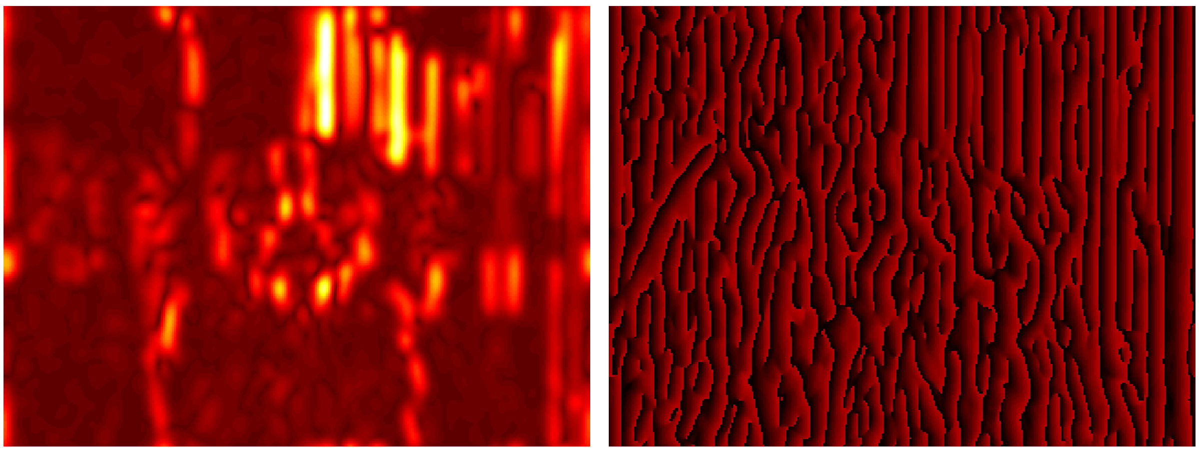
Helsinki University of Technology
Department of Biomedical Engineering and Computational Science Publications
Teknillisen korkeakoulun Lääketieteellisen tekniikan ja
laskennallisen tieteen laitoksen julkaisuja
REPORT A22

INCREMENTAL OBJECT MATCHING WITH PROBABILISTIC METHODS

Doctoral dissertation

Miika Toivanen



Aalto University
School of Science and Technology
Faculty of Information and Natural Sciences
Department of Biomedical Engineering and Computational Science

Helsinki University of Technology
Department of Biomedical Engineering and Computational Science Publications
Teknillisen korkeakoulun Lääketieteellisen tekniikan ja
laskennallisen tieteen laitoksen julkaisuja
REPORT A22

INCREMENTAL OBJECT MATCHING WITH PROBABILISTIC METHODS

Doctoral dissertation

Miika Toivanen

Doctoral dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Information and Natural Sciences for public examination and debate in Auditorium F239a at the Aalto University School of Science and Technology (Espoo, Finland) on the 22nd of October 2010 at 12 noon.

Aalto University
School of Science and Technology
Faculty of Information and Natural Sciences
Department of Biomedical Engineering and Computational Science

Aalto-yliopisto
Teknillinen korkeakoulu
Informaatio- ja luonnontieteiden tiedekunta
Lääketieteellisen tekniikan ja laskennallisen tieteen laitos

Distribution:
Aalto University
School of Science and Technology
Faculty of Information and Natural Sciences
Department of Biomedical Engineering and Computational Science
P.O. Box 12200
FI-00076 AALTO, Finland
Tel: +358 9 470 23172
Fax: +358 9 470 23182
URL: http://www.becs.tkk.fi/
ISSN 1797-3996

ISBN 978-952-60-3361-7 (printed)
ISBN 978-952-60-3362-4 (pdf)
URL: http://lib.tkk.fi/Diss/2010/isbn9789526033624

Espoo 2010

Aalto University

ABSTRACT OF DOCTORAL DISSERTATION	AALTO UNIVERSITY SCHOOL OF SCIENCE AND TECHNOLOGY P.O. BOX 11000, FI-00076 AALTO http://www.aalto.fi
Author Miika Toivanen	
Name of the dissertation Incremental object matching with probabilistic methods	
Manuscript submitted 3.5.2010	Manuscript revised 17.8.2010
Date of the defence 22.10.2010	
☒ Monograph	☐ Article dissertation (summary + original articles)

Faculty	Faculty of Information and Natural Sciences
Department	Department of Biomedical Engineering and Computational Science (BECS)
Field of research	Computational science
Opponent(s)	assist. prof. Josephine Sullivan
Supervisor	prof. Jouko Lampinen
Instructor	prof. Jouko Lampinen

Abstract

This thesis deals with object matching, that is, the problem of locating the corresponding points of an object in an image. Conventional approaches to object matching are batch methods, meaning that the methods first learn the object model from a training set of example images that contain instances of the object, and then use the learned object model to match instances of the same object (or object class) in unseen test images. In matching the object in a test image, the visual correspondence of the points as well as their spatial layout is usually considered simultaneously. Typically the corresponding points of the object are manually pre-annotated in the training images which facilitates the learning process.

In this thesis, a novel approach is taken: The object matching is incremental. This means that the system is given images one at the time, and the images are matched by updating the object model after each processed image. In addition, the methods presented in this thesis use images as such, without utilizing any pre-annotation or pre-segmentation information, so the task can be considered of being extremely difficult. Although an incremental learning procedure has been presented before that learns an object model incrementally and detects whether the object appears in a test image or not, the proposed methods are the first ones that try to locate the corresponding points of the object by handling images one by one.

Like in the traditional object matching methods, the object model of the proposed methods also consists of spatially distributed local features. The adopted approach to the incremental matching is Bayesian; the likelihood corresponds to the appearance of the features and the prior distribution to the spatial layout. Recursive Bayesian formulas are utilized in updating the object model after each processed image and particle Monte Carlo methods are used to sample the posterior distribution. Results show that the methods are able to locate the corresponding points with similar accuracy as the batch matching methods. The learned object model can also be used in detection tasks, and according to the results, the detection capabilities of the presented methods are (almost) on a par with the batch detection methods, and superior to the reference incremental detection method.

Keywords Object matching, incremental learning, Bayesian inference, Monte Carlo methods	
ISBN (printed) 978-952-60-3361-7	ISSN (printed) 1797-3996
ISBN (pdf) 978-952-60-3362-4	ISSN (pdf) 1797-3996
Language English	Number of pages 115 p.
Publisher Department of Biomedical Engineering and Computational Science	
Print distribution Department of Biomedical Engineering and Computational Science	
☒ The dissertation can be read at http://lib.tkk.fi/Diss/2010/isbn9789526033624	

Aalto-yliopisto

VÄITÖSKIRJAN TIIVISTELMÄ	AALTO-YLIOPISTO TEKNILLINEN KORKEAKOULU PL 11000, 00076 AALTO http://www.aalto.fi
Tekijä Miika Toivanen	
Väitöskirjan nimi Incremental object matching with probabilistic methods	
Käsikirjoituksen päivämäärä 3.5.2010	Korjatun käsikirjoituksen päivämäärä 17.8.2010
Väitöstilaisuuden ajankohta 22.10.2010	
☒ Monografia	☐ Yhdistelmäväitöskirja (yhteenveto + erillisartikkelit)

Tiedekunta	Informaatio- ja luonnontieteiden tiedekunta
Laitos	Lääketieteellisen tekniikan ja laskennallisen tieteen laitos
Tutkimusala	Laskennallinen tiede
Vastaväittäjä(t)	assist. prof. Josephine Sullivan
Työn valvoja	prof. Jouko Lampinen
Työn ohjaaja	prof. Jouko Lampinen

Tiivistelmä

Väitöskirja liittyy kohteiden sovitukseen, jolla tässä työssä tarkoitetaan kohteen vastinpisteiden tarkkaa paikantamista kuvasta. Perinteiset kohteensovitusmenetelmät ovat n.s. erämenetelmiä, joissa ensin opitaan kohteen esitys käsittelemällä samanaikaisesti iso määrä kohteen ilmentymiä sisältäviä opetuskuvia, jonka jälkeen opittua esitystä käyttäen sovitetaan kohde uudessa testikuvassa, joka sisältää ilmentymän samasta kohteesta (tai kohdeluokasta). Kohdetta sovitettaessa otetaan yleensä samanaikaisesti huomioon sekä vastinpisteiden visuaalinen vastaavuus että niiden keskinäinen muoto. Tyypillisesti kohteen vastinpisteet ovat etukäteen manuaalisesti merkitty opetuskuvissa, mikä helpottaa kohteen esityksen oppimista.

Tässä väitöskirjassa esitetään uusi menettelytapa; inkrementaalinen kohteensovitus, mikä tarkoittaa kuvien syöttämistä järjestelmälle yksi kerrallaan ja kohteen esityksen päivittämistä jokaisen prosessoidun kuvan jälkeen. Tässä työssä kehitetyt menetelmät käsittelevät kuvia sellaisenaan, ilman minkäänlaisia manuaalisia esimerkintöjä, joten tehtävää voidaan pitää varsin haastavana. Vaikkakin aiemmin on esitetty menetelmä, joka oppii kohteen esityksen inkrementaalisesti ja tunnistaa, esiintyykö opittu kohde testikuvassa, työssä esitetyt menetelmät ovat ensimmäiset, jotka pyrkivät löytämään kohteen vastinpisteet käsittelemällä kuvat yksi kerrallaan.

Kuten perinteiset kohteensovitusmenetelmät, myös tässä työssä kehitetyt menetelmät käyttävät kohteen esityksenä keskinäisen muodon huomioivia paikallisia piirrepisteitä. Sovituksessa käytetään bayesilaisia menetelmiä. Pisteiden ulkonäkö mallinnetaan uskottavuusfunktion avulla, kun taas pisteiden geometriset suhteen mallinnetaan priorijakaumalla. Kohteen esityksen päivittämiseen käytetään rekursiivisia bayesilaisia kaavoja, ja posteriorijakauman näytteistämiseen käytetään partikkeli Monte Carlo -menetelmiä. Tulokset osoittavat työssä esitettyjen menetelmien saavuttavan samankaltaisen sovitustarkkuuden kuin vertailtavat erämenetelmät. Systeemin oppimaa kohteen esitystä voidaan käyttää myös tunnistustehtävissä. Esitettyjen menetelmien saavuttamat tunnistustulokset ovat lähes yhtä hyvät kuin erätunnistusmenetelmillä ja selvästi paremmat kuin vertailtavalla inkrementaalisella tunnistusmenetelmällä.

Asiasanat Kohteen sovitus, inkrementaalinen oppiminen, bayesilainen päättely, Monte Carlo menetelmät

ISBN (painettu) 978-952-60-3361-7	ISSN (painettu) 1797-3996
ISBN (pdf) 978-952-60-3362-4	ISSN (pdf) 1797-3996
Kieli englanti	Sivumäärä 115 s.
Julkaisija Department of Biomedical Engineering and Computational Science	
Painetun väitöskirjan jakelu Department of Biomedical Engineering and Computational Science	
☒ Luettavissa verkossa osoitteessa http://lib.tkk.fi/Diss/2010/isbn9789526033624	

Preface

This work was carried in the Department of Biomedical Engineering and Computational Science, Aalto University School of Science, during the period November 2005 - April 2010 and was funded by the ComMIT graduate school. I thank prof. Jouko Lampinen for supervising this interesting study and for sharing his enthusiasm on science in general. Also, docent Aki Vehtari deserves my gratitudes for helping me out various times, as well as Dr. Rolf Würtz for providing me the research facilities during my delightful three-month stay in Bochum, Germany.

The road to the Ph.D. was hard but difficult; out of the eleven submitted articles, only four was accepted. These adversities have been calling for support which I have received from Satu - thank you. I also thank my family and our dog Lily whose wisdom we can only underestimate.

Contents

List of abbreviations

PCA	Principal component analysis
SIFT	Scale invariant feature transform
EBGM	Elastic bunch graph matching
AAM	Active appearance model
DC	Direct current
ML	Maximum likelihood
MAP	Maximum a posteriori
MC	Monte Carlo
MCMC	Markov chain Monte Carlo
SMC	Sequential Monte Carlo
PMC	Population Monte Carlo
ROC	Receiver operating characteristic
IMM-DTU	Informatics and Mathematical Modelling, Technical University of Denmark
AUR	Area under ROC
EER	Equal error rate

Chapter 1

Introduction

1.1 Background

The main goal of computer vision is to mimic the vision system of humans, and possibly of other mammals. The goal is far from being reached; despite recognizing objects in our environment is effortless to us, for artificial computer systems it is extremely demanding. Evolution has refined the human brain into such that its capabilities to handle visual information obtained from the three-dimensional world are outstanding. In every moment, we can easily recognize and locate dozens of objects surrounding us although the objects may appear in any scale or pose, the view point can be arbitrary, the objects can be partially obstructed by other objects, and the internal variability of certain object class may be huge. Humans possess also a large amount of prior information about objects in general, which help in learning to recognize new objects.

Images are two-dimensional projections of the three-dimensional world. A computer 'sees' the images as numbers, and using different mathematical algorithms, the computer vision systems aim to find something 'interesting' in the images such as edges which would aid in recognizing objects. Basically, this is similar to how images are processed in the early stages of the mammalian vision system. It is the less well known deeper structures of our visual system which enable the recognition of complex objects, and whose behavior is difficult, if not impossible, to emulate with artificial systems. The recognition of the computer vision systems is highly dependent on the appearance and shape of the objects. Also, different invariances — invariance against the location, scale and orientation of the objects — must be somehow explicitly added in the artificial systems, as for instance an (asymmetrical) object posing upwards and the same object posing downwards are basically treated as different objects. Also, unlike with the human vision system, the object recognition of an artificial system is often dependent on the lighting conditions of the object.

One aspect of computer vision is object matching. Somewhat different definitions of object matching in computer vision have been presented. In this thesis, object matching means locating corresponding points in the images. The corresponding points between, for instance, two images consist of pairs of points that correspond to the same points on the physical object (see Figure 1.1). Basically, the number of corresponding points between objects is infinite; in digital images, the resolution of the images limits the density of the correspondence mapping. Practical object matching methods often use a sparse set, consisting of a few dozen corresponding points. In addition to relying on the visual appearance of the points, their spatial relationships are also usually considered.

The object matching problem is typically formulated as follows: given a set of training images, also known as reference images, learn the model, or representation, of the object and match the object in an unseen test image using the learned object model. The training images and the test image contain so called instances of the object. For example, to locate a car in a test image, the artificial system is given images that contain exemplars of different cars. Because the appearance and shape of cars can vary a lot, the number of training images should be moderately high in order to learn a reliable model of a car. Often the corresponding points are manually pre-annotated in the training images which facilitates the learning process, and the actual problem is in matching the test image. Possibly the most prominent object matching methods are the Elastic Bunch Graph Matching method by Wiskott et al. (1997), and the Active Appearance Model of Cootes et al. (2001). An example of a Bayesian approach to object matching was presented by Tamminen and Lampinen (2006).

