions about what object class the object in each image belongs to, what to include in the modeling and what to consider as outliers explicitly in calculating the minimum description length. The algorithm therefore classifies the images and decides what in the images to build the model from and what to consider as irrelevant background data. Decisions about classification (what object class the object in each image belongs to), model complexity selection (e.g. number of parts), what to include in the modeling and what to consider as outliers and missing data are done using an expanded version of MDL (a somewhat similar but more limited approach is used in [84]). In this way it is possible to build MDL optimal models fully automatically from unlabeled images in the presence of outliers and missing data. This chapter focuses mainly on showing that this process can be used to establish correspondences to be used in order to build models. The uses of the resulting models can be various and explorations of this is future work.

9.1.2 Setup

A set of images are normalized to have approximately the same size and image intensities are normalized by dividing each pixel be the mean value in a neighborhood of the pixel so that only local information remains and global illumination differences for example are removed (we will use a scale space framework as described in Section 9.4 so what is meant by local will depend on the scale space level). Next a histogram equalization is performed on the images to enhance contrast. From the set of images a number of parts are extracted by calculating corners/edge-points and sampling patches around these points on a number of scales. The mean of each patch is then subtracted. so that the intensities in each patch have mean zero.

This gives an unordered set of parts for each image and some of these parts are to be said to be in correspondence and used to build a model. For unordered sets, as opposed to what is usually assumed for example for curves or surfaces used to build shape models, it is often the case that points are frequently missing and that there are outliers. Traditional methods for shape analysis have problem with missing data and outliers. Here outliers are detected using MDL and instead we get missing data. Methods for Procrustes analysis and principal component analysis for missing data must then be used ([17, 70]).

9.1.3 Problem Formulation

To begin with, assume that all the images are of objects from the same object class. and we will describe how to build a model. After this, we will show how this process can be expanded to handle unlabeled images resulting in a set of models, one for each object class. So for now, assume that there is only one object class. The main problem is

formulated as follows. Assume that a dataset of examples $D = \{S_1, \ldots, S_{n_s}\}$ are given, where each example S_i is a set of unordered feature points $z_{i,j}$ (with surrounding patches) $S_i = \{z_{i,1}, \ldots, z_{i,k_i}\}$, from image i and each point $z_{i,j}$ lies in $\mathbb{R}^p$ for some dimension p, typically $p = 2$. An example of such sets is the output of a feature point detector, e.g. [64], with patches extracted around the points. Now there are several questions.

(i) How should one determine which points in each set is to be considered as outliers and clutter and which should be included in the model building?

(ii) How should one determine the correspondences of feature points across the examples?

(iii) What is a suitable shape/appearance (parts+geometry) variation model for the inliers?

The solution is useful for un-supervised learning of shape variation in image sequences or unordered image sets. The resulting models could then be useful for a number of applications, including *tracking*, *recognition*, *object pose* etc.

The idea here is to transfer ideas from shape variation estimation for curves and surfaces, [135, 36], where the correspondence problem is a crucial problem, to the concept investigated here. As opposed to the theory for curves and surfaces, however, we do not here have the problem of how to weight different parts of the curves relative to the other. On the other hand we will here allow outliers and missing data. We will use the minimum description length paradigm, [7], to select a suitable model.

We pose the problem as that of selecting: (i) inliers, (ii) correspondences and (iii) model complexity, so that MDL is minimized. As we shall see this becomes a mixed combinatorial and continuous optimization problem, where for each of a discrete set of possible inliers and correspondences, there is a continuous optimization problem which has to be solved. These continuous optimization problems involve both the problem of missing data Procrustes and missing data principal component analysis.

Assuming for the moment that the model contains N points with patches, an ordering of the parts can be represented as a matrix O (for order) of size $n_s \times N$. The entries in O are either 0 - representing that a model point is not visible (missing) in an image or an identity number between 1 and k_i signifying which image point is to be considered as a representative of the model point, i.e. $O_{i,j} = 0$ if model point j is not visible (missing) in image i or $O_{i,j} = k$ if model point j in image i is represented by $z_{i,k}$. Also introduce the set I consisting of pairs of indices (i, j) such that model point j is visible in image i, i.e.

$$I = \{(i, j) \,|\, O_{i,j} \neq 0\} \ . \tag{9.1}$$

The matrix O then settles both questions about outliers/inliers, missing data and correspondences. Outliers/clutter are all image points not represented in O. Missing data are

signified by zeros. Image points who's indices are in the same column of O are corresponding.

Given an ordering matrix O the data can be reordered, possibly with missing data into a structure T of N points in n_s images, i.e.

$$T_{i,j} = \begin{cases} z_{i,O_{i,j}} & \text{if } (i,j) \in I, \\ \text{undefined} & \text{if } (i,j) \notin I. \end{cases} \tag{9.2}$$

9.1.4 Modeling

For such a ordered set T with missing data one can align the shapes from the different images with a Procrustes analysis with respect to a transformation group $\mathcal{G}$ giving a mean shape m and a number of transformations $\{g_i\}_{i=1\ldots n_s}$ with $g_i \in \mathcal{G}$ such that $g_i(m) \approx T_i$ or rather $g_i^{-1}(T_i) \approx m$. More specifically, typically the sum $\sum_{i=1}^{n_s} ||g_i^{-1}(T_i) - m||$ of the norms of the differences between the aligned shapes $g_i^{-1}(T_i)$ and the mean shape m is minimized. The norm used must be able to handle missing data. After this a Principal Component Analysis handling missing data can be performed on the residuals between $g_i^{-1}(T_i)$ and m. From this, d shape variational modes, denoted $\{v_l\}_{l=1\ldots d}$, and mode-parameters $\{\lambda_{li}\}_{l=1\ldots d, i=1\ldots n_s}$ can be determined. New shapes can then be synthesized as $g(m + \sum_{l=1}^{d} \lambda_l v_l)$, where λ_l are scalar coordinates and $g \in \mathcal{G}$.

This takes care of the geometry variation, i.e. the shape variation as defined by the corresponding points. Next, the appearance of the parts as defined by the corresponding patches is dealt with in a similar fashion. For each N model points all the patches belonging to that model point are also analyzed with a PCA. Note that for these PCA's there is no missing data but the number of patches differ if some points are missing. These PCA's then result in mean patches $\{m_i^p\}_{i=1\ldots N}$, numbers of modes used $\{d_i^p\}_{i=1\ldots N}$, patch-modes $\{v_{ij}^p\}_{i=1\ldots N, j=1\ldots d_i^p}$ and patch-mode-parameters $\{\lambda_{ijk}^p\}_{i=1\ldots N, j=1\ldots d_i^p, k=1\ldots n_s}$.

We need to assess a number of different choices: the number of model points N, the ordering O, the mean shape m, the transformation group $\mathcal{G}$, the transformations $\{g_i\}$, the number of shape variation modes d, the shape variation modes $\{v_l\}$, the shape coordinates $\{\lambda_{li}\}$, mean patches $\{m_i^p\}$, numbers of patch-modes $\{d_i^p\}$, patch-modes $\{v_{ij}^p\}$ and the patch-mode-parameters $\{\lambda_{ijk}^p\}$. The idea here is that a common framework such as the minimum description length framework could be used to determine all of these choices. This would put the whole chain of difficult modeling choices on an equal footing and would make it possible to use simple heuristics for making reasonable choices, while at the same time have a common criterion for evaluating different alternatives.

The whole process can thus be seen as an optimization problem

$$\min_{\mathcal{M}} \mathrm{dl}(D, \mathcal{M}), \tag{9.3}$$

over the unknowns

$$\mathcal{M} = (O, m, \{g_i\}, d, \{v_l\}, \{\lambda_{li}\}, \{m_i^p\}, \{d_i^p\}, \{v_{ij}^p\}, \{\lambda_{ijk}^p\})$$

given data D.

In principle, all these variables could now be optimized simultaneously using MDL. In practice however, this becomes impossibly cumbersome. Also for some of the parameters to be determined, more easy methods not using MDL will give very close to the same result. This includes for example the Procrustes analysis and the PCA's. Therefore these parameters are determined without MDL and we focus the use of MDL where it really makes a difference, namely in determining number of model points, inliers/outliers/missing data, model complexity (e.g. number of parts and numbers of modes) and for establishing correspondences.

9.2 Calculating MDL

9.2.1 Introduction

A number of sets of points with surrounding patches are given. In order to determine a model that explains points that can be seen in many of the images, the goal is to minimize the minimum description length that is needed to transmit all the interesting points of all images, in hope that a model will be able to make a cheaper description than simply sending the data bit by bit. Here we will present the minimum description length for the data and the model. For the outliers, one must simply send the information bit by bit. For the points and patches that are included as inliers in the model, the idea is that it is cheaper to send the model with parameters and residuals etc. to explain the data. For the modeled points and patches one must send: the model, the model parameters, information if a certain point is missing, the transformation and the residuals.

Figure 9.1 shows a simple illustration of outliers/inliers and missing data. In the figure a shape model is explaining most of the data. The points marked with 'x' are in the model and visible in that particular image. A point marked with 'o' is in the model but missing in the current image. The points marked with '+' are outliers.

9.2.2 Coding Data with Uniform Distribution

Assume α is uniformly distributed and quantified to

$$\alpha_{min} \leq \hat{\alpha} \leq \alpha_{max}, \;\; \hat{\alpha} = m\delta, \;\; m \in \mathbb{Z} \;\; . \tag{9.4}$$

Then $\hat{\alpha}$ can take

$$\frac{\alpha_{max} - \alpha_{min}}{\delta} \tag{9.5}$$

different values. Since uniform distribution is assumed, the probability for a certain value of $\hat{\alpha}$ is

$$\mathcal{P}(\hat{\alpha}) = \frac{\delta}{\alpha_{max} - \alpha_{min}} \;\; . \tag{9.6}$$

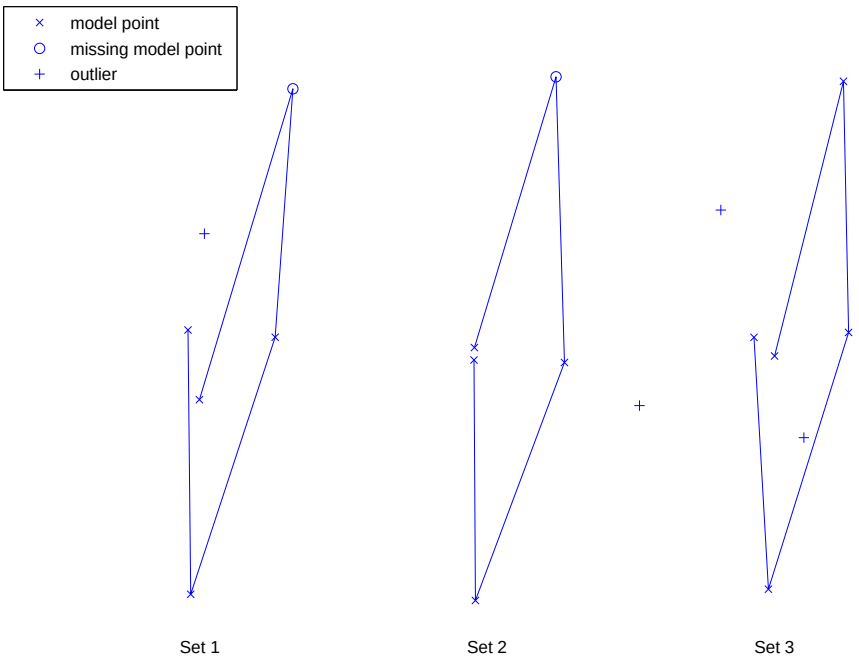


Figure 9.1: A schematic figure of three sets of points. Each set consists of a number of interesting points. These points are either considered to be outliers or modeled by a statistical shape model.