Object detection and recognition are another subfields of computer vision, which can be related to object matching. Again, different definitions exist, but usually object detection means answering the question ”does the learned object appear in the test image”, whilst object recognition tries to refine which of the learned objects (if any) appears in the test image. Object detection and recognition do not necessarily require the object to be matched in the test image. A prominent example of such an object detection method is the constellation model of Fergus et al. (2003) which learns the object model from natural, un-annotated training images by extracting parts in them and inferring with probabilistic methods which of the parts constitute the object model. Likewise, parts are extracted in the test image and

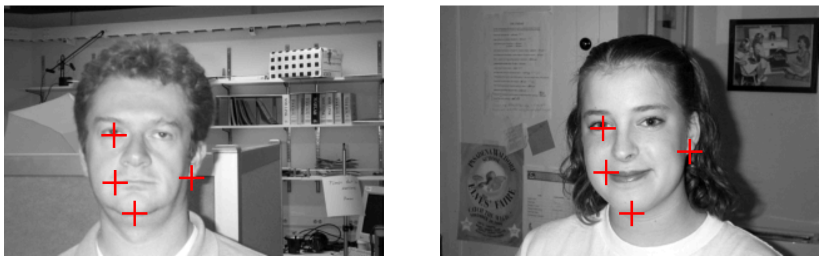


Figure 1.1: Two images which contain instances of the same object. Four pairs of corresponding points have been marked with crosses.

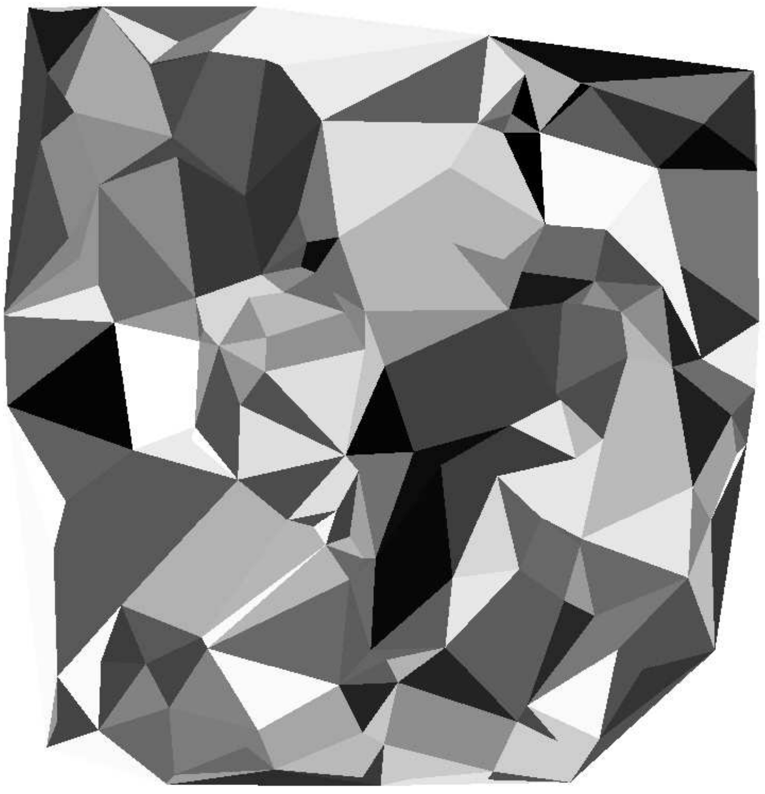


Figure 1.2: An example image which contains an instance of the same object as Figures 1.3 and 1.4. See text in Section 1.2 for details.

compared with the object model. As a result, a probability for the object appearing in the image is achieved without the object being accurately located.

1.2 Overview

This thesis deals with object matching, that is, the problem of locating the corresponding points in images. Learning the object model is taken to be incremental by nature. This means that the system is given images sequentially, one by one, and the matching result of the processed images is exploited in matching the next image. Incremental learning (also called online learning) contrasts with batch learning which handles all the training data simultaneously. Actually, in incremental learning there is no traditional separation into training set and test set as the nature of the data is dualistic; each image that is fed into the system serves first as a test image but becomes a training image for the upcoming images after having being processed. Of course a batch method can similarly expand the training set one image after another, but this requires that all the processed images are stored in

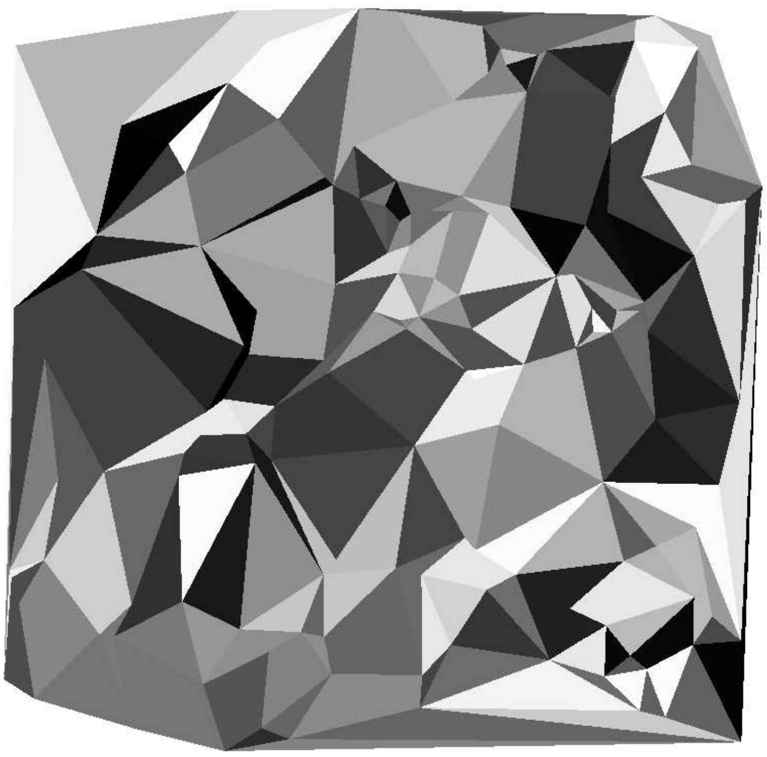


Figure 1.3: An example image which contains an instance of the same object as Figures 1.2 and 1.4. See text in Section 1.2 for details.

memory and that the learning begins again every time a new image is introduced. On the contrary, incremental methods update the object model recursively. As an object model is a compressed representation of the whole data, most of the information in the images is discarded after they have been incrementally processed. Thus, the performance of an incremental method is expected to be weaker than that of a batch method which processes all the available data simultaneously. However, incrementality is a desired property, for several reasons. Incremental learning is natural for all the living organisms as their way of perceiving the environment is sequential. A child probably learns a visual representation of a certain object by updating the representation after each view of the object, instead of first storing all the views in memory and then processing them simultaneously. Incremental learning, in principle, is faster and requires less memory than batch learning. Also, in many practical problems the data (images) arrive sequentially, which calls for incremental methods. Another nice thing about the incremental approach is that the learning can be stopped anytime as soon as the object model is 'ready'.

Figures 1.2, 1.3, and 1.4 aim to demonstrate how difficult the incremental object matching is for a computer system. Each of the images contains a same fully rigid object but in different orientations and scales, and the task is to locate the object in the images. As the system is incremental, the images appear in different pages and should be viewed only once, without watching the previous images again. Because one has not seen the seemingly meaningless object before, and because the background of the object differs very little from the object, finding the corresponding points is surprisingly difficult. Because computers lack prior information about objects and backgrounds, matching the presented artificial object or matching a real object, such as a dog, are equally problematic for a computer vision system — or actually, matching dogs is more difficult due to the changes in shape and appearance between different object instances. As a side note, the solution to the problem is found on page 86 which shows the matching result of the method that is developed in this thesis.

1.3 Aims of the thesis and author's contributions

The aim of this thesis is to develop probabilistic models and methods for incremental object matching. An important and complicating factor is the lack of manual pre-annotations or pre-segmentations, that is, no prior information about the location of the object in the images is utilized. The problem can thus be reformulated as follows: Given a set of natural images in which instances of the same object appear, locate the corresponding points of the object by processing the images incrementally. To the best knowledge of the author, no other published method has tried the same. The developed methods are also used for detection purposes. The object model used in this thesis is a sparse set of feature points. Bayesian methodology is adopted as it offers a natural framework for modeling the appearance and shape of the points separately, and for recursively updating the object model. In the Bayesian treatment, the uncertainties and lack of knowledge are expressed in a mathematically consistent way which also makes it an appealing approach. To approximate the multi-dimensional posterior distributions, particle Monte Carlo methods are used due to their capabilities to handle multimodal distributions. A major difficulty in these methods is the initialization of the particles.

The thesis is organized as follows. Chapter 2 reviews several object matching and detection methods presented in the literature, having an emphasis on Gabor features as they are used in this thesis. Chapter 3 introduces the Bayesian inference and some numerical sampling methods. The minor contribution in this chapter is the comparison of the particle Monte Carlo methods. Chapter 4 is the main contribution of the thesis as it first presents how the Bayesian methodology is theoretically applied in the studied problem, and then presents three practical implementations of the theoretical framework. The methods are published in (Toivanen and Lampinen, 2009a), (Toivanen and Lampinen, 2009b), (Toivanen and Lampinen, 2009c), and (Toivanen and Lampinen, 2010). Chapter 4 also discusses the differences between the methods and the problem of tuning the parameters. Experimental results are given in Chapter 5, and Chapter 6 concludes the work.

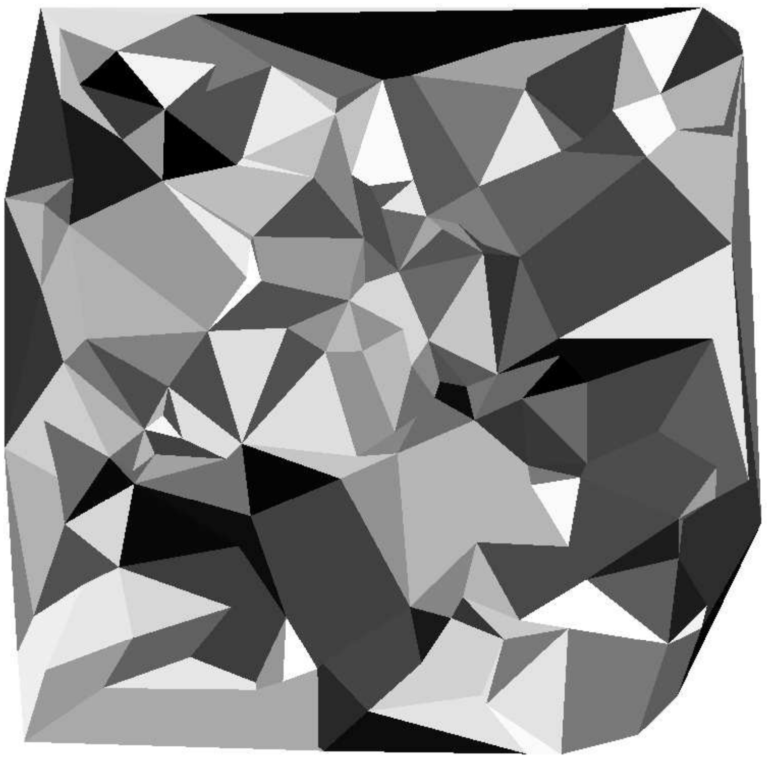


Figure 1.4: An example image which contains an instance of the same object as Figures 1.2 and 1.3. See text in Section 1.2 for details.

The background of this work is on the batch object matching method of Tamminen and Lampinen (2006), the occlusion model of which was utilized in this work to match novel objects without training. The idea for this came from Prof. Lampinen, and the author has thereafter developed the framework and the numerical sampling methods almost solely. The implementations of the methods and the experiments were coded by the author.

Chapter 2

A review of object models

2.1 Introduction

This chapter gives a review of different object models that have been used in object matching and detection. The models differ in their complexities and abilities to handle different complicating factors which are inherent in computer vision. The most basic complicating factor is the internal variability of the object class as the instances of the same object class always look different, a good example being a human face. A proper object matching method should also be able to cope with the basic transformations of the object — translation, scale and rotation. This means that the object instances should be matched despite their location, size and orientation differing from the training images. A more challenging task is to handle affine transformation. Robustness against occlusion is also a wanted property, meaning that the instances should be correctly matched even if parts of them are obscured by another object.

There are many possibilities to categorize different object models. The lowest common denominator between different models is probably the use of features. This means that the images are first somehow pre-processed so as to extract features with which the object matching or detection task is easier and faster to solve. The features can be based on the shape of the object, on its visual appearance, or both. Care must be taken in the feature extraction step, as information is typically wasted. In a successful case, the extracted features are fast to compute and use, and yet they are discriminative enough so that the object instance can be separated from the background in the test image. Perhaps the most meaningful division of the available models comes from the use of features; either the object of interest is modeled with one global feature, or with many local features which typically have spatial relationships. Exemplars of both types of features are presented in this chapter, the emphasis being in local features as they are used in this thesis. Having presented the features, the most competent object matching methods and detection methods — whose aim is not so much to localize the object as to detect whether it appears in the test image or not — are reviewed. Finally, the Gabor features, which are used in this thesis, are presented. It should be noted, however, that the proposed Bayesian framework for the incremental matching is not limited to the use of Gabor features, or even to the use of the local feature based approach.

2.2 Global features

The global features that are considered here are the shape features and the texture features. Basically, these features could also be used as local features, but here the focus is in their use as global features.

2.2.1 Shape features

Shape features are used when the primary interest is on modeling the shape of the object instead of its appearance. In shape modeling the test image is segmented into regions whose shape is compared with a model region using different shape features. This requires the object instance to be correctly segmented in the test image. For example, if a human face is searched in the image, in a successful case one of the image segments comprises the human face but nothing else. Hence, the success of using the shape features is strongly dependent on how the image is segmented, which is ambiguous. Another drawback of the segmentation approach is its sensitivity to occlusion, as whole segments are used in comparison and the segment of an occluded object is always different from the model segment. Also, scale difference is usually a problem. The shape features can be divided into contour based and region based features.

Of the contour based features, probably the simplest ones consider the length or curvature of the object boundary. A more advanced choice is to use chain codes, so that the boundary is coded with indices representing the 4- or 8-connectivity directions of the boundary at discretized steps (Jain, 1989; Gonzalez and Woods, 1993). Another option is to compute the distance from the centroid of the object to the boundary as a function of the angle. A more sophisticated option is to use B-splines, that is, to represent the contour with a piecewise polynomial interpolation (Sonka et al., 1999). A common choice is B-splines of third order as it is the lowest order including the change of curvature. The simplest region based approach is to compare the area of the region while more advanced methods consider also eccentricity and elongatedness (Sonka et al., 1999). Of the more recent methods, (Opelt et al., 2006b) combine the boundary and centroid knowledge to perform object detection, and (Jiang et al., 2009) represent the shape as an oriented edge map and use the mean of the training shapes for matching. Although being simple, shape features can be very efficient for certain object matching and detection tasks.

2.2.2 Texture features

A global texture feature refers to the appearance of the whole object. Modeling the joint appearance of all the pixels of the object can be advantageous compared with local feature based methods which model the appearance only around the feature points. Basically, having a local feature in each pixel of the object does the trick but this would be computationally unfeasible.

The simplest texture feature based method can be used only for classification: The test image is classified to the training image with maximum correlation in the image space whose dimension is the number of pixels in the images (Brunelli and Poggio, 1993). The method is highly dependent on the lighting conditions and is computationally expensive. A more sophisticated method is the Eigenfaces method (Turk and Pentland, 1991; Pentland et al., 1994) in which the object model is built from the training images by computing the object appearance as a linear sum of the mean appearance vector and linear basis vectors. The basis vectors are formed with the principal component analysis (PCA) framework (Chatfield and Collins, 1980; Bishop, 2006). Before computing the principal components, the images must be aligned to exclude the background effects from the representation. A test image is matched by finding the coefficients in the linear expansion so that, e.g., the pixel-wise squared difference of the intensity values between the model and the image is minimized. It is also straightforward to use the Eigenfaces method to synthesize new images. A probabilistic implementation of the Eigenfaces method is given by Moghaddam et al. (1998). In face classification, Eigenfaces can be problematic in a sense that much of the visual variation typically stems from the different lighting conditions and many principal components are 'wasted' in capturing this information which is non-essential for the classification purpose. A remedy was suggested by Belhumeur et al. (1997) in forms of Fisherfaces by maximizing the ratio of between-class variation to the within-class variation. The traditional Eigenface models are incapable to separate the appearance and shape of the object as these are combined into a single model, making it difficult to model objects with larger morphological changes.

2.3 Local features

In local feature based methods the objects are modeled as a collection of local features which are (typically) constrained by their mutual relations. Using a collection of local object descriptors is more robust against occlusion than comparing globally the whole object, and yields better results in matching non-rigid objects. Local methods have become the dominant framework during the last decade in the object detection community (Zhang et al., 2007).

A typical local feature based object matching method uses training images where several landmark points of the object instances are manually annotated. If the object is a human face, for instance, the annotations should cover eyes, mouth, etc. The reference appearance of the features and reference shape (which is also formed from the annotations) are learned from the training images. The combined reference model is then matched in a test image which is assumed to contain an instance of the learned object. This requires a similarity measure to be defined which measures the visual similarity between the reference appearance and the pixels in the test image. A good match is such that all the features have high similarities in the test image while also being in an approximately similar configuration as the reference shape. The differences between various feature based methods are, for instance, on the used features, the shape model, the similarity measure, the matching scheme, and on the use of a probabilistic framework. This section presents different local features which are divided into low-level and complex features.

2.3.1 Low-level features

The simplest, and probably also the longest-lasting, feature is basically a vector of the gray scale values in the image around the feature point, called a patch or a template. The idea is basically the same as in using the global texture feature except the object model consists of many local templates instead of one global template. Fitting the template in the test image is usually called template matching or gray-level matching (Jain, 1989; Cox, 1995). Normalized cross-correlation or sum of squared differences between the gray-level values of the model template and the gray-values extracted from the image are conventionally used as a similarity measure. To make the matching invariant to the absolute brightness, the mean values of the templates should be subtracted, and to achieve contrast invariance, the templates should be divided by their variance. Template matching is quite sensitive to noise in the images as well as to intra-class variation. Also, scale and orientation changes must be handled separately. Simple gray-level similarities are sensitive to occlusion if all the pixels under the template are used for the similarity, but template matching methods that can handle occlusion have been proposed (Nguyen et al., 2001; Jurie and Dhome, 2002). Despite being an old method, template matching techniques are still a topic of wide interest (Yoon et al., 2008; Jung et al., 2010).

Another simple approach is to use edges as features. To utilize edge information, edges must first be extracted from the image. A number of edge detectors have been developed (see, for instance, (Sonka et al., 1999)). Using edges as features requires a similarity between two edges to be established. There are different ways to translate the edge information into a similarity measure, such as using the geometric attributes, like the line length and orientation. Perhaps the most widely used approach is to utilize the Hausdorff distance (Rucklidge, 1997), which has the advantage of being invariant to affine transforms and robust against noise and occlusion. Another edge based similarity is based on the generalized Hough transform (Ballard, 1981; Cho, 2006). A drawback of edge based methods is the necessary binarization of the images, which makes the matching invariant only against a narrow range of illumination changes, as it depends on the image contrast. Steger (2002) has proposed an alternative similarity measure which is robust also against nonlinear illumination changes. The main drawback of edge matching is their non-specificity as lines occur frequently in natural images. Thus, the method is prone of false matches when the background in the test image is complex. A rarer feature is a corner, which is widely used in stereo matching (see, for instance, (Smith et al., 1998; Zhao et al., 2008)). From an object matching point of view, a corner might be even too rare for having a dense feature model.

It is also possible to use color as a feature which is basically more discriminative due to the two extra dimensions. For instance, the human skin forms a clear distinct cluster in the color space (even when different races are considered), which can be modeled with a Gaussian distribution (Yang et al., 1997). However, there are objects which appear in varying colors, such as cars, making it unfeasible to use color for generic object modeling. In addition, color matching is highly dependent on the lighting conditions, and moreover, the color information is not always available.

Histogram features can be used to represent different characteristics of shape and appearance (Mikolajczyk and Schmid, 2005). The simplest histogram depicts the distribution of the intensity values on a region of the image (Jain, 1989). Common features used to describe histograms are the absolute and central moments of different order and the entropy of the histogram (Gonzalez and Woods, 1993; Jain, 1989). The problem of the histogram approach is that the histogram carries no information about the relative position of pixels with regard to each other (Gonzalez and Woods, 1993). Hence, the histograms are rather used for classification than for accurate matching.

2.3.2 Complex features

The Scale Invariant Feature Transform (SIFT) framework, proposed by Lowe (1999, 2004), is an efficient and widely used local feature generation method. The SIFT features are invariant to translation, scale and rotation, and also partially invariant to illumination changes and affine or 3D projections. The SIFT method combines feature detection and description, which is based on the gradient information in the detected regions. The rotation invariance is realized by considering the image gradient in the detected region in (typically eight) discrete orientations, and likewise for scale invariance. The invariance against noise is achieved by down-sampling the gradient vector. The SIFT feature can be interpreted as a 4-dimensional histogram presentation as there are the orientation, scale, and the two spatial dimensions. A problem with SIFT is that it might be too invariant, and features that are more selective might be needed for a generic object recognition task. SIFT features have been an inspiration for various other features; the publication (Lowe, 2004) has over 6000 citations.

Lazebnik et al. (2004, 2005) introduced two sophisticated histogram based features. A spin feature is a rotation-invariant histogram of image intensities within a region of interest. The histogram is two-dimensional, dimensions being the intensity and the distance from the center of the region. A Rotation Invariant Feature Transform resembles the SIFT feature but the region of interest is divided into rings of equal width and the gradient orientation histogram is computed in each ring in order to obtain inherent rotation invariance. PCA-SIFT features of Ke and Sukthankar (2004) use the SIFT procedure, but instead of discretizing the region spatially, each pixel is used to compute the histogram which is then reduced with PCA to lower dimensionality to produce a feature vector which is smaller than the standard SIFT feature. A Speeded-Up Robust Feature (Bay et al., 2006) describes the Haar-wavelet responses within the region of interest and is, according to the authors, faster to use and more robust than the SIFT feature. A Gradient Location Orientation Histogram (Mikolajczyk and Schmid, 2005) is another extension of the SIFT feature. The aim is to increase robustness and distinctiveness by computing the SIFT histograms in a dense log-polar location grid. The size of the features is reduced, akin to the PCA-SIFT method, with PCA. The Spatial-color Mixture of Gaussians is a recently proposed feature which improves over the standard color histogram feature by also considering spatial layout of the colors (Wang et al., 2007). Ersi and Zelek (2006) use the histograms of Gabor responses as a feature.

Another sophisticated option to describe a local appearance is to use the responses of various (real or complex valued) filters. In this study, Gabor filter responses are used, and a succinct introduction of filtering theory is given in Section 2.5 which represents the Gabor filters. Here it should be mentioned that other possible filters include Gaussian filters (Sullivan et al., 2001), Gaussian derivatives, and mixtures of a complex exponential term and a Gaussian derivative or polynomial (Mikolajczyk and Schmid, 2005).

2.4 Object matching and detection methods

Previous sections have described several global and local features that can be used in object modeling, excluding Gabor features that are introduced in greater detail in Section 2.5. Next, some prominent object matching and detection methods that utilize the features are briefly reviewed. All the reviewed object matching methods require pre-annotated or pre-segmented training images whilst the annotations are not needed for many of the object detection and classification methods.

2.4.1 Object matching methods

Lades et al. (1993) have presented the Elastic Graph Matching algorithm which they use for face matching and recognition. It applies Gabor features which are extracted at a rectangular grid that is positioned in the passport photo -like training images, together with the geometrical relationships of the grid in terms of edges between them. In matching a test image, a cost function, which is the sum of quadratic edge distortions and negative similarity term of the Gabor responses, is minimized. Each face graph of the training images is separately matched in the test image and the one with the smallest cost function is recognized as being the face in the test image. The method was extended by (Wiskott et al., 1997) to use manually annotated feature locations and to stack the reference Gabor responses into something called bunch graphs; the method is termed Elastic Bunch Graph Matching (EBGM). The Gabor jet (see Section 2.5) of the bunch that yields the highest similarity is used independently of the other features when matching the test image. EBGM method was the top performer in recognizing faces in the FERET (Face Recognition Techonology) evaluation contest (Phillips et al., 2000).

Active Appearance Models (AAM) by Cootes et al. (2001) is another method that matches test images accurately, given manually annotated training images. In AAM, the object model is formed from the principal components for the appearance (as in the Eigenfaces method) and shape, and also for the mutual relations between them. The training images are warped to match each other, and the shapes in the training images are aligned with Procrustes analysis (Goodall, 1991). In order to obtain invariance against lighting conditions, the training images are normalized to have zero mean and unit variance.

Tamminen and Lampinen (2006) (see also (Tamminen, 2005)) have proposed a probabilistic method for matching the object model in a test image. The object model is learned, again, from annotated training images and consists of Gabor feature based appearance distribution and a Gaussian distribution for the shape. These are combined into a Bayesian posterior distribution for the feature locations in the test image which is matched with Monte Carlo methods. The method, presented in this thesis, is based on the method of (Tamminen, 2005; Tamminen and Lampinen, 2006).

The EBGM, AAM, and the Bayesian method of (Tamminen and Lampinen, 2006) are the most important ones with respect to this thesis as they represent the object with a rather dense graph of feature points which is accurately matched in the test images. Some other object matching methods have also been proposed that deserve a mention here. Sullivan et al. (2001) match objects with a rigid object model using the Bayesian framework and factored sampling, Yan et al. (2004) presented a Bayesian face matching and recognition model which combines global and local texture features, and (Würtz, 1997) uses a hierarchical Gabor feature based system for face matching and recognition.

2.4.2 Object detection methods

The primary goal of the methods presented in this subsection is to either detect whether there is an instance of the learned object in the test image or not, or to classify the content in the test image into one of the learned object classes. Many of these methods also provide an approximative location of the object. Albeit the method of this thesis finds a set of corresponding pixels in the images, that is, matches the feature points of the object, it shares some similarities with certain object detection methods which are reviewed here.

Perhaps the most important of these methods are the constellation models (Burl et al., 1998; Weber et al., 2000; Fergus et al., 2003; Fei-Fei et al., 2003, 2007) which model the object as a flexible constellation of parts, using probabilistic representations. In the constellation models, the distributions for the appearance of the parts and for their mutual positions are learned from a natural, unannotated training set. In each training image, a set of interesting regions (less than one hundred) is extracted whose appearance is represented with a PCA reduced gray-scale template. These vectors can be interpreted as the candidate features of the object. The object model is learned by finding with Expectation Maximization algorithm (Bishop, 2006) a maximum likelihood estimate of the model parameters — or by variational Bayesian methods an estimate of the posterior distribution — , that is, visually similar features (less than ten) in a similar configuration. In a test image, regions are extracted again and compared with the learned object model. Each region whose appearance is similar with the learned features gives a probabilistic vote for the presence of the object, and similarity of the shape of these regions with the reference shape gives another probability for the presence of the object. Combining these gives the likelihood ratio (*object / no object*) which, while exceeding a threshold, is an indication of the test image containing the object. The constellation models are able to model diverse objects with either tight variance in the appearance space but loose shape model (e.g. spotted cats), or a rather rigid shape model but more visual variance for some parts (e.g. faces, motorbikes). Hence, the models are able to capture the essence of each object, be it the appearance of features or their shape or both. A problem with the constellation models is the computational complexity which sets upper limits for the numbers of the extracted regions and features in the object model. If none of the extracted regions in a test image corresponds to the object's features, the object is unlikely to be detected. Also, the model of (Fergus et al., 2003) requires hundreds of training images, and it takes more than 24 hours to learn an object model. A partial remedy was suggested by Fei-Fei et al. (2003) who incorporate prior information, gained from other object categories, in the learning state so as to decrease the number of training images into few. Fei-Fei et al. (2007) came up with an incremental version of the constellation model of (Fei-Fei et al., 2003) which uses a recursive version of the variational Bayesian method. In the incremental method, the training images can be fed gradually and the object model is updated after each added image, or group of images. The main difference of the constellation models with the studied model is the number of possible locations for the features; in constellation models, it is typically a few dozens, among which the corresponding features are sought, whereas in this thesis, the posterior probability distribution of the feature locations is defined for all the image pixels and there are no limits for where to seek the features. Naturally this makes it impossible to exhaustively evaluate all the possible feature combinations, as is done in the constellation models; instead, Monte Carlo sampling methods are resorted to find probable feature locations.

Many methods have been proposed that resemble the constellation models in that they also model the object with parts and shape. Mikolajczyk et al. (2006) presented a part based model with a hierarchical tree structure to detect multiple classes. Another method for detecting multiple classes is the method of (Murphy-Chutorian et al., 2005) which uses both color and Gabor features which are shared between and within different object models. The geometric method of Lazebnik et al. (2004) uses semi-local affine parts each of which consists of spatially nearby features. In their method, candidate parts are extracted from pairs of images, and another validation set is used to select the best hypothesis to represent the object. In the object detection method of Lowe (1999, 2004) each detected SIFT feature votes for the location, scale and orientation of the object, and many nearby features voting for the same scale and orientation indicates the presence of the object. Fergus et al. (2006) improved the constellation model by having a simpler star-shaped model for the topology of the object parts and a more efficient feature model. Histogram features were used in the part based method of (Chum and Zisserman, 2007), and Crandall et al. (2005) consider different graphical models and spatial priors in their part based model.