Shannon's codeword length for $\hat{\alpha}$, is

$$-log(\mathcal{P}(\hat{\alpha})) = -log(\frac{\delta}{\alpha_{max} - \alpha_{min}}) \ . \tag{9.7}$$

If the parameters α_{min}, α_{max} and δ are unknown to the receiver, these need to be coded as well.

9.2.3 Coding Data with Assumed Gaussian Distribution

The derivation for the description length $\mathcal{L}_{guassian}$ of a one-dimensional dataset from a Gaussian 1-parameter distribution described via a PCA giving a number of normally distributed 1 dimensional variables for each object was done in [32] and later modified by different authors. The version used here is described in chapter 5 and more details can be found for example in [41].

9.2.4 The Total Minimum Description Length

Modeling Unknowns

For unknown points and patches we assume uniform distribution. The minimum description length for a point $\hat{x}$ uniformly distributed over the image is

$$dl_{rect} = -log(\mathcal{P}(\hat{x})) = -2log(\frac{dx}{X}) \ , \tag{9.8}$$

(rect being short for the rectangular shape of the distribution). Here X is the possible coordinate range, and dx is the resolution. The factor 2 comes from that an image point is two-dimensional. Similarly, the description length for a general patch $\hat{p}$ with uniformly distributed intensities is

$$dl^{p}_{rect} = -log(\mathcal{P}(\hat{p})) = -n_{patch}log(\frac{dy}{Y}) \ . \tag{9.9}$$

Here n_{patch} is the number of pixels in the patch, Y is the possible intensity range, and dy is the resolution.

Outliers

Corner-points not selected to represent a model point are considered as outlier points. The outlier points are assumed to be uniformly distributed over the image, so with n_o^{points} number of outlier points, MDL for the set of them is

$$dl^{points}_{outliers} = n_o^{points} \cdot dl_{rect} \ . \tag{9.10}$$

Image pixels that are not covered by an inlier patch representing a non-missing part are considered as outliers and coded with an outlier id (from 1 to $idmax$) and an intensity. The intensities of outlier pixels are assumed to be uniformly distributed so with n_o^{pixels} number of outlier pixels, MDL for the set of them is

$$dl^{pixels}_{outliers} = n_o^{pixels}(log(\frac{dy}{Y}) + log(idmax)) \ . \tag{9.11}$$

The total outlier cost then becomes

$$dl_{outliers} = dl^{points}_{outliers} + dl^{pixels}_{outliers} \ . \tag{9.12}$$

Missing Data

For each model point we need to know if the point is missing in an image or not. This means one bit for each n_p model points in all n_s images. Conversion to nats gives

$$dl_{index} = log(2)n_s n_p \ , \tag{9.13}$$

where n_p is the number of landmarks in the model and n_s is the number of images.

Transformations

For each image the transformation g of the model has to be encoded. Translations are assumed equally distributed within the size of the image and other transformation parameters are coded with equal accuracy. This gives the following expression for the MDL for the set of transformations:

$$dl_{trans} = n_s n_{dof} dl_{rect}/2 \ , \tag{9.14}$$

where n_s is the number of images and n_{dof} is the degrees of freedom in the transformation group.

The Model

The coordinates of the mean-shape and the coordinates of the shape-modes are also assumed uniformly distributed within the size of the image. Thus the cost is

$$dl_{meanshape} = n_p dl_{rect} \ , \tag{9.15}$$

$$dl_{shapemodes} = n_p n_m dl_{rect} \ , \tag{9.16}$$

where n_m is the number of shape modes used by the model. Similarly, for each model point i, the cost for the mean-patch and the patch-modes is

$$dl_{meanpatch} = dl^p_{rect} \ , \tag{9.17}$$

$$dl^i_{patchmodes} = n^i_m dl^p_{rect} \ , \tag{9.18}$$

where n^i_m is the number of patch appearance modes used by the model for model-point i. We put these together as

$$\begin{aligned} dl_{model} = {} & dl_{meanshape} + dl_{shapemodes} + \\ & n_p dl_{meanpatch} + \sum_{i=1}^{n_p} dl^i_{patchmodes} \ . \end{aligned} \tag{9.19}$$

Gaussian Parameters and Residuals

Parameters for shape modes and patch appearance modes as well as residuals for point coordinates and patch intensities are assumed Gaussian. Different possible range och different accuracy for point coordinates and for patch intensities gives two different costs for Gaussians, $\mathcal{L}^{point}_{guassian}$ and $\mathcal{L}^{patch}_{guassian}$. In both cases, MDL is a function of the standard deviation s and the number of data-points n. The total cost for these becomes

$$dl_{coordinates} = \sum_{k=1}^{n_m} \mathcal{L}^{point}_{guassian}(s_k, n_s) + \sum_{i=1}^{n_p} \sum_{k=1}^{n^i_m} \mathcal{L}^{patch}_{guassian}(s_{ik}, n^i_s) \ ,$$

$$dl_{residuals} = \mathcal{L}^{point}_{guassian}(s, 2\sum_{i=1}^{n_p} n_s^i) + \sum_{i=1}^{n_p} \mathcal{L}^{patch}_{guassian}(s_i, n_s^i n_{patch}) \ ,$$

$$dl_G = dl_{coordinates} + dl_{residuals} \ , \tag{9.20}$$

where n_s^i is the number of images that model point i is not missing in.

Total MDL

So the full cost for sending the data is

$$MDL_{tot} = dl_{trans} + dl_{index} + dl_G + dl_{model} + dl_{outliers} \ . \tag{9.21}$$

where we note that dl_{trans} actually does not depend on any decisions once the transformation group is decided, so it may not need to be included.

9.2.5 Note on Total MDL

Often when using MDL to establish correspondence, the formula for the total MDL is simplified further by always using all the available modes. Since we want to simultaneously make decisions about model complexity and about correspondence that is not done here. The gain in reduced residuals is weighed against the cost of more modes.

9.3 Optimizing MDL

9.3.1 Evaluating the Objective Function

Given a current set of parameters $\mathcal{M}$ as in Equation (9.3), a model of global shape and part appearance can be built and MDL for the model and for the data described using the model can be calculated. For each suggested model one needs to calculate the minimum description length for sending all the outliers and all the data modeled with that particular model. The number of shape modes can vary between zero and $n_s - 1$ and all these models must be evaluated in order to pick the best one. Note here that since the modes calculated when using missing data PCA depends on the number of modes used, the model needs to be calculated over and over again as the different numbers of modes are tested. In the optimization procedure the tested model with least MDL is then compared to previous best solution.

9.3.2 Optimization Strategy

The whole optimization process over all unknowns to find

$$\min_{\mathcal{M}} \mathrm{dl}(D, \mathcal{M}) \ , \tag{9.22}$$

where D is the data, can be divided into three parts: (1.) Optimization over the discrete ordering matrix O, (2.) optimization over the transformations $\{g_i\}$ and (3.) optimization over the remaining parameters

$$\tilde{\mathcal{M}} = (m, d, \{v_l\}, \{\lambda_{li}\}, \{m_i^p\}, \{d_i^p\}, \{v_{ij}^p\}, \{\lambda_{ijk}^p\}) \ . \tag{9.23}$$

It is possible to include the choice of $\mathcal{G}$ as an unknown parameter and determine it as well using MDL. You may simply try some different groups and see which one gives the lowest MDL. This would mean weighing the higher cost of coding transformations with more degrees of freedom against the gain of smaller residuals left to encode using the model. However, it is just as likely that you want to choose $\mathcal{G}$ based on the application, before starting the optimization. In the experiments in this thesis, $\mathcal{G}$ was chosen to be the group of similarity transformations.

Assume that an ordering O is given. Then it is straightforward to reorder the inlier points into the data structure T as described above. Each ordering O determines the number of model points, the n_{inlier} inliers, the $n_{outlier}$ outliers and the correspondences. The minimum description length for the outliers is independent of $\{g_i\}$ and $\tilde{\mathcal{M}}$.

Alignment Transformations

Given O, $\{g_i\}$ is to be determined using some version of Procrustes analysis that can handle missing data. Ideally, $\{g_i\}$ should be evaluated using MDL. However, this considerably increases time complexity and the results of alignments based on Euclidean distance is normally very close to the same as alignments based on MDL (although sometimes the difference can be significant) , cf. [42]. Therefore MDL is not used to determine $\{g_i\}$. Instead the alignments are based on minimizing Euclidean distances. If there are no missing data this can be done optimally without iteration, but this requires a singular value decomposition which with missing data is itself an iterative procedure. So instead this is done in a number of steps where the process can be terminated without doing all the steps if precision is less important than speed. By a shape will here be meant the set of points from one image corresponding to the points in the model. First all shapes are translated to the origin and scaled so that the mean distance of points in the shape from the origin is 1. Either this is simply done using all inliers, or the quality can usually be improved by only using inliers that are present in most of the images. Calculate the number of images that each model point is nonmissing in. We want to use model points with high results. Pick the cutoff such that all images use at least the minimum number of points needed to determine the transformation. Next all shapes are aligned to the first shape $shape_1$. If the shapes are represented as complex column vectors this alignment is achieved by multiplying with

$$shape_i^* shape_1 / (shape_i^* shape_i) \ ,$$

as described in for example [38]. When calculating the alignment parameters, if there are missing points in one of the shapes, those points are excluded from both shapes so that the alignment is done based on the non-missing points. Next follows a loop of estimating a mean-shape and aligning all shapes to it, estimating a new mean-shape etc. Finally the error remaining is minimized to find

$$\min_{m,\{g_i\}} \sum_{(i,j)\in I} |T_{i,j} - g_i(m_j)|^2 \tag{9.24}$$

using Gauss-Newton while keeping the size of the mean-shape fixed to avoid shrinking everything.