The part based methods, described above, do not give precise information about the object location. However, there are some methods that aim to segment the target in addition to detecting or classifying the image content. Pantofaru et al. (2006) combine shape based and appearance based descriptors into something called Region-based Context Feature with which the learned object can be segmented and detected in the test image. Ahuja and Todorovic (2007) represent images by multiscale segmentation trees which are clustered to model different object categories. Object segmentation and classification are considered as two closely collaborating, intertwined processes by (Leibe and Schiele, 2003; Leibe et al., 2004, 2008) who combine local appearances and a shape model in their probabilistic framework. Winn and Jojic (2005) combine color and edge information with the shape and pose information and learn the object model by assuming a small within-instance variability and large between-instances variability on the appearance. Borenstein et al. (2004) combine bottom-up and top-down information into a single segmentation framework.

The so-called *bag-of-features* methods (Willamowski et al., 2004; Zhang et al., 2007; Opelt et al., 2006a) use a collection of features which discard the spatial layout. They are hence used only for object detection and classification, although, according to (Zhang et al., 2007), "it is possible to perform localization with a bag-of-keypoints representation, e.g., by incorporating a probabilistic model that can report the likelihood of an individual feature for a given image and category". The *bag-of-features* methods may classify images using also the information of the background; for instance, in a car representation the features might also include regions extracted from the road as they give further hint of the presence of the car in the image as opposed to general background images. Like the histogram methods, the *bag-of-features* methods have good invariance against occlusion and they often outperform the *parts-and-shape* models in pure classification tasks.

Some other methods that are difficult to categorize but should be mentioned here are the complex biologically inspired hierarchical feature model by (Serre et al., 2007), the optic flow algorithm that aims at locating the disparity between two (typically stereo) images by (Koeser et al., 2006), the method of Wolf and Martin (2005) which learns the object model from a few images with regularization by using corrupted copies of the training data, and the method of Miao and Rao (2007) who have taken a different approach in that they use Lie groups to learn the transformations between images.

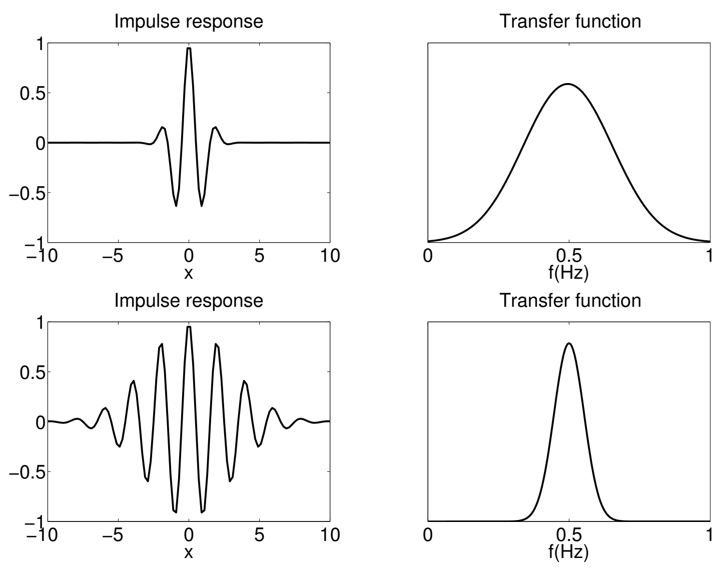


Figure 2.1: Two examples of impulse responses (real part) and transfer functions of one-dimensional Gabor filters. The frequencies are 0.5 Hz on both cases. The standard deviation of the Gaussian part of the filter in the upper row is $\sigma = 1$; in the lower row it is $\sigma = 3$.

2.5 Gabor features

The features that are employed in the studied object matching system are Gabor features, of which this section gives an overview. Gabor features have especially succeeded in face recognition (for a review, see (Shen and Bai, 2006)). As the main contribution of the thesis is on the probabilistic formulations, an exhaustive survey will not be given. The emphasis is on the practicalities instead of theoretical issues. Those interested on the theoretical background of Gabor filtering are referred to, for instance, (Lee, 1996).

2.5.1 Gabor filters

A filter can be described as something that changes a signal when applied to it. The characteristics of the filter imply how the signal changes. For instance, a low-pass filter attenuates the high frequencies of the signal so that the filtered signal contains mainly low frequencies. The filtering, that is, the process of the filter acting on the signal, can be done in two different ways, according to the convolution theorem. Either the signal is convolved with the impulse response of the filter on the domain of the signal (be it time or spatial). Another option is to multiply the Fourier transformed signal with the transfer function of the filter, which is the Fourier transform of the impulse response of the filter, in the frequency domain and inverse transform the result back to the original domain. The filtered signal is called the response of the filter.

The Gabor filters are band pass filters that have certain optimality properties, described in Subsection 2.5.2. The impulse response of a Gabor filter is a product of an infinite complex sinusoidal function and a real Gaussian function. The sinusoidal part is often called the carrier as it carries the frequency (and orientation in 2D) information, and the Gaussian part is called the envelope since it controls the overall shape of the filter. Two filters with different standard deviations in the Gaussian part are illustrated in Figure 2.1.

Because the signals here — the images — are two-dimensional, the employed Gabor filters are also two-dimensional. The response of the ordinary 2D Gabor filter depends on the average grayscale level in the images. This is unwanted because one of the invariances that a computer vision system should have is the invariance against brightness. Fortunately, this invariance can be achieved by removing the DC component of the filter (the abbreviation stems from direct current). The resulting filter is thus DC-free, meaning that the mean value of its impulse response is zero and at zero frequency its transfer function is zero.

The impulse response of the DC-free Gabor filter is

$$
\begin{aligned}
h(u,v) = \frac{f^2}{2\pi\sigma^2} \exp\left(-\frac{f^2}{\sigma^2}(u^2+v^2)\right) \times \\
\left(\exp(i(f\cos(\theta)u + f\sin(\theta)v)) - \exp(-\frac{\sigma^2}{2})\right), \qquad (2.1)
\end{aligned}
$$

where u and v are the horizontal and vertical spatial components, f is the frequency of the filter, θ is the orientation of the filter, and σ is the standard deviations of the Gaussian function (σ can differ in u and v directions). The transfer function of the DC-free Gabor filter is the Fourier transform of 2.1:

$$
\begin{aligned}
H(\omega_u,\omega_v) =& \exp\left(-\frac{\sigma^2}{2f^2}[(\omega_u - f\cos(\theta))^2 + (\omega_v - f\sin(\theta))^2]\right) \\
&- \exp\left(-\frac{\sigma^2}{2}(1+\frac{\omega_u^2}{f^2}+\frac{\omega_v^2}{f^2})\right) \qquad (2.2)
\end{aligned}
$$

from which it can be verified that $H(0,0) = 0$.

Figure 2.2 gives an example of two different Gabor filters and their responses. In the filtered images, the lines whose orientation matches the orientation of the filters are clearly emphasized. Also, the wavelength of the filter affects what lines are especially amplified: for the vertical filter, the bars on the traffic sign and the spaces between them happen to fit the wavelength, whereas the skew filter with higher frequency emphasizes finer details in the image. The phases are related to the nature of the edges; if the change (from left to right) in brightness is from low to high, the phase response is $+\pi/2$ on the edge, and vice versa.

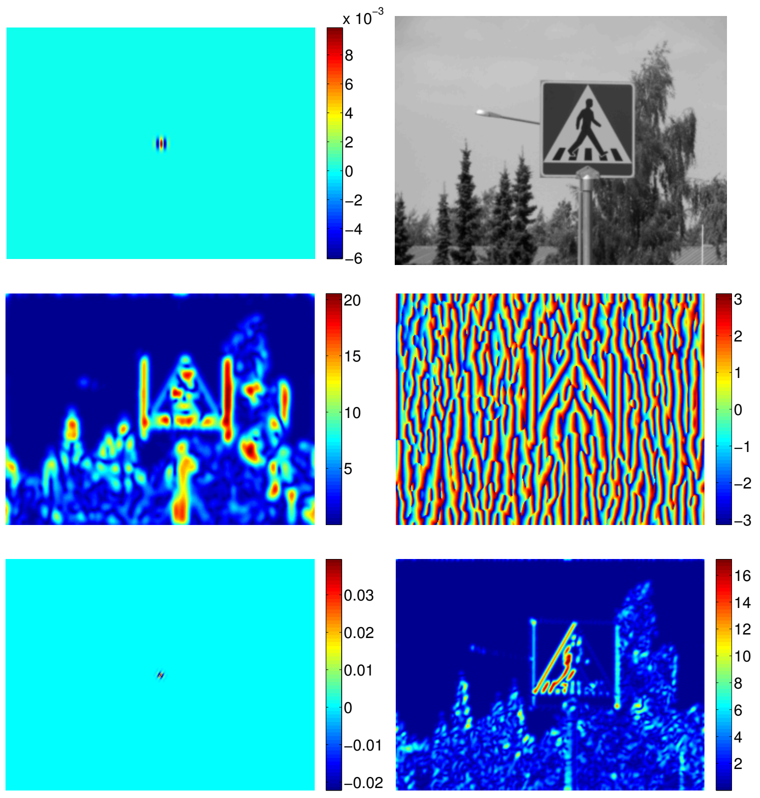


Figure 2.2: Top left: the impulse response of a 2D Gabor filter with parameters $\{f = 0.75, \theta = 0, \sigma = \pi\}$. Top right: an example image. The size of the filter equals the size of the image. Middle left: the amplitude of the filter response. Middle right: the phase of the filter response. Bottom left: the impulse response of a 2D Gabor filter with parameter $\{f = 1.5, \theta = 30°, \sigma = \pi\}$. Bottom right: the correspondent amplitude of the filtered image.

2.5.2 Properties of the Gabor filter

It can be thought that a Gabor filter acting on certain image pixel is sensitive to, or 'listens' to, a signal whose shape corresponds to the filter parameters. The standard deviation of the Gaussian part controls the extent of the image area which is being listened to. Hence, the Gabor filter acts as a local edge-like feature detector, as Figure 2.2 also demonstrates. The amplitude part of the filter response gives approximate information about the locations of the lines which is further specified by the phase response.

Figure 2.3 demonstrates the influence of the Gaussian part of the Gabor filter. Two vertical filters with different values on the standard deviations, $\sigma = \pi$ and $\sigma = 3\pi$, are applied on a binary image. Middle panel reveals how the sharper filter obtains positive responses only in the very vicinity of the edge whereas with the broader filter the edge information spreads into a wider range. Due to the

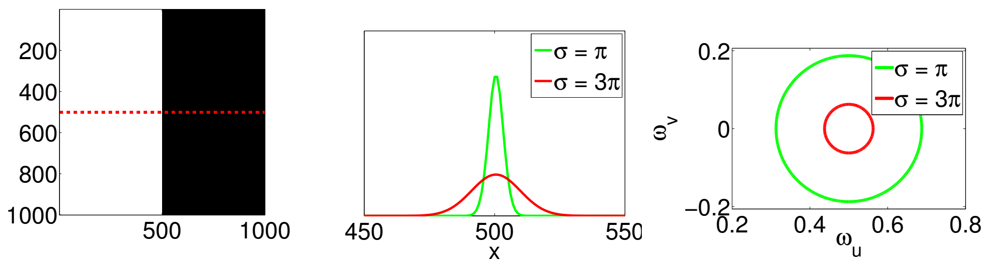


Figure 2.3: Left panel shows a binary image which is filtered with two different Gabor filters having different values for σ. Frequency and orientation parameters were set to $f = 0.5, \theta = 0$. The responses of both filters are computed along the red dashed line. Middle panel shows the amplitudes of the responses as function of the position along the vertical line (note the limited x axis). Right panel shows the contour lines along which the values of the transfer functions in the frequency domain are 0.5.

normalization factor in the Gaussian part, the response on the edge is lower with the broader filter. Thus, it seems to make sense to use filters that are tight in the spatial domain. For instance, when using the standard deviation $\sigma = \pi$, the filter 'sees' the edge in the area of less than ten pixels in both directions which should yield good matching accuracy, at least when used with the phase information.

However, the coin has another side as well. The right panel of Figure 2.3 reveals how the transfer function of the filter that is sharper in the spatial domain is broader in the frequency domain. The same phenomenon can be seen in Figure 2.1 where the one-dimensional filter with larger extent in the impulse response results in a tighter transfer function, or a smaller width of its band pass. This is not a property of Gabor filters alone but is a general principle in the filtering theory, known as the uncertainty principle. It states that the higher the precision in the spatial domain, the lower it is in the frequency domain. In other words, if one wants to measure the frequency of a signal with infinite accuracy, it comes with the cost of losing the spatial (or temporal if the signal is in the time domain) knowledge. On the contrary, a delta function can be perfectly localized but nothing can be said about its frequency contents. These extremes correspond to having the Gaussian width parameter at $\sigma = \infty$ and $\sigma = 0$, respectively.

It can be shown that there is a lower bound for the joint uncertainty of a filter in the two domains:

$$\Delta u \Delta v \Delta \omega_u \Delta \omega_v \geq \frac{1}{16\pi^2}\,, \tag{2.3}$$

where $(\Delta u, \Delta v)$ are the normalized second moments of the spatial variables and $(\Delta \omega_u, \Delta \omega_v)$ those of the frequency variables. The uncertainty principle is also familiar from quantum mechanics in which it states that the product of the uncertainties of any two observables with non-commutative operators has a lower bound. For instance, both the position and momentum of a particle cannot be known with arbitrary accuracy. It is not a coincidence that the same phenomenon occurs in these two different fields of science. In quantum mechanics, particles are represented by a wave function whose magnitude can be inferred as a probability distribution of the location of the particle, and the moments of an observable are computed by integrating the corresponding operator over the square of the wave function (Ballentine, 1998). Because the position and momentum operators are conjugate operators, that is, the Fourier transforms of each other, the situation is mathematically very similar as in filtering theory.

Despite the joint uncertainty in the two conjugate domains being lower bounded by equation (2.3), it is possible to derive a filter that achieves this lower bound. This filter is the Gabor filter (Gabor, 1946; Daugman, 1985, 1988). Applying Gabor filter yields thus an optimal extraction of the spatial and frequency information from the signal. Also in quantum mechanics the wave function that satisfies the minimum uncertainty relation is of the same form as the Gabor impulse response. The widths of the Gabor filter in spatial and frequency domain are

$$\Delta u = \Delta v = \frac{\sigma}{\sqrt{2}f} \ , \qquad \Delta\omega_u = \Delta\omega_v = \frac{f}{2\sqrt{2}\pi\sigma} \ , \tag{2.4}$$

from which it can be verified that the Gabor filter indeed achieves the lowest possible joint uncertainty. Apart from the information theoretical reasons, there are also psychophysical arguments that favor Gabor filters: the receptive fields in the mammalian primary visual cortex have been reported to resemble the impulse responses of the Gabor filter (Daugman, 1988; Palmer, 1999). This indicates that the cells in the cortex act as local frequency analyzers for the incoming images. Finally, it should be noted that the Gabor filters do not form an orthogonal basis, that is, an inner product of the impulse responses of the different Gabor filters is non-zero. At first sight, this sounds like bad news as orthogonal basis functions are typically preferred in constructing signals due to their convenience. However, Daugman (1988) showed that the Gabor filters, while non-orthogonal, are optimal frequency / spatial analyzers. In quantum mechanics, the consequence of having a non-orthogonal set of wave functions is that the state of the particle is pure, that is, a state with just single non-zero term in its spectral representation (Ballentine, 1998).