PCA

Assume next also that the number of shape modes d and the numbers of part appearance modes $\{d_i^p\}$ are given. What remains to determine are the modes and the parameters for the data in those modes. As for the alignments this should ideally be done using MDL but again more direct approaches will give very close to the same result at less cost. For $\{v_{ij}^p\}$ and $\{\lambda_{ijk}^p\}$ there are no problems with missing data since if a point is missing that only means that the number of examples of that part is smaller. So $\{v_{ij}^p\}$ and $\{\lambda_{ijk}^p\}$ are simply determined by the standard procedure of doing a singular value decomposition. For the shape modes the missing data is a problem and we must use a PCA that can handle missing data. What is needed then is a principal component analysis (via a singular value decomposition $X = USV^T$) of the residuals X after subtracting the mean-shape from all the aligned shapes and this PCA must handle missing data. There exist different approaches for this but since if we could afford very time consuming approaches we would use MDL anyway, here a relatively simple (but not too simple) approach is used. Since the typical value of the residuals is zero, first all missing entries are set to zero. Then a standard PCA is performed via $X = USV^T$. Next the missing entries in X are updated with the corresponding values of the rank d approximation $U_d S_d V_d^T$ of X. Here U_d is the first d columns of U, S_d is top left $d \times d$ part of S and V_d is the first d columns of V. Then a new PCA is done and the missing entries are updated etc. The number of loops are chosen as deemed fit.

Resulting Approach

To sum up, given O, $(m, \{m_i^p\}, \{g_i\})$ are determined as described above. Then each valid value of the number of nodes d and $\{d_i^p\}$ are examined. For each value of d, missing value PCA is performed and then the part of the total MDL depending on d is calculated in order to pick the d that gives the lowest MDL. For the patch appearance models only one PCA per part is needed, not one for every value of d_i^p. For each value, the part of the total description length depending on d_i^p is calculated in order to pick the d_i^p that gives

the lowest MDL. Thus, the minimal description length can be seen as a function of the ordering O.

9.3.3 Optimization of O

Optimizing minimum description length with respect to O is a combinatorial optimization problem. We suggest the following algorithm that (1) finds a reasonable initial guess and (2) searches for a local minima in a combinatorial optimization sense by attempting the following six types of changes and evaluating using MDL. The optimization terminates when none of the operations below give an improvement in minimum description length.

1. Addition of a model part. In each image either the new point is set as missing or as corresponding to one of the points from the image.

2. Deletion of a model part. In each image the points corresponding to this model part are set as outliers.

3. Change of a point from outlier to inlier belonging to one of the model parts.

4. Change of a point from inlier to outlier.

5. Changing correspondence by moving an inlier to a neighboring point, i.e. setting the old point as an outlier and changing the new one from being an outlier to representing the same model point that the old point represented.

6. Using the current model to restart the process for the points from one image by fitting the model to the points, thereby defining a row of O. The alignment of the model to the image can either be tried in a few different versions or it can be determined in RANSAC fashion by using minimal sets of current inliers. The matching is then done as for the initial guess described below.

The final algorithm for determining minimal description length is then

I Make an initial guess on point ordering O.

II Calculate optimal MDL for that ordering.

III See if any of the perturbations above lowers MDL.

IV
- If it does, make those changes and continue with step III.
- If not, then we are at a local minima, stop.

9.3.4 Initial Guess

The initial guess is made by using the (for example 5 or a random number) strongest corner points (for example Harris-corners) and their surrounding patches in one image as a model m_0. For each of the other images $\{image_i\}$ this model m_0 is fitted to the set of available parts from the image. Matching is done as follows. Given n points in the model and m points extracted in $image_i$, form an $n+1 \times m+1$ cost matrix C whose top left $n \times m$ elements are $C_{ij} = d_E + w \cdot d_P$, where d_E is the Euclidean distance between model point i and feature point j, d_P is the Euclidean difference in patch appearance (between the mean-patch from the model and the patch from the image) and w is a weight parameter which is needed since point coordinates and image intensities have different ranges. The last row $C_{n+1,j}$ is set to a constant representing the cost of not associating a feature point j with a model point. Similarly the last column $C_{i,m+1}$ is set to a constant representing the cost of not associating a model point j to any of the feature points. These positive constants can be chosen large or small depending on whether we prefer to get a strong matches or many matches. In this chapter we set both these constants to the same high value, thus enforcing many matches. The matching is then done by solving the transport problem with supply of $s = [1, \ldots, 1, m]$ and demand $t = [1, \ldots, 1, n]$. Here we used standard algorithms for solving the transport problem [92]. A number of images from the training set can be tested for giving initial models m_0 in this way and MDL can be used to pick the best initial guess.

9.4 Scale Space

The process is performed in a scale space framework, and the resulting model also has patch-information from several scales.

The original images are smoothed and down-sampled to create scale space representatives. Next, the intensities are normalized as described in section 9.1.2. The local neighborhood used for this covers larger parts of the image at coarser scale space levels, so that at coarse levels global structures are stressed and at finer levels local structures are stressed. Also the coordinates of the extracted feature points (extracted at the original level) are scaled to fit the down-sampled images and also rounded off to integers. In this way both gray level intensities (parts) and point coordinates (geometry) are represented with different detail at different scales. Next 7×7 patches are sampled around each point so that the same size patch (as measured in pixels) covers different relative amounts of the images at different scale.

In usual fashion the optimization is first run till convergence at the coarsest level and then the result is used as an initial estimate for the next level. As we move to the next level, point coordinates are simply scaled to the appropriate range. For the patches however the patches from the new level do not simply replace the patches from the old level, but instead the coarse patches are kept so that the new patch represents the local

information at greater detail as well as the wider appearance information with less detail. Since the patch description becomes richer, this of course means that the description length increases, so before starting the optimization at the new level, a new best MDL is calculated using the richer descriptions. For the point coordinates there is no such effect, since before calculating MDL, the shapes are temporarily scaled so that they have mean distance 1 from the points to the center of the shape.

The resulting models then describe the geometry at the detail of the finest level and the parts at several levels of locality and detail.

9.5 Unlabeled Images

Assume now that not all the images are of objects from the same object class, but instead there are images of objects from different object classes mixed together and we want to automatically get a model for each object class.

Introduce a label for each image, denoting which object class it is currently assumed to belong to. Given such labels for all the images we get a set of problems as described in the previous sections, one for each object class. The total MDL is simply the sum of the minimum description lengths for the individual object classes.

9.5.1 Initial Guess

Instead of just picking *one* random image to give the initial model, as described in section 9.3.4 above, we can pick $n_{classes}$ random images to give $n_{classes}$ initial models. For each image, every initial model is tried and the initial label is set corresponding to the model giving the smallest cost in the transport problem solution. As above, we can try a number of sets of initial models and use MDL to pick the best initial guess.

9.5.2 Optimization

After the initial guess, the optimization terminates when none of the available operations give an improvement in MDL. The available operations are the six described in section 9.3.3 above and also

7. Change the object class label for an image and fit the model for the new class to the image.

We can now use MDL to also get an automatic classification of the images as a natural part of the process.

9.5.3 Unknown Number of Classes

Finally, we can also let the number of classes be unknown. For the initial guess then pick $n_{classes}$ at random for each attempt and for the optimization add the perturbations:

8. Add an object class and assign images to it.

9. Remove an object class and assign the images to other classes.

9.6 Experimental Validation

First we test the algorithm on images from only one object class.

9.6.1 Points Only

In the first experiment we only work with the points them self, i.e. not with the patches. We took a digital film recording of a persons face as it moves in the scene. A sequence of 100 frames was captured and a standard interest point detector [64], was run on all of the frames. In each frame a face detector was run and those interest points that were within the rectangular frame of a face detector [45], was kept.

This gave roughly 880 feature points (between 5 and 13 points in each frame). Three such frames are shown in Figure 9.2 together with the extracted feature point shown as small rectangular points.

The initial guess ordering resulted in 584 of the 880 feature points being associated with any of the 9 model points.

After local optimization to lower MDL we got a model with 12 model points. Here 740 of the 880 points were associated to a model point. In Figure 9.3 is shown three frames out of the 100 overlayed with feature points and best fit of the 12 model points obtained after optimizing MDL. Notice that certain points in Figure 9.2 are classified as outliers and are not shown in Figure 9.3.

9.6.2 One Object Class with Patches

After these initial experiments we move on to using the patches as well as the points. The training set consists of head shots of different persons and no face detector is used. Feature points and surrounding patches are extracted and the optimization is performed as described in the previous sections. As can be seen in Figure 9.4 relevant model points are selected and correspondences are established.

9.6.3 Occlusions

Next the algorithm was tested on images with synthetic occlusions. For every image in the training set there was a 50% chance of putting in a random occlusion. The occlusions were rectangles with random proportions, size, location and gray level intensity. As can be seen in Figure 9.5 the algorithm handles the occlusions well. Occluded points are set as missing and the correspondences of remaining points are not destroyed by the occlusions.

Figure 9.2: Three out of 100 frames used for testing. Detected feature points are shown as white squares

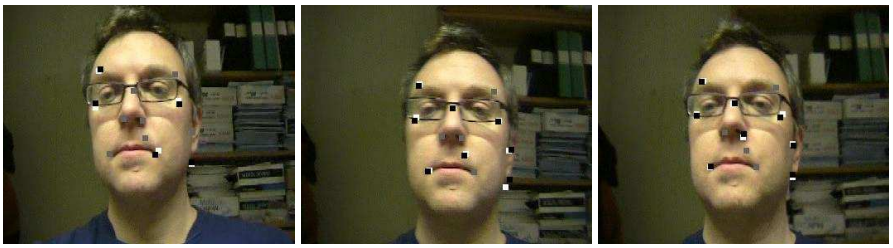


Figure 9.3: Three out of 100 frames used for testing. For measured points (in white) the fitted model points are shown in black. For missing data the fitted model points are shown in gray.

9.6.4 Unlabeled Images

Next we test the algorithm on unlabeled images. We use the head shots from the above experiments together with some images of star fish from the Caltech101 database [46]. The images are mixed together in random order before they are fed to the algorithm. The result is that all images are classified correctly and we get two separate sets of correspondences and models as illustrated in Figure 9.6.

Then we also test the algorithm on unlabeled images with occlusions. The images are as in the previous experiment but for every image there was a 40% chance of putting in a random occlusion. All images are still classified correctly and the occlusions are handled well, see Figure 9.7.

We now also let the number of classes be unknown, first without occlusions. The algorithm correctly selects two classes and works as before, see Figure 9.8. Note also that the algorithm automatically selects to model the face but not the shirt in the head shot images. Figure 9.9 is an illustration of the parts of the images covered by the patches at different scale space levels. Remember that the final model contains patches from all the scale space levels.

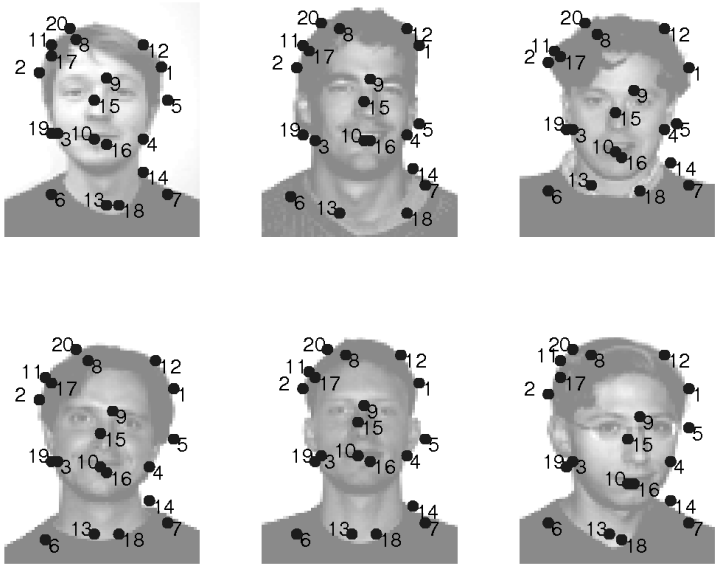


Figure 9.4: The selected model points on a subset of the images.

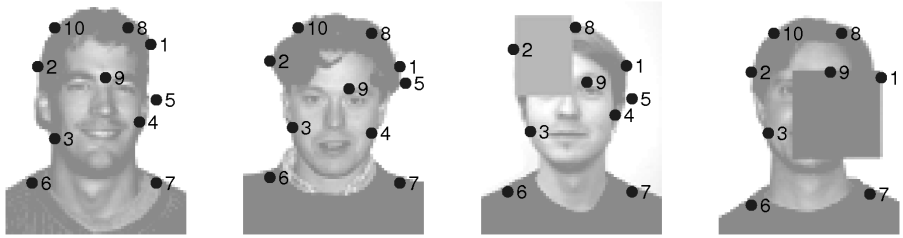


Figure 9.5: The selected model points on a subset of the images in the presence of occlusions in the training images.

Finally, unknown number of classes also works with occlusions, see Figure 9.10.

9.7 Conclusions

In this chapter we have solved the problem of automatic construction (i.e. no manual annotation of training data) of parts+geometry models (giving robustness as compared to modeling the entire image) from unlabeled images (i.e. mixed object classes) with

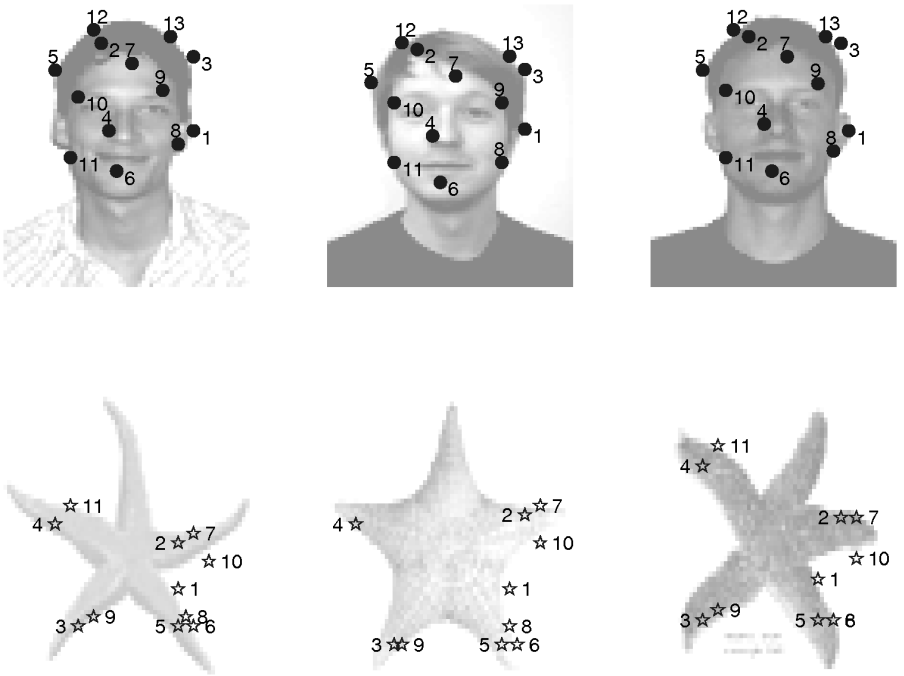


Figure 9.6: The selected model points on a subset of the images. Stars/dots denote that the image is classified as belonging to class 1 / class 2.

occlusions, using MDL (proven to be successful at establishing correspondence for model building).

The result is an algorithm that given a set of unlabeled images of objects from different object classes extracts points and patches as different scales and from this determines among other things (i) the number of object classes, (ii) clustering/classification of images according to object class, (iii) the number of model parts for each class, (iv) which image points are outliers/clutter and which are inliers, (v) correspondences, (vi) the mean shape and shape variational modes of the models (geometry) and (vii) for each model part the mean patch and patch appearance variational modes.

The resulting models can be used for many different applications including for example recognition, pose determination, tracking and shape/appearance analysis.

An extension to three dimensional images, common for example in medical image analysis, should be fairly straight forward. The only reason it has not been explored in this thesis is lack of time. Interest points can of course be detected similarly in 3D, and the patches can be replaced with boxes around the points. Alignments can still be done

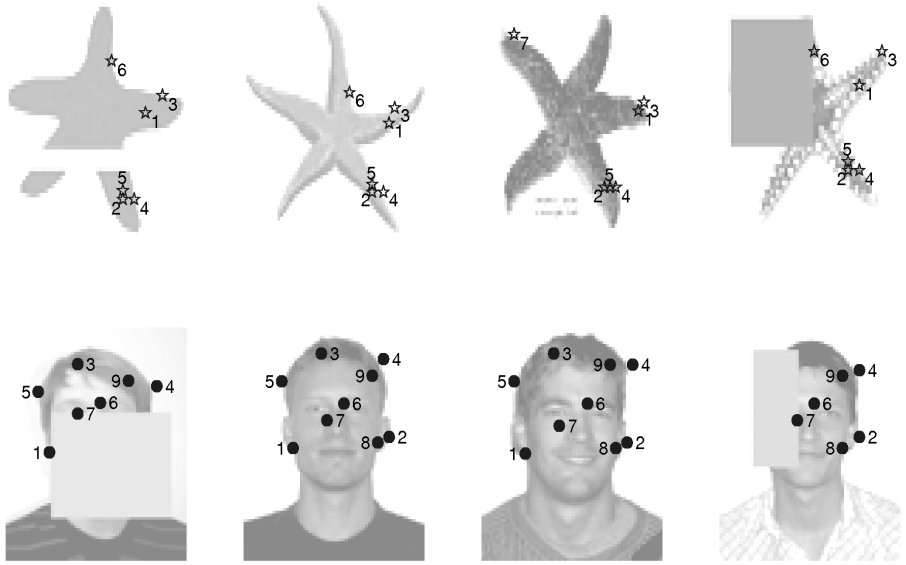


Figure 9.7: The selected model points on a subset of the images. Stars/dots denote that the image is classified as belonging to class 1 / class 2.

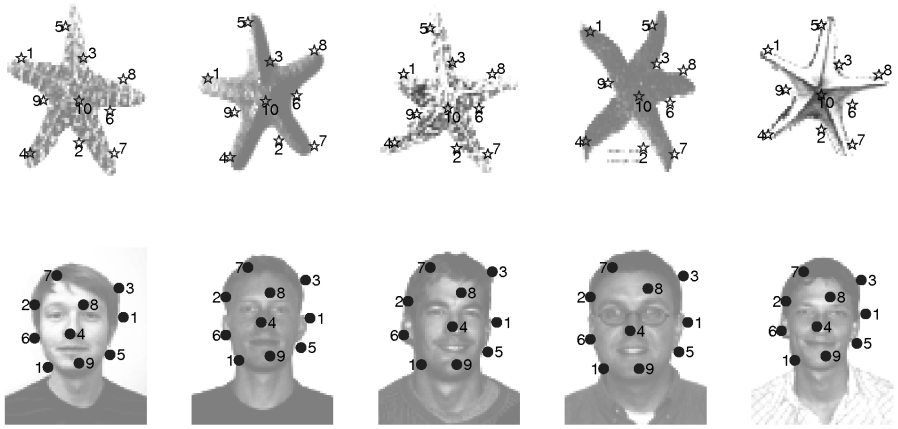


Figure 9.8: Unknown number of classes. The selected model points on a subset of the images. Stars/dots denote that the image is classified as belonging to class 1 / class 2.