2.5.3 Gabor filter bank

The equations (2.1) and (2.2) define a filter with certain frequency and orientation. Filters with different frequencies and orientations constitute a filter bank. If an image is filtered with a filter bank, each filter of the bank is applied in the image, one at the time. The result is an $N_u \times N_v \times N_f \times N_\theta$ -sized (complex valued) matrix, where N_u and N_v are the dimensions of the image, N_f is the number of different frequencies in the filter bank, and N_θ is the number of different orientations. An example of a filter bank is given in the spatial domain in figure 2.4 and in the frequency domain in figure 2.5. The area that the filters cover in the frequency

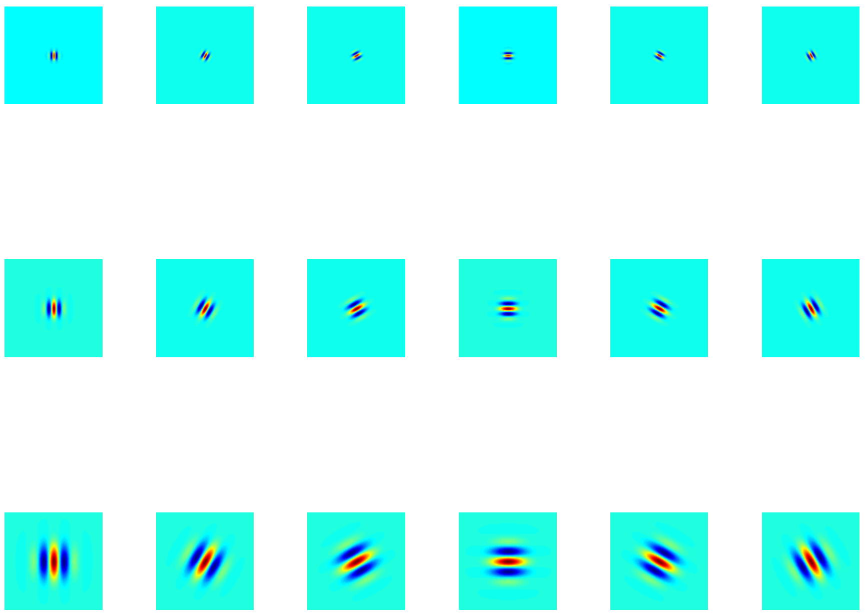


Figure 2.4: The real parts of the impulse responses of a Gabor filter bank with three frequencies $f = \sqrt{2}\pi/\{4, 8, 16\}$ and six orientations $\theta = \{0, \pi/6, ..., 5\pi/6\}$. The Gaussian standard deviation is $\sigma = \pi$. Red color indicates high values.

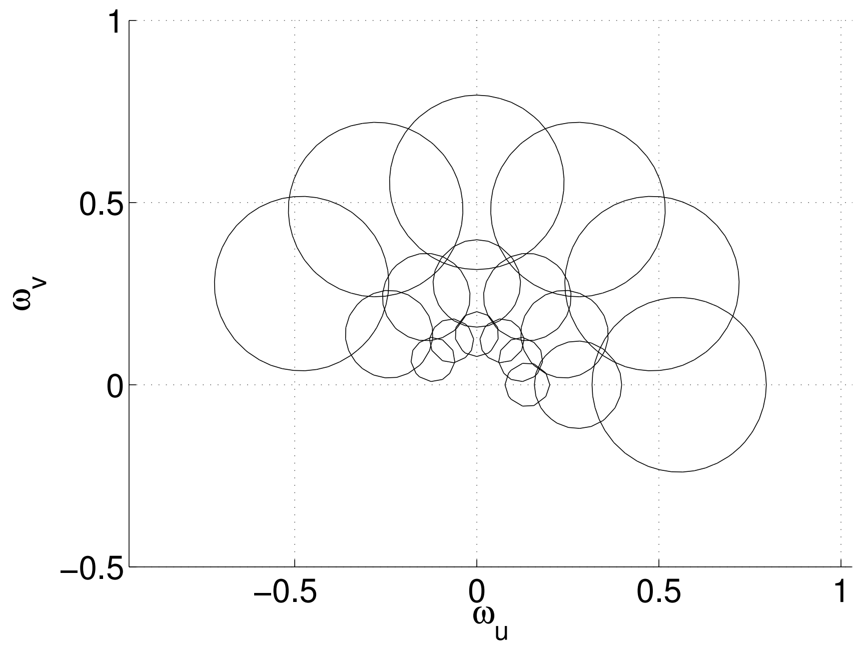


Figure 2.5: The transfer functions of the filter bank of Figure 2.4. The value along the contours is 0.4.

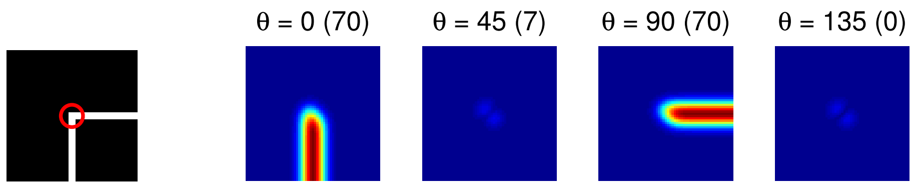


Figure 2.6: The amplitudes of Gabor filter bank responses of a binary image, shown on the left panel. The filter bank contains four differently oriented filters: $\theta = \{0, 45, 90, 135\}^\circ$. The orientation angle is shown above each panel, as well as in parentheses the amplitude value (multiplied by 1000 for clarity) at the corner point, which is marked with a red circle in the binary image.

space is controlled by the number of the filters in the bank and the value of σ, which is related to a band pass of a filter through equation (2.4). Note that only the upper half of the frequency space needs to be covered as the orientation of any line is between zero and 180 degrees.

Each filter in the bank detects lines that correspond to its frequency and orientation. Applying a filter bank makes it possible to detect more complex features than just lines. For example, the filter bank responses of a corner are depicted in figure 2.6. The absolute values of the responses at the marked point indicate that the feature consists of vertical and horizontal lines, but not diagonal lines, which suggests the feature to be a right-angle. Hence, corner points can be represented by a vector whose elements indicate the responses of differently oriented Gabor filters. The problem of distinguishing a corner and an intersection of lines with the filter bank is discussed in the next subsection which introduces a similarity measure between two Gabor filter bank responses.

2.5.4 Gabor similarities

Having introduced the Gabor filter banks in the previous subsection, a computational tool which compares the filter bank responses at two different image points should be derived. Such a tool would enable points that are visually similar with a given pixel to be located in another image. The vector of the responses of a filter bank at certain pixel is called a filter jet. Each pixel can thus be transformed into the Gabor jet space which is $N_f \times N_\theta \times 2$ dimensional, as the jets are complex, and a visual similarity between two pixels can be assessed by comparing their vectors in the jet space. A natural similarity measure between two vectors is their normalized inner product which reveals the angle between the vectors. The inner product of the amplitudes of two jets J and J', first proposed as a similarity by Lades et al. (1993), is defined as

$$S_a(J, J') = \frac{\sum_i a_i a'_i}{\sqrt{\sum_i a_i^2 \sum_i a'^2_i}} \in [0, 1] \, , \tag{2.5}$$

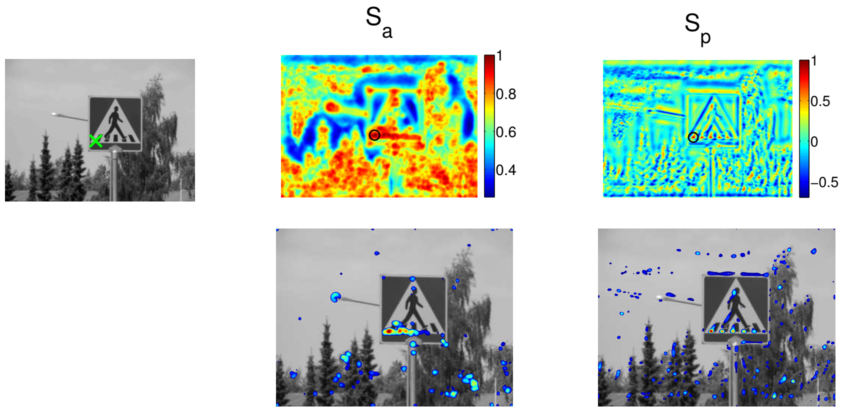


Figure 2.7: Two different similarity fields between the point, shown as green cross in the leftmost image, and the image. S_a and S_p refer to equations (2.5) and (2.6). In the bottom row, the contours show the top 3% similarity values.

where a_i and a'_i are the amplitudes of the components of the two jets, the number of which is $N_f \times N_\theta$. Another option would be to compute the sum of squared differences of the components which for normalized vectors is, as an optimized quantity, equivalent to their inner product.

The similarity measure (2.5) ignores the phase of the filter responses which carries additional information of the images. The phase can be included by computing the inner product of the complex jets and taking the real part of the result. In terms of amplitudes a_i and phases ϕ_i, the inner product is computed as

$$S_p(J, J') = \frac{\sum_i a_i a'_i \cos(\phi_i - \phi'_i)}{\sqrt{\sum_i a_i^2 \sum_i a'^2_i}} \in [-1, 1] \,. \tag{2.6}$$

Wiskott et al. (1997) modify the similarity measure (2.6) by adding a phase displacement into the argument of the cos function. The purpose of this displacement term is to compensate for the rapid spatial variance of the phase by minimizing the phase difference. As a result, the widths of the local similarity peaks in the image are increased which facilitates the optimization of the location, but the similarity fields also tend to be less smooth and contain discontinuities (Kalliomäki, 2007). Therefore, the complicated estimation of the displacement term is discarded and the similarity measure (2.6) is used as such.

An illustration of the two similarity measures is given in Figure 2.7, where the similarities are computed between an image and a point that is extracted from the same image. Naturally, the similarities are equal to one at the comparison point. However, multiple local maxima can be seen elsewhere in the image, which complicate the object matching. The local peaks in the phase-ignoring similarity S_a are broader than in the phase-sensitive similarity S_p which produces multiple sharp maxima in the image. Having the filter jets normalized makes the similarities contrast invariant, which causes local maxima also in the image area which seems

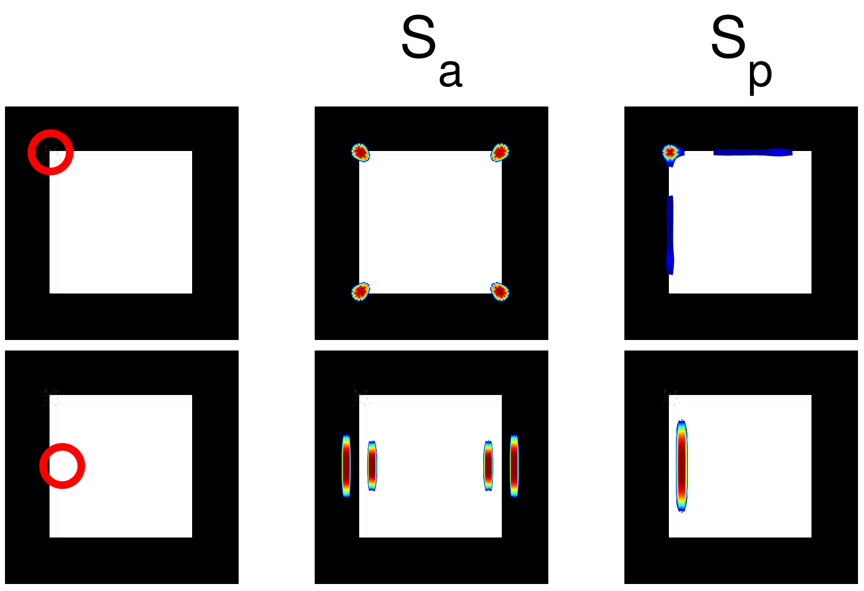


Figure 2.8: The similarity fields of the points, shown with red circles in the leftmost panels, within the image. S_a and S_p refer to equations (2.5) and (2.6). The contours show only the highest 1% of the values.

to be uniform. The similarity fields in Figure 2.7, as well as in the upcoming examples, are formed with a Gabor filter bank that was presented in Figures 2.4 and 2.5.

The properties of the two similarity measures become more apparent in Figure 2.8 which shows the similarities for a simple binary image. With phase-sensitive similarity S_p, also the phases must match, that is, the signals must be in the same phase at the pixels being compared. The phase at a transition from low grayscale values to high values is opposite to a reversed transition and the similarity S_p possesses high values only when the signal changes in the same direction; the similarity S_a ignores the direction as it utilizes only the amplitude information. In addition, the Gabor filter bank only registers if there are lines whose alignment corresponds to the orientations in the filter bank; the mutual relations between the lines are ignored. Hence, a pixel from which a line goes down and another line goes right is Gabor-wise the same pixel as the one from which a line goes up and another line goes left (or any other combination of these). As a result, each corner of the square is equally similar with S_a whereas the similarity S_p is high only in the corner where the changes from white to black are in the correct order. The same thing happens when the comparison point is near the edge. Although it seems that S_p is a better measure of visual appearance, problems occur when the features are on the edge of the object as the background may reverse the direction of the signal. The similarity S_a is invariant to such reversions. However, for the features that are inside the object the similarity S_p has been found to outperform S_a as it is more

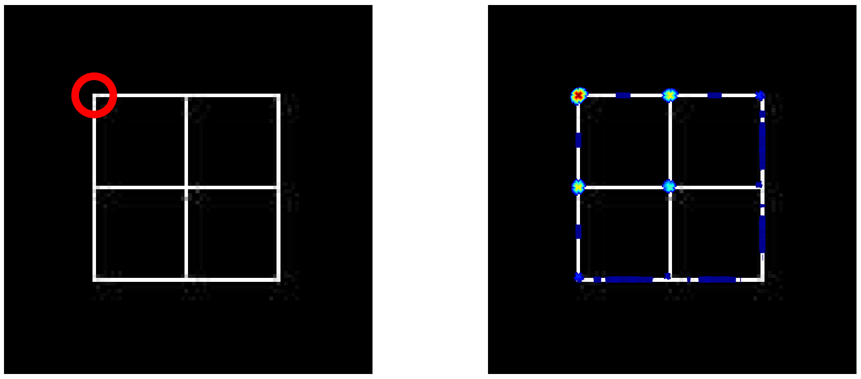


Figure 2.9: The phase-sensitive similarity field (2.6) of the point, shown with a red circle in the left panel, within the image. The contours show only the highest 1% of the values.