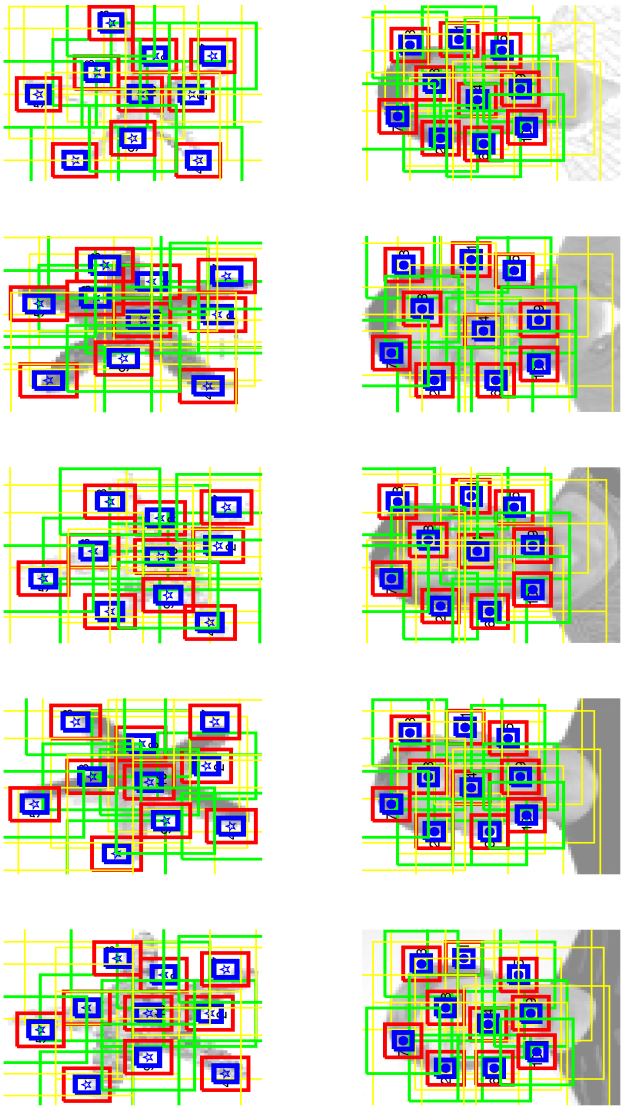


Figure 9.9: The same result as in Figure 9.8. Here is an illustration of the parts of the images covered by the patches at the different scale space levels all included in the final model.

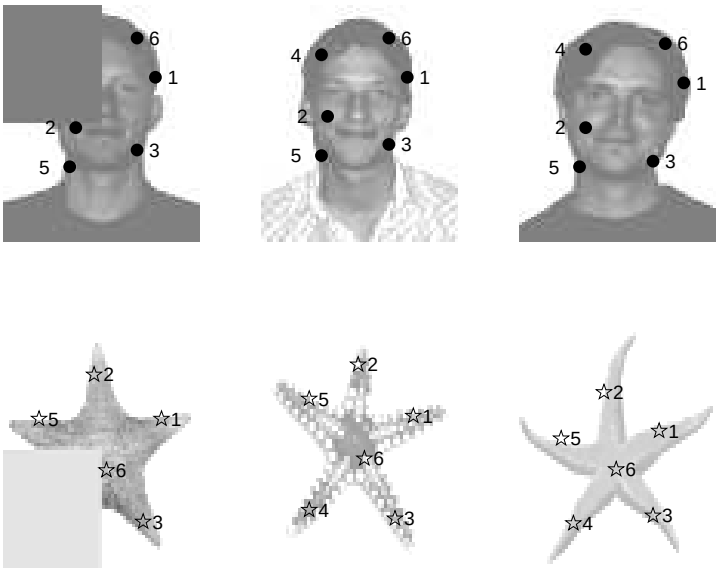


Figure 9.10: Unknown number of classes. Stars/dots denote that the image is classified as belonging to class 1 / class 2.

using translations (with three parameters instead of two), scale and rotations (with three parameters instead of one). The rest should be the same as with two dimensional images. Note that time and memory complexity need not increase significantly in some major aspects, since the number of model points/boxes in 3D may not need to be significantly larger than the number of model points/patches in 2D. However, in one aspect complexity will increase significantly. Using $7 \times 7 \times 7$ boxes instead of 7×7 patches will have an impact on memory consumption as well as time.

Chapter 10
Conclusions

Shape models and the automatic building of such models have proven over the last decades to be powerful tools in image segmentation and analysis. This thesis makes contributions to this field.

The segmentation algorithm typically uses an objective function summing up contributions from each sample point. In Chapter 4 this is replaced by the approximation of a surface integral which improves the segmentation results.

Before building a model the shapes in the training set have to be aligned. This is normally done using Procrustes analysis. In Chapter 6, an alignment method based on MDL is examined and MDL is optimized using the gradient which is derived in that chapter. This alignment is shown to be superior to standard Procrustes alignment, sometimes a little, sometimes a lot.

When trying to build optimal models by optimizing the parameterizations of the shapes in the training set to get a low MDL there is a tendency for the parameterizations to put most of their weight on small parts of the shapes by doing a mutual reparameterization since this gives a low MDL, but this is because of a flaw in the objective function and these parameterizations do not result in a good model. This has formerly been addressed using ad hoc methods. In Chapter 7, this problem is solved by replacing the standard scalar product with a formula that is invariant to mutual reparameterizations and this is shown to result in better models.

To evaluate the quality of shape models the standard measures have been generality, specificity and compactness as defined in [31]. In Chapter 8, these measures are shown to have severe weaknesses. An alternative measure called Ground Truth Correspondence Measure is presented and this measure is shown to perform better. This measure is then used to benchmark a number of algorithms for automatic model building.

Typically, shape modeling, both earlier work and the work in this thesis, assumes that the training set consists of images where the shape has been segmented as a curve or a surface as a preprocessing step. There are methods for building models without this information by modeling the whole images after registration, but sometimes there are other things in the image than the object to be modeled and sometimes not the whole object is visible. In object recognition, techniques for dealing with non-preprocessed data are available, but these give weaker models that are less useful for segmentation and analysis. In Chapter 9, a method is introduced that does not need preprocessed manually segmented data, automatically handles outliers/background and missing data, and still produces strong models. The training set is here just a set of images. The algorithm picks

a set of feature points with patches around them in each image and makes all the decisions about what to include in the model and what to consider as background and about what points in the different images are to be considered to be corresponding. This results in patch-based shape and appearance models generated fully automatically.

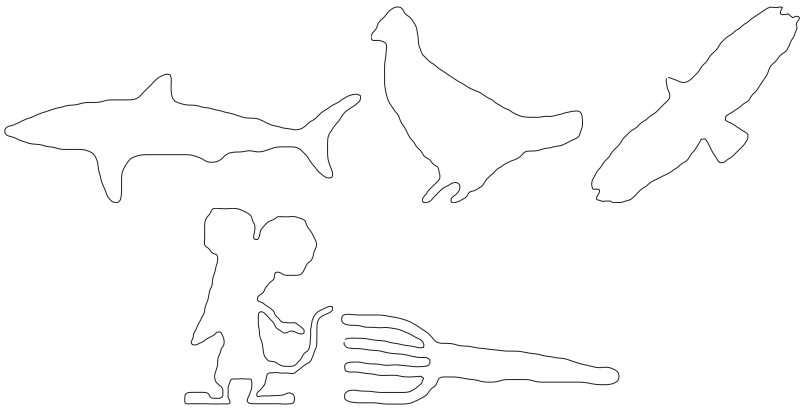
I believe that the future of this field is in this fully automatic generation of shape and appearance models. As this thesis shows, it is possible to build strong generative models, with interpretable parameters, fully automatically from unlabeled images with occlusions. However, it is not proven (and not likely) that this first attempt in this fashion is the best procedure for doing this. More research is needed about alternative fully automatic approaches.

Appendix A

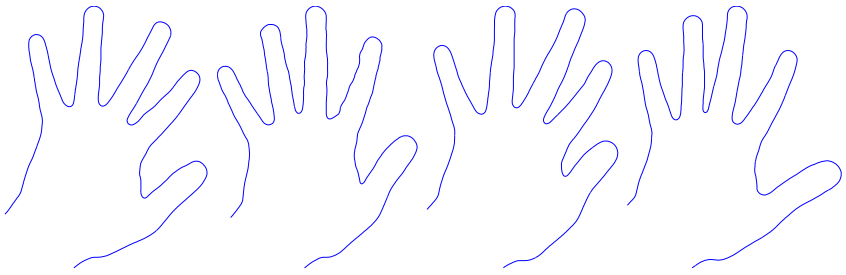
Datasets

Here the data sets used for the experiments in the thesis are briefly presented.

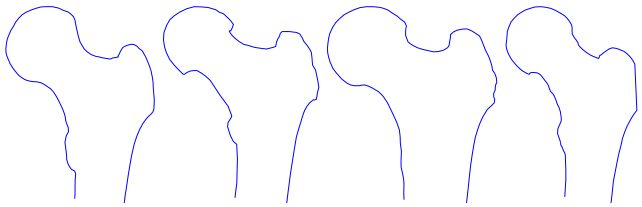
LEMS-database A number of shape classes used in this thesis (sharks, birds, flight birds, rats, forks) are curves generated from images in the LEMS database from Brown University [113, 114, 117]. Each of these shape classes consists of 13-23 shapes.



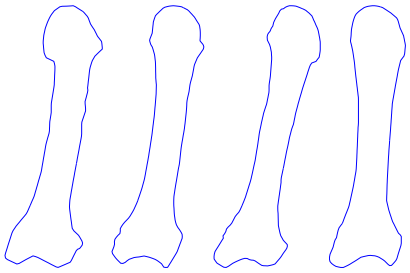
Hands 23 contours of a hand segmented semi-automatically from a video stream. To simplify the segmentation the hand was filmed against a dark background. The hands were segmented by manually placing some points and between those points Dijkstra's algorithm, [2], was used to find the most suitable edge. These shapes consist of 1522-1747 landmarks defining the curve.



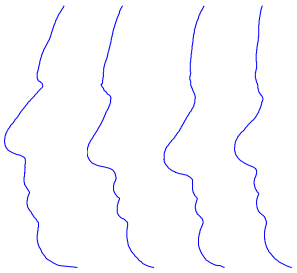
Femurs 32 contours of femurs in the hip taken from X-rays. They consists of 89-217 landmarks defining the curve.



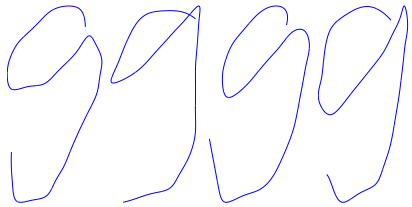
Metacarpal 24 contours of metacarpals (a bone in the hand) extracted from standard projection radiographs of the hand in the posterior-anterior projection. They consists of 281 landmarks defining the curve.



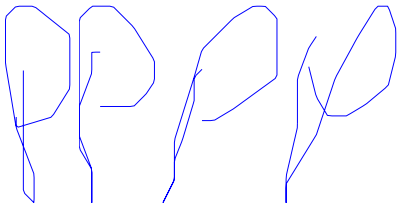
Silhouettes The silhouette data set consists of 22 contours of silhouettes of faces. 22 persons were photographed using a digital camera. The silhouettes were then extracted using an edge detector. They consists of 448-801 landmarks defining the curve.



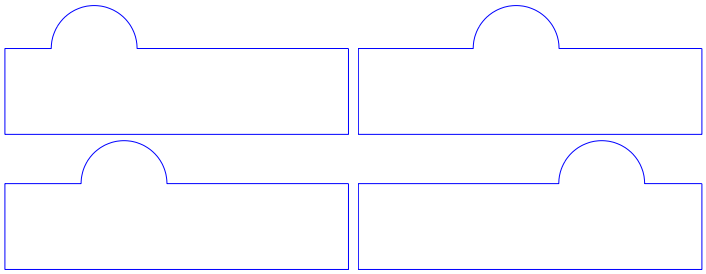
The letter g A data set of 17 curves of the letter g. The curves of the letter g were produced using a device for handwriting recognition. Each g is built up from 64 landmarks defining the curve.



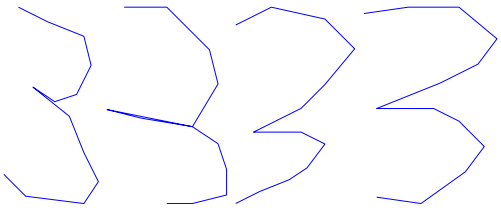
The letter p A data set of 90 curves of the letter p. This letter was taken from the MIT database of Latin letters, initially collected by Rob Kassel at MIT (`ftp://lightning.lcs.mit.edu/pub/handwriting/mit.tar.Z`). Each p is built up from 32 landmarks defining the curve.



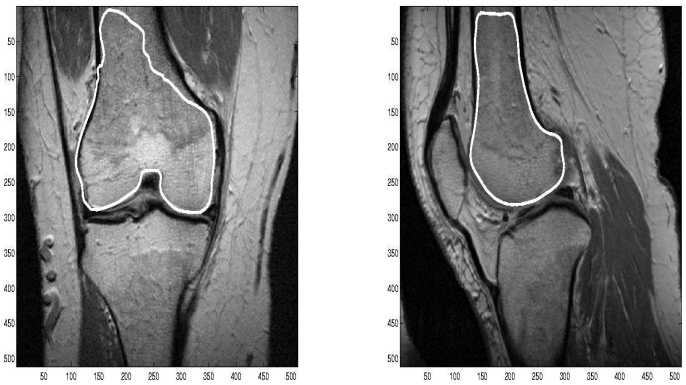
Box-bumps A dataset of 24 synthetic box-bump shapes each consisting of 61 landmarks defining the curve.



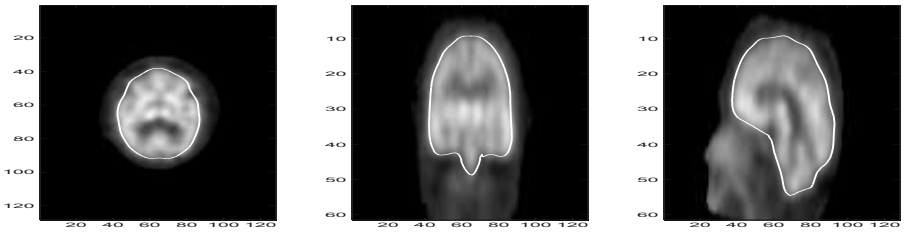
The digit 3 A dataset of 29 shapes of the digit 3. Each digit is built up from 13 landmarks defining the curve.



Knees 3D volumetric Magnetic Resonance Images of knees. Several different contrasts were available but only one type was used in the experiments. 2D-slices of images of this type looks like this with the segmented shape in white.



Brains 3D volumetric SPECT images of brains. 2D-slices of these look like this with the segmented shape in white.



Faces These are 30 head shots of mathematicians (so in a sense they are all outliers). A subset of this dataset consisting of contributors to this thesis is shown here.

Stars These are images of starfish from the Caltech101 database [46]. That database was constructed primarily for object recognition experiments. For the purposes in this thesis the dataset as given in that database is not ideal so a subset is selected for the experiments in this thesis. The images in the selected subset look typically as shown here.

Appendix B

Notations

$(\cdot)^*$	denotes the transpose of the complex conjugate of $(\cdot)$.
$:=:$	denotes correspondence.
$\mathbf{1}_{n_p}$	is a $n_p \times 1$ vector in $\mathbb{C}^{n_p}$ with all elements equal to 1.
$\hat{\alpha}$	is the quantization of α.
$\{\gamma_i\}_{i=1}^{n_s}$	are parameterization functions.
δ	is quantization accuracy.
$\mathbf{\Phi} = \left[\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_{n_m}\right]$	is the shape mode matrix.
λ_i	is the variance of the data in the direction of $\mathbf{\Phi}_i$.
$\{\lambda_{ijk}^p\}$	are patch-mode-parameters.
$\{\lambda_{li}\}$	are shape-mode-parameters.
μ_p	is the probability measure for the shapes $\mathbf{y}_j'$.
$\mathbf{\Omega}$	is the shape space.
Ω_M	is the subspace generated by the model.
$\mathbf{\Omega}_{n_p}^{n_d}$	is the space of all shapes with n_p points in n_d dimensions.
σ	is a standard deviation.
θ_j	is a rotation angle.
A_i	is a small part of the surface with area ΔA_i.
$\mathbf{b}$	is the parameter vector for the model.
$\bar{\mathbf{c}}(s)$	is a continuous mean shape curve.
$\mathbf{c}_i(s)$	is a continuous shape curve.
$\mathbf{c}_i^x(s)$	is a function describing the x-coordinate for a shape curve.
$\mathbf{c}_i^y(s)$	is a function describing the y-coordinate for a shape curve.
c_{ij}	is the element on row i and column j of $\mathbf{C}$.
$\mathbf{C}$	is the shape covariance matrix.
$C(n_m)$	Compactness.
$\{d_i^p\}_{i=1\ldots N}$	are numbers of modes used.
dA	is an area-measure.
dL	is a length-measure.
$D = \{S_1, \ldots, S_{n_s}\}$	is a set of training data examples.
$\{g_i\}_{i=1\ldots n_s}$	are transformations.
$\{\mathbf{g}_i\}$	are profiles describing appearance around each landmark.
$\bar{\mathbf{g}}$	is the mean profile.
$\mathbf{g}_{ij}$	is ground truth point j on shape i.

$\mathcal{G}$	is a group of transformations.
$G(n_m)$	Generality.
L_i	is a small part of the curve with length ΔL_i.
$\mathcal{L}$	is the total description length of the set of all b_i^j.
$\mathcal{L}_j$	is the description length of $\mathcal{Y} = \{b_1^j, \ldots, b_{n_s}^j\}$.
m	is a mean shape.
$\{m_i^p\}_{i=1\ldots N}$	are mean patches.
$\mathcal{M}$	is the model.
$\mathcal{M}(\mathbf{b})$	is a shape generated by the model.
$n_{classes}$	is the number of object classes.
n_d	is the dimension.
n_{dof}	is the degrees of freedom in the transformation group.
n_g	is the number of ground truth points for each object.
n_{kl}	are the variables controlling the basis functions.
n_m	is the number of shape modes used by the model.
n_m^i	is the numbers of patch appearance modes.
n_o^{points}	is the number of outlier points.
n_o^{pixels}	is the number of outlier pixels.
n_p	is the number of landmarks or temporary sample points.
n_s	is the number of training examples.
n_s^i	is the number of images that part i is not missing in.
O	is the ordering matrix.
$p(\mathbf{y}'_j)$	is the probability that the model generates the shape $\mathbf{y}'_j$.
$p_{kl}(t)$	are parameterization function basis functions.
$\mathcal{P}$	is a probability density.
$\mathbf{S}$	is a similarity transform.
$\mathbf{S}_g$	is the covariance for the profiles.
$\mathcal{S}$	is a geometric object.
$\mathcal{S}_0$	is a set of parameter values.
$S_i = \{z_{i,1}, \ldots, z_{i,k_i}\}$	is a set of unordered points $z_{i,j}$ with surrounding patches.
$S(n_m)$	Specificity.
t_{ij}	is the parameterization parameter value giving $\mathbf{g}_{ij}$.
$\{v_l\}_{l=1\ldots d}$	are shape-modes.
$\{v_{ij}^p\}_{i=1\ldots N, j=1\ldots d_i^p}$	are patch-modes.
$\bar{\mathbf{x}}$	is a mean-shape.
$\mathbf{x}_i = [\mathbf{l}_i^1; \ldots; \mathbf{l}_i^{n_p}]$	is a shape vector.
$\mathbf{x}'_j(n_m)$	is the model approximation of $\mathbf{x}$ using n_m modes.
$\mathcal{X} = \{\mathbf{l}_1, \ldots, \mathbf{l}_{n_p}\}$	is a set of points on a geometric object.
$\mathbf{X}_0 = [\mathbf{x}_1, \ldots, \mathbf{x}_{n_s}]$	is a matrix with training set shape vectors as columns.
$\mathbf{X}$	contains the Procrustes residuals.

Bibliography

[1] D. Adalsteinsson and J. Sethian. The fast construction of extension velocities in level set methods. *Journal of Computational Physics*, 148:2–22, 1999.

[2] A. Aho, J. Ullman, and J. Hopcroft. *Data Structures and Algorithms*. Addison-Wesley Pub Co, January 1983.

[3] Y. Amit, U. Grenander, and M. Piccioni. Structural image restoration through deformable template. *J. Amer. Statist. Assoc.*, 414(86):376–387, 1991.

[4] A. Andrew, E. Chu, and P. Lancaster. Derivatives of eigenvalues and eigenvectors of matrix functions. In *SIAM93 J. Matrix Anal. Appl.*, pages 903–926, 1993.

[5] J. Ashburner, J. L. Andersson, and K. J. Friston. Image registration using a symmetric prior-in three dimensions. *Human Brain Mapping*, 9(4):212–225, 2000.

[6] R. Bakis, N. M. Herbst, and G. Nagy. An experimental study of machine recognition of hand-printed numerals. *IEEE Transactions on systems science and cybernetics*, 4(2):119–132, 1968.

[7] A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE trans. on information theory*, 44(6):2743–2760, 1998.

[8] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. *Proc. Fifth International Joint Conf. on Artificial Intelligence*, pages 659–663, 1977.

[9] B. Bascle and R. Deriche. Region tracking through image sequences. *Proc. Int. Conf. on Computer Vision, ICCV'95, MIT, Boston, MA*, pages 302–307, 1995.

[10] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(24):509–522, 2002.

[11] S. Benayoun and N. Ayache. Dense non-rigid motion estimation in sequences of medical images using differential constraints. *International Journal of Computer Vision*, 26(1):25–40, 1998.