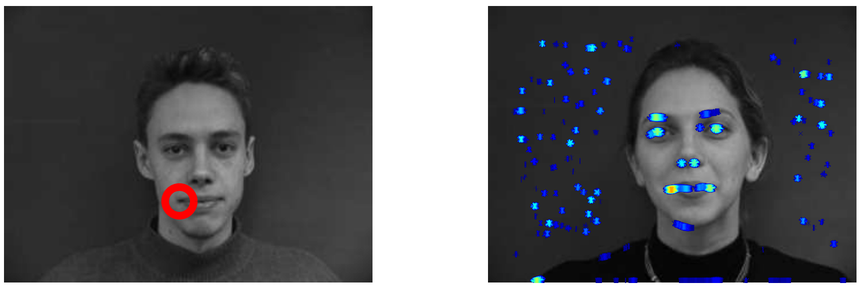


Figure 2.10: The phase-sensitive similarity field (2.6) between the point, shown with a red circle in the left panel, and another image. The contours show only the highest 1% of the values.

discriminant due to the phase information, and the object matching methods of this thesis mainly use the phase-sensitive similarity S_p.

Figure 2.9 gives an example of how also the phase-sensitive similarity has difficulties in separating a corner point from an intersection of two lines that are aligned similarly than the lines that form the corner. Although the similarities at the confusing points are clearly less than unity, such false local maxima complicate the matching. Finally, Figure 2.10 shows a real example how well the Gabor similarity generalizes into another object instance. Again, there are multiple fallacious similarity peaks in the image. This time it happens that the correct location is the global maximum of the similarity, the value of which is 0.88.

2.5.5 Rotation and scale invariance

The Gabor similarities decrease when the scale and rotation differences between the object instances increase. This phenomenon is illustrated in Figure 2.11 where a dog image is rotated in the upper row and rescaled in the bottom row. The more

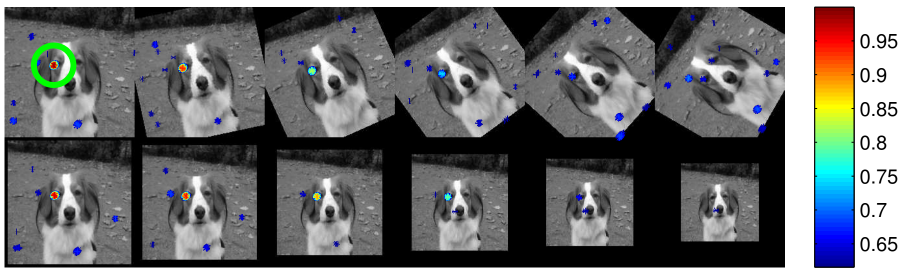


Figure 2.11: The phase-sensitive similarity field (2.6) of the point, shown with a green circle, within the image. The contours show only the highest 0.1% of the values.

the images are rotated and scaled, the lower is the similarity. Rotation and scale invariances are — among the almost inherent translation invariance — characteristics of a proper object matching system. If the orientations and the logarithms of the frequencies are uniformly distributed in the filter bank, it is possible to obtain some level of invariance by shifting the filters in the jet J' (see equations (2.5) and (2.6)) so that all the center frequencies and orientations of the filters are shifted by the same amount, and using the shift which most closely corresponds to the scale and rotation. For instance, if the orientation angles in the filter bank are $\theta_k = k(\Delta\theta), k = 1, 2, ..., N$, and if the rotation was exactly the same as the orientation differences in the filter bank $(\Delta\theta)$, the similarity could be computed precisely by using filters whose orientations are shifted by one index. That is, if the reference jet consists of uniformly spaced orientations $\theta_1, ..., \theta_N$, the image in which the similarity is to be computed would be filtered with orientations $\theta_2, ..., \theta_{N+1}$, where the filter with θ_{N+1} is the complex conjugate of the filter with θ_1. Likewise, for a rotation angle θ_2 the filters with orientations $\theta_3, ..., \theta_{N+2}$ would obtain an exact similarity, and so on. For a rotation angle that falls between the orientation differences, the nearest shift would have to be used.

If the differences between the adjacent orientations and frequencies in the filter bank are small, the shifting approach performs well. This, however, leads to a large filter bank which increases the computation times. It would be nice to have a sparse filter bank while also being able to compute the similarities for an arbitrary rotation and scale. This is possible — at least approximately — by computing the filter responses as a linear combination of the responses of the filter bank. For instance, the impulse response for rotation φ would be (Kalliomäki, 2007)

$$h(u, v; \varphi) = \sum_{i=1}^{N_\theta} w_i(\varphi) h(u, v; \theta_i) , \tag{2.7}$$

where $w_i(\varphi)$ are the weights that depend (only) on the rotation angle. Interpolating the filters according to (2.7) is called *steering*. Kalliomäki (2007) derived

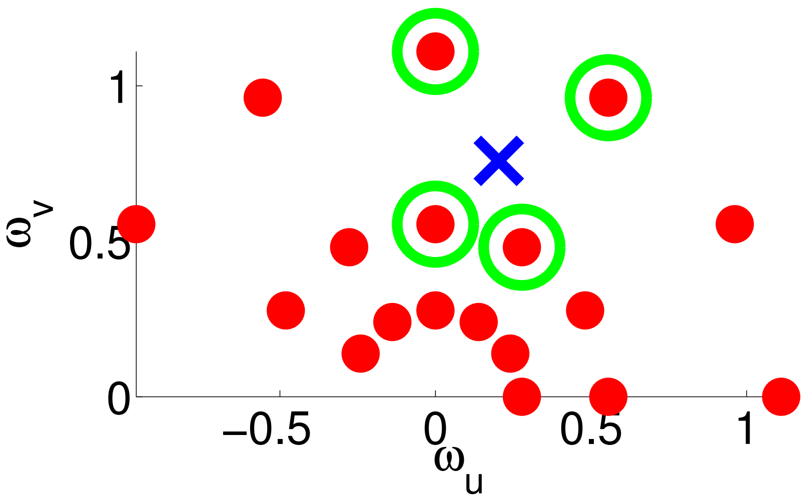


Figure 2.12: An illustration of interpolating the filter responses. The filter response at the blue cross can be estimated from the four neighboring filters that are marked with green circles.

analytical expressions for the coefficients w_i so that the square error between the exact response and the linear combination is minimized. In this thesis, a simpler approach is taken: Giving a scale and rotation angle, the filter responses are computed from the four neighboring filter responses (see Figure 2.12). The weights are interpolated linearly for the rotation and logarithmically for the scale (Günther and Würtz, 2009). In addition, the weights are the same for each new filter so the whole new filter bank can be computed from the existing filter bank as a weighted sum. For instance, let us consider a rotation angle $\theta_3 + (\Delta\theta)/2$, where $\Delta\theta$ is

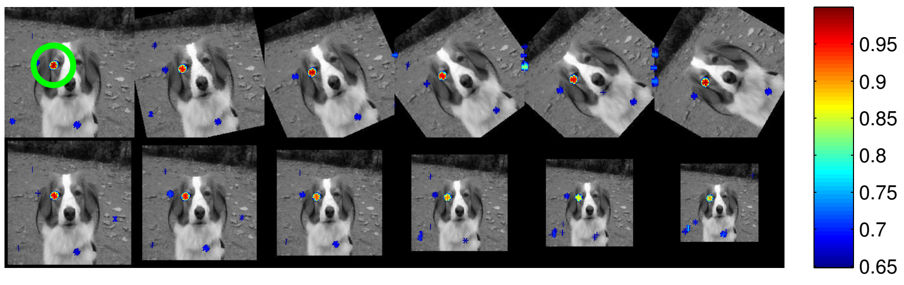


Figure 2.13: An illustration of applying the interpolation scheme into the transformed dog image. The contours show the highest 0.1% of the phase-sensitive similarity values (2.6) of the point, shown with a green circle, within the image. Compare with Figure 2.11.

the gap between two orientations in the filter bank. The weights in equation (2.7) would be $w_3 = 0.5$, $w_4 = 0.5$, and other weights zero, so the filter bank would be first shifted two units (to reach the gap between θ_3 and θ_4) and then interpolated using θ_3 and θ_4. The filter bank must be extended to be able to estimate the responses. The responses of the lower half plane are complex conjugated from the upper half, and for the scales, extra frequencies must be added to both ends. An example of the interpolation approach is given in Figure 2.13 which shows a clear improvement over the Figure 2.11; the similarities are high at the correct location. Naturally, the scale and rotation can occur at the same time. The rotation could be estimated by optimizing it in each pixel so that the similarity is maximized. Instead of rotating and scaling each feature independently, in this thesis all the features are transformed with the same scale and rotation that is related to the object instance.

Chapter 3

Bayesian methodology

3.1 Introduction

Thomas Bayes, a reverend and mathematician, has given his name to the statistical approach called Bayesian inference. What makes Bayesian inference attractive is its consistent handling of uncertainties which are always inherent in real data. Bayesian treatment often leads to complex models, needing heavy computer algorithms to solve them and the recent interest in Bayesian modeling can be partially explained with the increase of computational power on personal computers. Bayesian inference is based on subjective prior probability, which has been criticized. However, in science there usually is some inter-subjective prior knowledge about the process being measured that can be explicitly incorporated in Bayesian methodology (Bolstad, 2004; Spiegelhalter et al., 2004). On the other hand, in the absence of prior beliefs, non-informative prior probabilities can always be used. In Bayesian inference, the model parameters are considered as random variables with probability distributions and the inference is based on the actual observed data. This contrasts with the classical frequentist approach — another statistical viewpoint — where the model parameters are considered as fixed but unknown constants and the inference is based on all the possible data sets that might have occurred but did not. Interpreting the uncertainty about the true value of the model parameters is natural in Bayesian approach (Bernardo and Smith, 1994). In Bayesian methodology it is also straightforward to handle non-interesting parameters, to find predictive distributions and to build hierarchical models.

This chapter presents the basic idea behind Bayesian inference, as well as the sampling methods used in analyzing the complex distributions. For further information in Bayesian modeling, the interested reader is referred to (Bolstad, 2004), (Gelman et al., 2004) and (Bernardo and Smith, 1994).

3.2 Bayesian inference

3.2.1 Bayes' theorem

Bayes' theorem is the cornerstone of Bayesian inference. Basically, it is an outcome of a simple use of the basic laws of probabilities. For (any) events D and θ it can be written that $p(D, \theta) = p(D|\theta)p(\theta)$. On the other hand, it also holds that $p(D, \theta) = p(\theta|D)p(D)$. This yields

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} \; . \tag{3.1}$$

This formula is known as Bayes' theorem when we let D denote the data and θ the (unknown) parameter or parameter vector of the model (there may be one or more parameters). With such a notation, $p(\theta|D)$ is the posterior probability of the model parameters given the data, $p(D|\theta)$ is the likelihood of the parameters, $p(\theta)$ is the prior probability of the parameters and $p(D)$ is a normalization constant, also called the evidence of the model. This normalization constant is the integral of the numerator over the model parameters:

$$p(D) = \int p(D|\theta)p(\theta)d\theta \; . \tag{3.2}$$

Bayes' theorem thus says that the prior belief of the parameters turns into posterior probability after observing the data by multiplying the prior probability with the likelihood and normalizing. This is exactly what intuition would also say; before experiments, we may have some belief about the parameter values, and the observations modify the belief. The posterior distribution is often computed only in unnormalized form, $p(\theta|D) \propto p(D|\theta)p(\theta)$, as it is usually enough to compare the relative probabilities for different values of θ, without needing to know the absolute values. Here, the probability distributions are taken to implicitly depend on the used model assumptions. When comparing different models, which is again straightforward in the Bayesian approach, model dependencies become explicit.

By using a uniform prior distribution for θ, $p(\theta) \propto 1$, it holds that $p(\theta|D) \propto p(D|\theta)$. Thus, without any prior beliefs the posterior distribution equals (up to a normalization constant) the likelihood function that is used in the frequentist approach to make inferences about the parameters. However, there is a philosophical difference in interpreting the probabilities. Whereas $p(D|\theta)$ is the probability of the data, given some parameter value, the posterior distribution is the probability distribution of the unknown parameter, given the data. The posterior distribution is the most important tool needed in Bayesian inference. With the posterior distribution, it is easy to compute, for instance, the probability that the true value of the unknown parameter is in some range. Frequentists assess this with confidence intervals, which are based on the idea of having an infinite number of hypothetical repetitions of an experiment (Bernardo and Smith, 1994). Interpreting the probabilities of the possible values of the parameters is thus more natural in the Bayesian

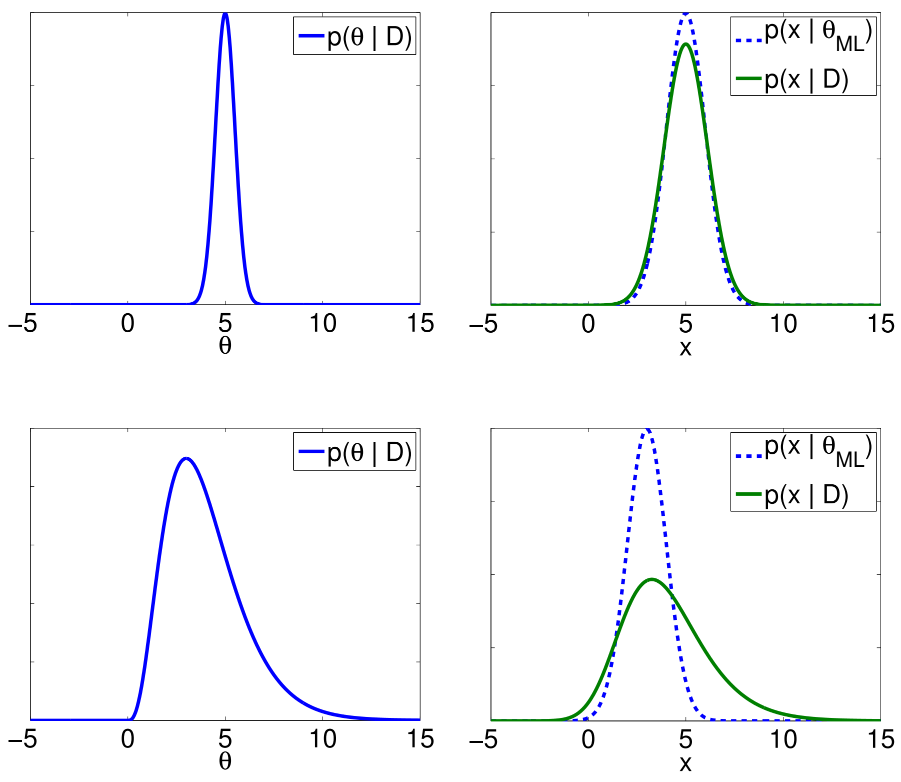


Figure 3.1: The panels on the left show two different artificial posterior distributions of θ, and on the right are the corresponding distributions $p(x|D)$, computed using the ML method and the Bayesian method. The distribution $p(x|\theta)$ is taken to be a Gaussian distribution, with the parameter θ as mean value and with unit variance.

treatment. Furthermore, to make decisions that are based on the model parameters, Bayesian inference can directly apply the decision theory whereas in the frequentist approach the properties of the used estimators must be considered (Bernardo and Smith, 1994).