[12] P. Besl and H. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.

[13] A. D. Bimbo and P. Pala. Visual image retrieval by elastic matching of user sketches. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(2):121–132, 1997.

[14] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *Internat. J. Comput. Vision*, 11(2):127–145, 1993.

[15] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1998.

[16] F. Bookstein. Size and shape spaces for landmark data in two dimensions. *Statistical Science*, 1(2):181–242, 1986.

[17] M. Brand. Incremental singular value decomposition of uncertain data with missing values. In *Proc. European Conf. on Computer Vision*, 2002.

[18] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, pages 690–6 vol.2, 2000.

[19] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *Int. Journal of Computer Vision*, 22(1):61–79, 1997.

[20] R. Casey. Moment normalization of handprinted characters. *IBM J. Res. Develop.*, pages 548–557, 1970.

[21] Y. Chow, U. Grenander, and D. Keenan. Hands. a pattern theoretical study of biological shapes. *Research Notes in Neural Computing*, 2, 1991.

[22] H. Chui, A. Rangarajan, J. Zhang, and C. Morison Leonard. Unsupervised learning of an atlas from unlabeled point-sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):160–172, 2004.

[23] L. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, 1993.

[24] T. Cootes, A. Hill, C. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. *Information Processing in Medical Imaging. 13th International Conference, IPMI '93 Proceedings*, pages 33–47, 1993.

[25] T. Cootes, A. Hill, C. Taylor, and J. Haslam. Use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12(6):355–365, 1994.

[26] T. Cootes and C. Taylor. Combining point distribution models with shape models based on finite element analysis. *Image and Vision Computing*, 13(5):403–409, 1995.

[27] T. Cootes and C. Taylor. A mixture model for representing shape variation. *Image and Vision Computing*, 17(8):567–573, 1999.

[28] T. Cootes and C. Taylor. Statistical models of appearance for computer vision. Technical report, Division of Imaging Science and Biological Engineering, Manchester, England, 2004.
http://www.isbe.man.ac.uk/∼bim/Models/app_models.pdf.

[29] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Processing*, 61(1):38–59, 1995.

[30] S. Darkner, R. R. Paulsen, and R. Larsen. Analysis of deformation of the human ear and canal caused by mandibular movement. In *MICCAI 2007, Brisbane Australia*, B, oct 2007. Springer Lecture Notes.

[31] R. Davies. *Learning Shape: Optimal Models for Analysing Natural Variability*. PhD thesis, Division of Imaging Science and Biological Engineering, University of Manchester, Manchester, England, 2002.

[32] R. Davies, T. Cootes, and C. Taylor. A minimum description length approach to statistical shape modeling. In *Information Processing in Medical Imaging*, pages 50–63, 2001.

[33] R. Davies, T. Cootes, J. Waterton, and C. Taylor. An efficient method for constructing optimal statistical shape models. In *Medical Image Computing and Computer-Assisted Intervention MICCAI'2001*, pages 57–65, 2001.

[34] R. Davies, C. Twining, T. Cootes, P. Allen, and C. Taylor. Shape discrimination in the hippocampus using an mdl model. In *Information Processing in Medical Imaging*, pages 684–695, 2003.

[35] R. Davies, C. Twining, T. Cootes, J. Waterton, and C. Taylor. A minimum description length approach to statistical shape modeling. *IEEE Trans. medical imaging*, 21(5):525–537, 2002.

[36] R. Davies, C. Twining, T. Cootes, J. Waterton, and C. Taylor. A minimum description length approach to statistical shape modeling. *IEEE Trans. medical imaging*, 21(5):525–537, 2002.

[37] A. Drobchenko, J. Ilonen, H. Kamarainen J, A. Sadovnikov, H. Kalviainen, and M. Hamouz. Object class detection using local image features and point pattern matching constellation search. In *SCIA*, volume 1, pages 273–282, 2007.

[38] I. Dryden and K. Mardia. *Statistical Shape Analysis*. John Wiley & Sons, 1999.

[39] M. Dubuisson-Jolly, S. Lakshmanan, and A. Jain. Vehicle segmentation using deformable templates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(3):293–308, 1996.

[40] H. Eiriksson. Shape representation, alignment and decomposition. Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2001.

[41] A. Ericsson. *Automatic Shape Modelling with Applications in Medical Imaging*. PhD thesis, Lund University, Centre for Mathematical Sciences, Box 118, SE-22100, Lund, Sweden, 2006.

[42] A. Ericsson and J. Karlsson. Aligning shapes by minimising the description length. In *Proc. Scandinavian Conf. on Image Analysis, SCIA'05, Joensuu, Finland*, volume 3540 / 2005, pages 709–718, 2005.

[43] A. Ericsson and K. Åström. An affine invariant deformable shape representation for general curves. In *Proc. Int. Conf. on Computer Vision, ICCV'03, Nice, France*, 2003.

[44] A. Ericsson and K. Åström. Minimizing the description length using steepest descent. In *Proc. British Machine Vision Conference, Norwich, United Kingdom*, 2003.

[45] A. P. Eriksson and K. Åström. Robustness and specificity in object detection. In *Proc. International Conference on Pattern Recognition, Cambridge, UK*, 2004.

[46] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples. In *CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

[47] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples. *Computer Vision and Image Understanding*, 106:59–70, 2007.

[48] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *IJCV*, 71(3):273–303, 2007.

[49] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92, 1973.

[50] R. Fisher. Caviar project, 2005. Ground truth labelled video sequences.

[51] Forsyth and Ponce. *Computer Vision*. Prentice Hall, 2003.

[52] D. Geiger, A. Gupta, L. Costa, and J. Vlontzos. Dynamic programming for detecting, tracking, and matching deformable contours. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(3):294–302, 1993.

[53] S. Gibson and B. Mirtich. A survey of deformable modeling in computer graphics. Technical Report Technical Report TR-97-19, Electric Research Laboratory, 1997.

[54] Ginneken, Frangi, Staal, Haar, Romeny, and Viergever. Active shape model segmentation with optimal features. *IEEE transactions on medical imaging*, 21(8), 2002.

[55] C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(2):285–339, 1991.

[56] J. Gower. Generalized procrustes analysis. *Psychometrica*, 40:33–50, 1975.

[57] E. Greanias, C. Hoppel, M. Kloomok, and J. Osborne. Design of logic for recognition of printed characters by simulation. *IBM Journal*, 1(January):8–18, 1957.

[58] E. Greanias, P. Meagher, R. Norman, and P. Essinger. The recognition of handwritten numerals by contour analysis. *IBM Journal Res. and Develop.*, 7(January):14–21, 1963.

[59] A. Green, M. Berman, P. Switzer, and M. Craig. A transformation for ordering multispectral data in terms of image quality with implications for noise removal. *Geoscience and Remote Sensing, IEEE Transactions on*, 26(1):65–74, 1988.

[60] U. Grenander. *General Pattern Theory*. Oxford University Press, 1993.

[61] U. Grenander and D. Keenan. Towards automated image understanding. *Journal of Applied Statistics*, 20(5-6):89–104, 1993.

[62] M. F. Hansen, M. R. Blas, and R. Larsen. Mahalanobis distance based iterative closest point. In *Spie - Medical Imaging*, feb 2007.

[63] L. Harmon, M. Khan, R. Lasch, and P. Ramig. Machine identification of human faces. *Pattern Recognition*, 13(2):97–110, 1981.

[64] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[65] K. M. Henriksson, K. Wickstrom, N. Maltesson, A. Ericsson, J. Karlsson, F. Lindgren, K. Astrom, T. F. McNeil, and I. Agartz. A pilot study of facial, cranial, and brain mri morphometry in men with schizophrenia. *Psychiatry Research: Neuroimaging*, (147):187–195, 2006.

[66] W. Highleyman. *Linear Decision Functions, with Application to Pattern Recognition*. PhD thesis, Elec. Engrg. Dept. Polytechnic Inst., Brooklyn, N.Y., June 1961.

[67] A. Hill and C. Taylor. A framework for automatic landmark indentification using a new method of nonrigid correspondence. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22:241–251, 2000.

[68] L. Horwitz and G. Shelton JR. Pattern recognition using autocorrelation. *Proceedings of the IRE*, pages 175–185, 1961.

[69] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psych.*, 24:417–441, 1933.

[70] D. Jacobs. Linear fitting with missing data: Applications to structure-from-motion and to characterizing intensity images. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 206–212, 1997.

[71] A. Jain and M. Dubuisson-Jolly. Deformable template models: A review. *Signal Processing*, 71(2):109–129, 1998.

[72] A. Jain and D. Zongker. Representation and recognition of handwritten digits using deformable templates. *IEEE Trans. Pattern Anal.Mach. Intell.*, 19(12):1386–1390, 1997.

[73] F. Kahraman, M. Gokmen, S. Darkner, and R. Larsen. An active illumination and appearance (aia) model for face alignment. In *Proceedings of the CVPR 2007, IEEE Computer Society Workshop on Biometrics*. IEEE Computer Society, jun 2007.

[74] C. Kambhamettu and D. Goldgof. Curvature-based approach to point correspondence recovery in conformal nonrigid motion. *CVGIP:Image Understanding*, 60(1):26–43, 1994.

[75] L. Kamentsky and C. Liu. Computer-automated design of multifont print recognition logic. *IBM Journal*, (January), 1963.

[76] J. Karlsson, A. Ericsson, and K. Åström. Parameterisation invariant statistical shape models. In *Proc. International Conference on Pattern Recognition, Cambridge, UK*, 2004.

[77] A. Kelemen, G. Szekely, and G. Gerig. Elastic model-based segmentation of 3-d neuroradiological data sets. *IEEE Transactions on Medical Imaging*, 18(10):828–839, 1999.

[78] D. Kendall. The diffusion of shape. *Advances in Applied Probability*, 9:428–430, 1977.

[79] D. Kendall, D. Barden, T. K. Carne, and H. Le. *Shape and Shape Theory*. John Wiley & Sons, October 1999.

[80] J. T. Kent. The complex bingham distribution and shape analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(2):285–299, 1994.

[81] C. Kervrann and F. Heitz. Robust tracking of stochastic deformable models in long image sequences. *Proc. Int. Conf. on Image Processing ICIP94'*, 3:88–92, 1994.

[82] M. Kirby and L. Sirovich. Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on pattern analysis and machine intelligence*, 12(1):103–108, 1990.

[83] A. Kotcheff and C. Taylor. Automatic construction of eigenshape models by direct optimization. *Medical Image Analysis*, 2:303–314, 1998.