3.2.2 Applying Bayesian inference

Applying Bayesian inference is more than just using Bayes' theorem. When making inferences that are based on the model parameters, the Bayesian framework offers a natural way to take into account the uncertainty involved in the parameters, namely marginalization. Consider a variable x that depends on the model parameters θ. For instance, x might be future observations. We are interested in the probability distribution of x, given the observed data, $p(x|D)$. The Bayesian way to compute the distribution is by integrating over θ, that is, by marginalizing the joint distribution:

$$p(x|D) = \int p(x,\theta|D)d\theta = \int p(x|\theta,D)p(\theta|D)d\theta = \frac{\int p(x|\theta)p(D|\theta)p(\theta)d\theta}{\int p(D|\theta)p(\theta)d\theta} \; , \tag{3.3}$$

where x has been assumed independent of D, when θ is given.

The frequentist approach would be to compute a point estimate for the model parameter and plug that into the equation. A widely used point estimate for the parameter is the one that maximizes the likelihood, called the maximum likelihood (ML) estimate: $\theta_{ML} = \arg\max_\theta p(D|\theta)$. The probability in (3.3) then reduces to $p(x|D) \approx p(x|\theta_{ML})$. This corresponds to having a posterior distribution centered at θ_{ML} and having zero variance, i.e. a delta function at θ_{ML}. According to the large sample theory, with lots of observed data this assumption approaches the reality and the approximation holds. If the variance of the posterior distribution is large, using the point estimate alone is prone to leading to poor results and confidence intervals which take into account the variance should be used. A multimodal likelihood is especially difficult to handle with frequentist methods, but then again, multimodality causes problems also in Bayesian inference, at least in high dimensions. To illustrate the differences of Bayesian and ML approaches, Figure 3.1 shows two artificial posterior distributions of θ and the distributions $p(x|D)$, computed using the ML estimate and Bayesian methods. The prior distributions have been assumed uniform so that the maximum of the likelihood is the same as the maximum of the posterior. In the upper panels, the variance of the posterior is fairly low and the two approaches result in similar distributions. In the lower panels, the posterior is skewed and the ML approach gives too optimistic results as the uncertainty in x is underestimated. On the contrary, computing $p(x|D)$ with equation (3.3) takes into account the whole posterior distribution of θ.

Another advantage of the Bayesian approach, worth mentioning here, is the possibility to build hierarchical models. This is useful, as the structure in real problems often consists of multiple levels of hierarchies. In hierarchical models, the parameters, on which the data depends, have prior distributions whose parameters are not fixed but have their own prior distributions. Such parameters are called hyperparameters and their priors are called hyperpriors. It is thus possible to pass the selection of some parameter values further up in the hierarchy, 'away from the data' (Gelman et al., 2004). Let us denote with φ the hyperparameters. Applying Bayes' theorem and the rules of probabilities, the joint posterior distribution is

$$p(\varphi,\theta|D) \propto p(D|\theta,\varphi)p(\varphi,\theta) = p(D|\theta)p(\theta|\varphi)p(\varphi) \; . \tag{3.4}$$

Note that in the likelihood the data is independent of the hyperparameters. The marginal posterior distribution is again obtained by marginalizing:

$$p(\theta|D) = \int p(\varphi,\theta|D)d\varphi \; . \tag{3.5}$$

Hierarchical modeling is especially useful when the data is divided into groups, each of which have their own parameters with common hyperparameters.

A fully Bayesian analysis usually refers to a hierarchical model where the non-interesting parameters are integrated out. If point estimates are used (for hyper-parameters), the approach is usually called empirical Bayes. The empirical Bayes analysis is computationally lighter than the full treatment but with few data may be inferior as all the uncertainties in the model are not integrated out.

3.2.3 Computational difficulties in using Bayesian inference

There are few distributions, such as Gaussian, for which the posterior distribution can be analyzed in closed form. In such cases one can compute, for instance, the mean and variance of the model parameters or the integral in equation (3.3) with pen and paper. Typically, however, computational algorithms are needed to analyze the posterior distribution. If the parameter space is of low dimension, that is, if the parameter vector θ has few components, it usually suffices to divide the parameter space into a dense grid and evaluate the value of the posterior at each grid point, or to use the traditional quadrature methods, such as Riemann quadrature. With an increasing number of parameters in the model, these methods become rapidly unfeasible due to a phenomenon called the curse of dimensionality: the number of grid points needed for accurate analysis grows exponentially with the dimension of the parameter space. Even a model with as few as five parameters may be too complex for the grid method; assuming we wish to divide each component of the parameter vector into one hundred elements, the matrix containing the posterior values will have $100^5 = 10^{10}$ elements. In the object matching problems studied in this thesis, there may be one hundred parameters, which take approximately 200 different values (depending on the number of pixels in the images). The number of elements in the posterior matrix would thus be $200^{100} \approx 10^{230}$, which greatly exceeds the number of elementary particles in the universe! Alternative numerical methods are thus needed. The two most widely used approaches are to approximate the posterior with simpler distributions or to sample the posterior with Monte Carlo sampling methods (Gelman et al., 2004).

There are three main methods for approximating the posterior (Bishop, 2006). In the variational Bayesian method, the posterior distribution is approximated with a more tractable distribution that minimizes the Kullback-Leibler divergence between the two distributions; in Laplace approximation, the posterior distribution is approximated with a Gaussian distribution centered at the MAP (maximum a posteriori) estimate; and in the expectation propagation algorithm, assumed-density filtering and loopy belief propagation are combined to obtain an approximation of the posterior distribution. However, the true posterior might not be easily approximated with simpler distributions. If the posterior distribution is multimodal, one has to use a mixture distribution as the approximating distribution, which requires the locations of all the modes to be found.

The Monte Carlo sampling methods, on the other hand, aim to approximate the posterior distribution with samples. These samples are concentrated in such parameter values for which the value of the posterior distribution is large. Theoretically,

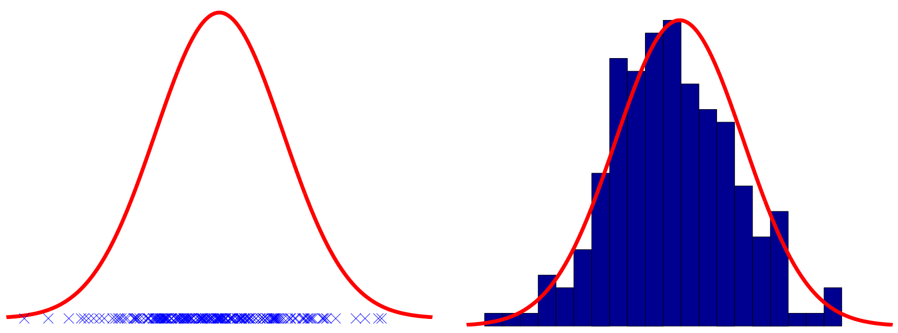


Figure 3.2: An illustrative example of Monte Carlo sampling. Left panel shows a probability distribution and 200 samples, collected from the probability distribution, as crosses. Right panel shows a histogram of the samples.

there is no limitation on the form of the posterior or the dimension of the parameter space although in practice these have a huge impact on the performance of different Monte Carlo sampling methods. In the object matching problem studied in this thesis, the posterior distribution is multimodal and difficult to approximate with simple functions. Hence, Monte Carlo methods are used, especially particle Monte Carlo methods, which generally work well also for multimodal distributions. Monte Carlo methods are dealt more extensively in the next section.

3.3 Monte Carlo methods

3.3.1 Introduction

The idea behind Monte Carlo (MC) sampling methods is to collect samples of the target distribution, i.e. the distribution one is interested in, and to represent the target distribution with these samples. Using the sample representation, any integral that is based on the target distribution can be estimated. An example of a sample representation is given in Figure 3.2. Luckily, it is adequate to collect samples of the unnormalized target distribution since the value of the normalization constant can be ignored. This makes the MC methods appealing for Bayesian modeling as the normalization constant of the posterior distribution is often difficult to compute. Let us denote with $\{x^{(i)}\}, i = 1, ..., N$ a sample set of size N that represents the target distribution $p(x)$. The mean value of a function $f(x)$ w.r.t. the distribution $p(x)$ is approximated as

$$\mathbb{E}_{p(x)}[f(x)] = \int f(x)p(x)\,dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}) \;. \tag{3.6}$$

Inserting $f(x) = x$, it holds that the Monte Carlo estimate of the mean value of the parameter x is the mean value of the samples, in accordance with intuition. By the strong law of large numbers, the MC estimate converges almost surely to the exact

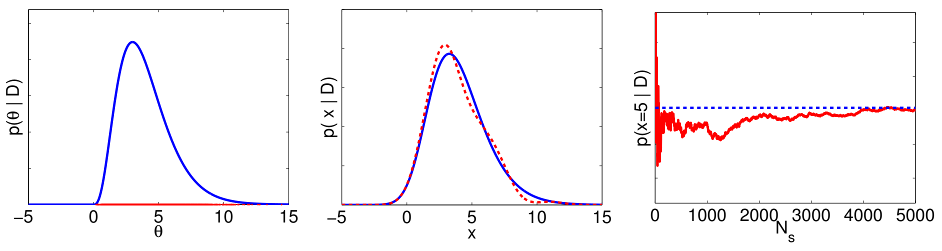


Figure 3.3: Left panel: the posterior distribution $p(\theta|D)$ of the left lower panel in Figure 3.1 and 100 samples drawn from it, shown as red dots. Middle panel: the distribution $p(x|D)$ computed with numerical trapezoidal integration method (blue solid line), and the distribution $p(x|D)$ estimated from the Monte Carlo samples (red dashed line). Right panel: A Monte Carlo estimate of $p(x|D)|_{x=5}$ for different number of samples (one realization), and the 'exact' value, computed with trapezoidal method, shown as blue dashed line.

value with an increasing number of samples, irrespective of the dimension of the parameter space (Robert and Casella, 2004; Gelman et al., 2004).

For instance, the integral of equation (3.3) can be estimated with N samples $\theta^{(i)}, i = 1, ..., N$, collected from the (normalized or unnormalized) posterior distribution $p(\theta|D)$:

$$p(x|D) = \frac{\int p(x|\theta)p(D|\theta)p(\theta)d\theta}{\int p(D|\theta)p(\theta)d\theta} \approx \frac{1}{N}\sum_{i=1}^{N} p(x|\theta^{(i)}) \,. \tag{3.7}$$

In Figure 3.3, the distribution $p(x|D)$ of the example of the lower panels in Figure 3.1 is estimated using MC sampling. First, 100 samples are obtained from the posterior distribution $p(\theta|D)$, which are then used in estimating the distribution $p(x|D)$. The result is close to the 'exact' distribution, which is computed using the deterministic trapezoidal integration method ('exact' in a sense that the trapezoidal integral is also an approximative numerical method, but in one dimension and small step size it is in practice equal to the exact solution). The rightmost panel shows how in this experiment the estimate of the value of $p(x|D)$ at $x = 5$ approaches the true value with a growing number of samples.

Basically, Monte Carlo sampling can be used to approximate any integral by suitably factorizing the integrand into a probability function, which is sampled, and another function whose expectation value is the integral in question, so that the formula (3.6) can be used. Monte Carlo integration can actually be thought of as a Riemann sum, with the samples $x^{(i)}$ randomly simulated from $p(x)$ instead of being deterministically spaced. Hence, a reasonably small number of samples is often adequate to represent the target distribution, especially if most of the probability mass is concentrated on a tiny area in the parameter space. Also, when the dimension grows, the number of MC samples needed for convergence does not necessarily scale with the complexity of the other numerical integration methods. This happens, for instance, if the probability distribution forms a subspace in the parameter space.

So far we have not considered how the samples are obtained from the target distribution. This is the most difficult part of using MC methods. The factors that complicate the collection of a fair sample representation are the number of local modes, the amount of correlation in the modes and the dimension of the parameter space. Two Monte Carlo techniques are introduced here. The first one is based on Markov chains and is among the earliest MC strategies invented. The second one is a more recent method that is based on weighted particles and in general works better with multimodal distributions.

3.3.2 Markov chain Monte Carlo methods

In Markov chain Monte Carlo (MCMC) methods, a chain of random steps in the parameter space is constructed so that it converges to the target distribution. This Markov chain evolves over time by switching its state through a transition kernel and has the target distribution as its stationary distribution. The convergence rate depends on the starting value of the chain, but theoretically, after a sufficient long time, the chain eventually 'forgets' its initial state and the samples of the chain resemble the samples of the target distribution (Gelman et al., 2004; Neal, 1993).

The differences between various MCMC methods lie in how the state of the chain is updated. The most general MCMC method is the Metropolis-Hastings algorithm, which uses accept/reject rule for the updating. In Metropolis-Hastings algorithm with $p(x)$ as the target distribution, at time t a candidate x^* is drawn from a proposal distribution $q(x|x^{t-1})$ and this proposal is accepted as a new state x^t with probability

$$\zeta = \min\left(1,\ \frac{p(x^*)}{p(x^{t-1})}\frac{q(x^{t-1}|x^*)}{q(x^*|x^{t-1})}\right)\ , \tag{3.8}$$

otherwise the state stays unchanged, $x^t = x^{t-1}$. Theoretically, under certain weak assumptions, this chain is guaranteed to converge to the target distribution (Robert and Casella, 2004). The transition kernel of the underlying Markov chain consists of the proposal distribution and the acceptance step. If the proposal distribution is symmetric, that is, if $q(a|b) = q(b|a)$, the acceptance probability reduces to

$$\zeta = \min\left(1,\ \frac{p(x^*)}{p(x^{t-1})}\right) \tag{3.9}$$

and the name of the algorithm shrinks to Metropolis algorithm, which is actually the first MCMC algorithm proposed (Metropolis et al., 1953). Such acceptance probability means that if the value of the target distribution is higher at the candidate than at the previous stage, the step is always accepted, and otherwise it is sometimes accepted, akin to some stochastic optimization algorithms. Because the initialization value of the chain, x^1, may be far from (any) mode of the target distribution, the first values are typically biased and these burn-in samples have to be removed from the final sample representation. Also, successive samples are not independent and may even have the same values, if the transition is rejected. Hence, it might be a good idea to thin the sample set by picking, for instance, only every fifth sample.

A widely used mutation of the Metropolis-Hastings algorithm is the Gibbs sampler, in which the state vector is updated component by component. The proposal distribution for updating the ith component at time step t is

$$q(x_i|\mathbf{x}^t_{1:i-1}, \mathbf{x}^{t-1}) = p(x_i|\mathbf{x}^t_{1:i-1}, \mathbf{x}^{t-1}_{i+1:d}) , \tag{3.10}$$

where d is the dimension of the parameter space, and the moves are always accepted. Hence, in Gibbs sampling the components of the parameter vector are sampled from the conditional target distributions, conditioned on the already sampled components at the current iteration and the remaining components of the previous iteration. The advantage of Gibbs algorithm is therefore the avoidance of having to design a specific proposal distribution. Also in Gibbs sampling, the parameter vector must be somehow initialized.