[84] G. Langs, R. Donner, P. Peloschek, and H. Bischof. Robust autonomous model learning from 2d and 3d data sets. pages 968–976, 2007.

[85] R. Larsen. Non-linear shape decomposition using ISOMAP. In *9th Scnadinavian Conference on Chemometrics*. University of Iceland, aug 2005.

[86] R. Larsen. L1 generalized procrustes 2D shape alignment. *Journal of Mathematical Imaging and Vision*, 2008.

[87] R. Larsen, H. Eiriksson, and M. Stegmann. Q-MAF shape decomposition. In M. A. V. Wiro J. Niessen, editor, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2001, 4th Int. Conference, Utrecht, The Netherlands*, volume 2208 of *Lecture Notes in Computer Science*, pages 837–844. Springer, oct 2001.

[88] R. Larsen and K. B. Hilger. Statistical shape analysis using non-euclidean metrics. *Medical Image Analysis*, 7(4):417–423, 2003.

[89] L. Le Briquer and J. Gee. Design of a statistical model of brain shape. In *Int. Conf. on Information Processing in Medical Imaging*, pages 477–482, 1997.

[90] M. Leventon, W. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *Proceedings. IEEE Conference on Computer Vision and Pattern Recognition*, pages 316–323, 2000.

[91] L. Ljung. *System Identification*. Prentice Hall, 1998.

[92] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.

[93] A. Machado, J. Gee, and M. Campos. Structural shape characterization via exploratory factor analysis. *Artificial Intelligence in Medicine*, 30(2):97–118, 2004.

[94] Z. Mao, X. Ju, J. Siebert, W. Cockshott, and A. Ayoub. Constructing dense correspondences for the analysis of 3d facial morphology. *Pattern Recognition Letters*, 27(6):597–608, 15 April 2006.

[95] T. Marill and D. Green. On the effectiveness of receptors in recognition systems. *IEEE Trans. on Information Theory*, IT-9(January):11–17, 1963.

[96] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *IEEE Transactions on pattern analysis and machine intelligence*, 26(5):530–549, 2004.

[97] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, 1996.

[98] M. Miller, A. Srivastava, and U. Grenander. Conditional mean estimation via jump-diffusion process in multiple target tracking/recognition. *IEEE Trans. Signal Process.*, 43(11):2678–2690, 1995.

[99] M. Minsky. Steps toward artificial intelligence. *IRE*, 1960.

[100] C. Nastar and N. Ayache. Fast segmentation, tracking, and analysis of deformable objects. *Proc. 4th Int. Conf. on Computer Vision, Berlin, Germany*, pages 275–279, 1993.

[101] T. Papadopoulo and M. Lourakis. Estimating the jacobian of the singular value decomposition. In *Proc. European Conf. on Computer Vision, ECCV'00*, pages 555–559, 2000.

[102] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 6(2):559–572, 1901.

[103] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

[104] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[105] M. Rogers and J. Graham. Rubust active shape model search. In *Proc. European Conf. on Computer Vision*, pages 517–530, 2002.

[106] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, (290):2323–2326, 2000.

[107] D. Rueckert, A. Frangi, and J. Schnabel. Automatic construction of 3d-statistical deformation models of the brain using nonrigid registration. *IEEE Trans. on Medical Imaging*, 22(8):1014–1025, 2003.

[108] D. Rueckert, F. Frangi, and J. Schnabel. Automatic construction of 3d-statistical deformation models using nonrigid registration. In *Medical Image Computing and Computer-Assisted Intervention MICCAI'2001*, pages 77–84, 2001.

[109] T. Sakai, M. Nagao, and S. Fujibayashi. Line extraction and pattern detection in a photograph. *Pattern Recognition*, 1(March):233–248, 1969.

[110] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1/2/3):7–42, 2002.

[111] R. Schestowitz, C. Twining, T. Cootes, V. Petrović, C. Taylor, and B. Crum. Assessing the accuracy of non-rigid registration with and without ground truth. In *Proc. IEEE International Symposium on Biomedical Imaging*, 2006.

[112] S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(6):545–561, 1995.

[113] T. Sebastian, P. Klein, and B. Kimia. Constructing 2d curve atlases. In *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, pages 70–77, 2000.

[114] T. Sebastian, P. Klein, and B. Kimia. On aligning curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1):116–125, 2003.

[115] C. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423 and 623–656, 1948.

[116] C. E. Shannon. Communication in the presence of noise. *Proc. IRE*, 37, 1949.

[117] D. Sharvit, J. Chan, H. Tek, and B. Kimia. Symmetry-based indexing of image databases. *Journal of Visual Communication and Image Representation*, 9(4):366–380, 1998.

[118] Y. Shi and W. Karl. Shape reconstruction from unorganized points with a data-driven level set method. In *International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 13–16, 2004.

[119] A. Singh, D. Goldgof, and D. Terzopoulos. *Deformable Models in Medical Image Analysis*. IEEE Computer Society, 1998.

[120] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *J. Optical Society of America*, 4(3):519–524, 1987.

[121] K. Sjöstrand, E. Rostrup, C. Ryberg, R. Larsen, C. Studholme, H. Baezner, J. Ferro, F. Fazekas, L. Pantoni, D. Inzitari, and G. Waldemar. Sparse decomposition and modeling of anatomical shape variation. *IEEE Transactions on Medical Imaging*, 26(12):1625–1635, feb 2007.

[122] C. Small. *The Statistical Theory of Shape*. Springer Verlag, July 1996.

[123] J. Solem. *Variational problems and level set methods in computer vision - Theory and applications*. PhD thesis, Lund university, Sweden, 2006.

[124] P. Sozou, T. Cootes, C. Taylor, E. Di Mauro, and A. Lanitis. Non-linear point distribution modelling using a multi-layer perceptron. *Image and Vision Computing*, 15(6):457–463, 1997.

[125] M. B. Stegmann. *Generative Interpretation of Medical Images*. PhD thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2004. Awarded the Nordic Award for the Best Ph.D. Thesis in Image Analysis and Pattern Recognition in the years 2003-2004 at SCIA'05.

[126] M. B. Stegmann, K. Sjöstrand, and R. Larsen. Sparse modeling of landmark and texture variability using the orthomax criterion. In *International Symposium on Medical Imaging 2006, San Diego, CA*, volume 6144. The International Society for Optical Engineering (SPIE), feb 2006.

[127] J. Sternby and A. Ericsson. Core points - a framework for structural parameterization. In *Proc. Internation Conference on Document Analysis and Recognition, ICDAR'05, Seoul, Korea*, 2005.

[128] C. Studholme, C. Drapaca, and V. Cardenas. Intensity robust viscous fluid deformation based morphometry using regionally adapted mutual information. In *IEEE Engineering in Medicine and Biology*, pages 470–473, 2005.

[129] M. Styner, K. Rajamani, L. Nolte, G. Zsemlye, G. Szekely, C. Taylor, and R. H. Davies. Evaluation of 3d correspondence methods for model building. In *Information Processing in Medical Imaging (IPMI)*, pages 63–75, 2003.

[130] H. Tagare. Shape-based nonrigid correspondence with application to heart motion analysis. *IEEE Trans. medical imaging*, 18:570–579, 1999.

[131] X. Tao, J. Prince, and C. Davatzikos. Using a statistical shape model to extract sulcal curves on the outer cortex of the human brain. *IEEE Trans. on Medical Imaging*, 21(5):513–524, 2002.

[132] J. M. ten Berge. Orthogonal procrustes rotation for two or more matrices. *Psychometrika*, 42(2):267–276, 1977.

[133] D. Terzopoulos and K. Fleischer. Deformable models. *Visual Computer*, 4(6):306–331, 1988.

[134] D. Terzopoulos and K. Waters. Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(6):569–579, 1993.

[135] H. Thodberg. Minimum description length shape and appearance models. In *Image Processing Medical Imaging, IPMI'03*, pages 51–62, 2003.

[136] H. H. Thodberg and H. Olafsdottir. Adding curvature to minimum description length shape models. In *Proc. British Machine Vision Conference*, pages 251–260, 2003.

[137] D. Thompson. *On Growth and Form*. Cambridge University Press, 1917.

[138] C. Twining, T. Cootes, S. Marsland, V. Petrovic, S. R.S., and C. Taylor. Information-theoretic unification of groupwise non-rigid registration and model building. *Proceedings of Medical Image Understanding and Analysis*, 2:226–230, 2006.

[139] C. Twining, R. Davies, and C. Taylor. Non-parametric surface-based regularisation for building statistical shape models. *IPMI*, pages 738–750, 2007.

[140] C. Twining and C. Taylor. Kernel principal component analysis and the construction of non-linear active shape models. In *Proc. British Machine Vision Conference*, pages 23–32, 2001.

[141] C. Twining and C. Taylor. Specificity as a graph-based estimator of cross-entropy. In *Proc. British Machine Vision Conference*, pages 459–468, 2006.

[142] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *pamiito*, 13:376–380, 91.

[143] S. Unger. Pattern detection and recognition. *Proceedings of the IRE*, 47(October):1737–1752, 1959.

[144] M. Uzumcu, A. Frangi, J. Reiber, and B. Lelieveldt. The use of independent component analysis in statistical shape models. In *Proc. Int. Symp. Med. Imag.*, pages 375–383, 2003.

[145] A. Vailaya, Y. Zhong, and A. Jain. A hierarchial system for effcient image retrieval. *Proc. International Conference on Pattern Recognition, ICPR'96*, pages 356–360, 1996.

[146] Y. Wang, B. Peterson, and L. Staib. Shape-based 3d surface correspondence using geodesics and local geometry. In *Proc. Conf. Computer Vision and Pattern Recognition, CVPR'00*, pages 644–651, 2000.

[147] B. Widrow. The 'rubber-mask' technique. i. pattern measurement and analysis. *Pattern Recognition*, 5(3):175–97, 1973.

[148] B. Widrow. The 'rubber-mask' technique. ii. pattern storage and recognition. *Pattern Recognition*, 5(3):199–211, 1973.

[149] A. Yuille, P. Hallinan, and D. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.

[150] H. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. *Computer Vision and Image Understanding*, 80:295–314, 2000.

[151] H. Zhao, C. T., B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *Journal of Computational Physics*, 127:179–195, 1996.

[152] Y. Zheng and D. Doermann. Robust point matching for nonrigid shapes by preserving local neighborhood structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):643–649, 2006.

[153] Y. Zhong and A. Jain. Object localization using color, texture and shape. *Proc. Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 279–294, 1997.