Basically, MCMC algorithms are efficient methods for exploring a unimodal un-correlated target distribution even in high dimensional spaces. If the posterior is correlated, the convergence is usually slower. However, multimodality of the posterior is a more serious problem. This is due to the low probability of switching the mode. Depending on the initial value, the chain will usually converge to the nearest local mode. In principle, the chain will eventually switch to another mode, and after a long time the chain has visited all the main modes of the distribution and represents it with high accuracy. However, in high dimensional problems, this 'long' time may exceed the remaining age of our solar system, even if all the CPU time of the earth were in use. The result is thus highly biased by the initialization of the chain which is an unwanted feature unless the interest is only on the main mode and prior information about its location is present. Generally, this is not the case and more efficient methods are needed.

3.3.3 Particle Monte Carlo methods

The particle Monte Carlo methods rest upon the idea of representing the target distribution as a weighted particle set. A particle consists of a sampled parameter value and a weight. The parameter values are sampled from a proposal distribution. By denoting the proposal distribution with $q(x)$ and the target distribution with $p(x)$, the Monte Carlo estimate of the mean value of a function $f(x)$ w.r.t. the distribution $p(x)$ can be rewritten as

$$\mathbb{E}_{p(x)}[f(x)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx \approx \frac{\sum_{i=1}^N w^{(i)} f(x^{(i)})}{\sum_{i=1}^N w^{(i)}} , \tag{3.11}$$

where $x^{(i)}$ are sampled from $q(x)$, N is the number of particles and

$$w^{(i)} \equiv \frac{p(x^{(i)})}{q(x^{(i)})} \,. \tag{3.12}$$

The weight w of a particle is thus defined as the ratio of the target distribution and the proposal distribution at the sampled point. The 'ordinary' Monte Carlo estimate (3.6) can be considered as a special case of (3.11) with weights $w^{(i)} = 1/N \;\; \forall i$.

Particle methods often utilize the resampling scheme. Resampling means that the particle indices are sampled again according to their weights. The idea behind resampling is to prune away the samples that are in practice useless for representing the posterior and to produce multiple copies of 'good' particles (Doucet et al., 2001). Resampling is theoretically correct because, on average, a particle set $\{x^{(i)}, w^{(i)}\}$ represents the target distribution equally well as a particle set $\{\tilde{x}^{(i)}, 1\}$, where $\tilde{x}^{(i)}$ have been resampled from $x^{(i)}$ according to $w^{(i)}$ and the weights have been replaced by unity. For instance, consider having four (an artificially small number) particles representing the target distribution, with weights 5, 2, 2 and 1. Hence, the first particle, for example, represents half of the probability mass. Let us name the particles as a, b, c and d. A resampled particle set could be $\{a, a, b, c\}$ with unit weights where the first particle is duplicated and the last particle eliminated (the probability of having particle d in the resampled particle set is $1 - 0.9^4 \approx 0.34$). Different resampling strategies exist, but deterministic methods should be favored as these avoid adding extra variance. Actually, resampling according to the 'full' weight or not resampling at all are just two extremes of resampling according to w^{α}, namely having $\alpha = 1$ and $\alpha = 0$, accordingly. The value of α may be anything between 0 and 1. A low value means that the original particle set is changed only a little, whereas a value close to unity indicates almost full resampling. After resampling according to w^{α}, the weights must be replaced by $w^{1-\alpha}$ to have a properly weighted particle set. Particle methods can be divided into two groups which are next described.

Sequential Monte Carlo method

Sequential Monte Carlo (SMC) method has been popular in dynamic settings where new data arrive sequentially, a typical example being the tracking of a moving object (Robert and Casella, 2004; Doucet et al., 2001). If at time stage t the observation is denoted as y_t and the state of a Markovian system as x_t, and the observations depend only on the current state, the weight can be evaluated recursively as

$$\begin{aligned} w_t &\equiv \frac{p(\mathbf{x}_{1:t} |\, \mathbf{y}_{1:t})}{q(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})} = \frac{p(x_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t})p(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})}{q(x_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t})q(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})} \\ &\propto w_{t-1} \frac{p(y_t|x_t)p(x_t \mid x_{t-1})}{q(x_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t})} \,, \end{aligned} \tag{3.13}$$

where the normalization factor $p(y_t|\ x_{t-1})$ has been omitted. Hence, one only needs to consider the latest observation for estimating the whole posterior distribution. Another appeal of SMC method is that it can be used with whatever kind of state-space models. This makes it more applicable than the traditional Kalman filter which is an analytical method for solving a linear state-space model with Gaussian process noise (Haykin, 2001). However, more complex models can be estimated with the extended Kalman filter and the unscented Kalman filter which are thereby alternatives to using sampling methods. The SMC particles are usually resampled after each observation although it is not necessary. The SMC algorithm is also called particle filter, or bootstrap filter when the prior distribution is used as a proposal.

The SMC algorithm can also be used in static problems although this has not been very common. Chopin (2002) divides the available data into several blocks which are incorporated into the system sequentially. The dimension of the parameter space stays constant; the reasoning behind his approach is to decrease the computation times. A similar idea was presented by Berzuini and Gilks (2003). Tamminen and Lampinen (2006) presented a different approach in which the whole data is made available from the beginning and the components of the parameter vector are updated one by one so that the parameter space expands sequentially. The particles are thus sampled from conditional distributions, conditioned on the already sampled components of the parameter vector, and resampling occurs after updating the components. So, having a t-dimensional parameter vector $\mathbf{x}_{1:t}$, the weight equation (3.12) can be rewritten as (superscript i has been dropped for clarity)

$$w_t \equiv \frac{p(\mathbf{x}_{1:t})}{q(\mathbf{x}_{1:t})} = \frac{p(x_t|\mathbf{x}_{1:t-1})p(\mathbf{x}_{1:t-1})}{q(x_t|\mathbf{x}_{1:t-1})q(\mathbf{x}_{1:t-1})} = w_{t-1}\frac{p(x_t|\mathbf{x}_{1:t-1})}{q(x_t|\mathbf{x}_{1:t-1})}\ . \tag{3.14}$$

The SMC algorithm is illustrated in algorithm 1.

As in MCMC methods, the initialization of SMC algorithm is problematic, since for the first component x_1 there is no conditional distribution. In a Bayesian model, x_1 might be sampled from its marginal likelihood. Another problem with the basic SMC scheme is that the previous components are never re-simulated, that is, their value stays the same throughout the procedure. In light of new observations, the values sampled at the beginning of the algorithm may seem poor. In other words, the values $\mathbf{x}_{1:k}^{(i)}, k < d$ may represent better the marginal posterior $p(\mathbf{x}_{1:k})$ than the full posterior $p(\mathbf{x}_{1:d})$. Also, due to the resampling, all the particles at some stage of the process may share the same ancestor, meaning that for all i there is just one unique value for $\mathbf{x}_{1:k}^{(i)}$ for some $k < d$. However, resampling is advantageous as it prevents the degeneracy, which means that only few particles have non-negligible weights. This phenomenon is unwanted because it is useless to carry irrelevant particles in the set. A neat remedy, if only computationally heavy, for these problems is to add an MCMC step after each resampling stage, which moves the particles in the parameter space to better represent the current posterior distribution. One

```
1. For $i = 1, ..., N$
    • Sample $\tilde{x}_t^{(i)} \sim q(x_t|\mathbf{x}_{1:t-1}^{(i)})$ and set $\tilde{\mathbf{x}}_{1:t}^{(i)} = \{\mathbf{x}_{1:t-1}^{(i)}, \tilde{x}_t^{(i)}\}$.
    • Compute the weight:
          $$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{p(\tilde{x}_t^{(i)}|\mathbf{x}_{1:t-1}^{(i)})}{q(\tilde{x}_t^{(i)}|\mathbf{x}_{1:t-1}^{(i)})}.$$
2. Resample, with replacement, $N$ particles $\mathbf{x}_{1:t}^{(i)}$ from the $\tilde{\mathbf{x}}_{1:t}^{(i)}$'s according to
   values $(\tilde{w}_t)^\alpha$ and set $w_t = \tilde{w}_t^{(1-\alpha)}$.
3. If $t < d$ set $t = t + 1$ and go to step 1. Otherwise, return the set
   $\{\mathbf{x}_{1:d}{}^{(i)}, w_d^{(i)}\}$.
```

Algorithm 1: The SMC algorithm for sampling a d-dimensional target distribution $p(\mathbf{x}_{1:d})$. The number of particles is N.

must also be careful not to assign too large a value to α, at least at the beginning of the process, because the resampling may eliminate particles representing a strong posterior mode that is inferior in the marginal posterior.

Population Monte Carlo method

In SMC algorithm the particles interact after updating each parameter, whereas in the population Monte Carlo (PMC) method the whole parameter vector is sampled before the particles are resampled (Robert and Casella, 2004; Cappe et al., 2004; Mengersen and Robert, 2003). Naturally, in dynamic settings where inferences must be made between the observations the PMC approach is impractical. In static problems, on the other hand, the whole data is available and it seems reasonable to compare the particles based on their estimates of the whole parameter vector instead of their estimates in sub-spaces.

The PMC algorithm is presented as pseudo-code in algorithm 2. A nice property about PMC algorithm is that different proposal distributions q_{it} can be composed for each particle i and at each iteration t. Hence, heuristics can be applied in choosing the proposal distributions without jeopardizing the validity of the method. After each iteration the resampled particles represent the target distribution, and the procedure can be stopped at anytime. Also the parameter values of previous iterations could be included in the final particle set although this is not usually done. In a clever implementation, the proposal distributions are chosen so that the variance in the weights decreases with t, and the particles at $t = T$ give the best representation of the target distribution. In addition, when $t < T$ the particles can be resampled according to w^α, where $\alpha < 1$. This does not generate samples of the

```
1. For $i = 1, ..., N$
     • Sample $\tilde{\mathbf{x}}^{(i)} \sim q_{it}(\mathbf{x})$
     • Compute the weight:
           $w_t^{(i)} = \frac{p(\tilde{\mathbf{x}}^{(i)})}{q_{it}(\tilde{\mathbf{x}}^{(i)})}$
2. Resample, with replacement, $N$ particles $\mathbf{x}_t^{(i)}$ from the $\tilde{\mathbf{x}}^{(i)}$'s according to
   values $w_t$
3. If $t < T$ set $t = t + 1$ and go to step 1. Otherwise, return the set $\{\mathbf{x}_T^{(i)}\}$.
```

Algorithm 2: The PMC algorithm for sampling a target distribution $p(\mathbf{x})$. The number of particles is N and the number of iterations is T.

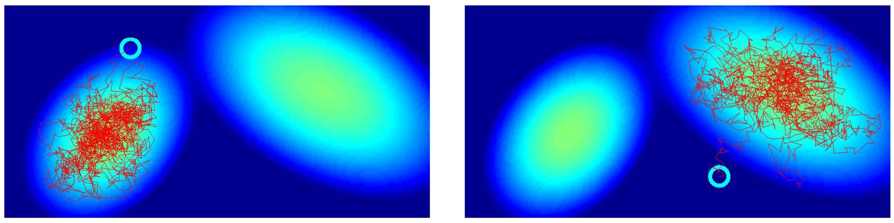


Figure 3.4: Two experiments of a Metropolis algorithm sampling the target distribution (3.15). The starting values of the chains are illustrated as cyan circles. Both the chains have 2000 samples.

target distribution but it prevents the degeneration of the particles. Theoretically, it is enough to use the 'correct' resampling only at the last iteration, as the result of the preceding iterations can be thought of heuristic means to initialize the particles for the final iteration. The last resampling step can also be omitted and weighted particle set be used. Basically, an SMC algorithm where resampling is done only at $t = N$ is a special case of the PMC method with just one iteration. The SMC sampling is actually sometimes called a population Monte Carlo method — the nomenclature is not very clear with particle methods.

3.3.4 Comparison of the sampling methods

The MCMC and particle methods are compared with an example that is simple enough to introduce here yet complex enough to show the differences in the performance. The target distribution is chosen to be a bivariate mixture of two Gaussian distributions:

$$p(\mathbf{x}) = 0.4\mathcal{N}(\mathbf{x}|\mathbf{m}_1, C_1) + 0.6\mathcal{N}(\mathbf{x}|\mathbf{m}_2, C_2)\ . \tag{3.15}$$

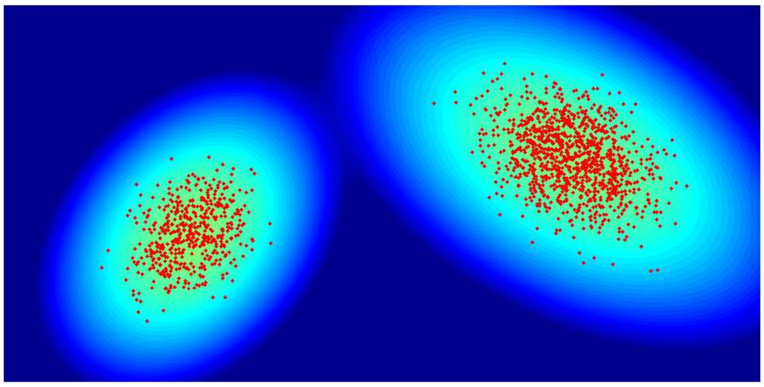


Figure 3.5: The result of the PMC algorithm with target distribution (3.15). The number of particles was 2000 and the algorithm was iterated five times, decreasing the variance of the proposal distribution at each iteration.

The covariances C_1 and C_2 have elongations 2:1 and 3:1, and are rotated by 40 and 120 degrees, respectively. The midpoints of the distributions, $\mathbf{m}_1$ and $\mathbf{m}_2$, were chosen so that the modes are clearly separable in the mixture.

First, the distribution was sampled with the Metropolis sampler using a Gaussian proposal distribution with variance 10 (the distance between the midpoints is 100). Two Metropolis chains containing 2000 samples are shown in Figure 3.4, together with the logarithm of the target distribution. The experiments differ only in the starting values, which are shown as circles in the panels. In both cases, the chains are able to explore only the nearest mode. To get a representation of the whole distribution, the algorithm would have to be run many times with different starting values.

Figure 3.5 shows the performance of the implementation of the PMC algorithm. 2000 particles were randomly initialized over the support of $p(\mathbf{x})$ and the algorithm was iterated five times. The proposal distribution was simply a uniform distribution, whose mean point was the previous sample and whose range decreased linearly during the process, being 100 in the first iteration and 10 in the last iteration. The first iterations are meant to approximately locate the modes and the last iterations are to give a better estimation of the shapes and the masses of the modes. As a result, the particles are distributed in both the modes. The percentage of the particles in the left mode is 34, being relatively close to the true mass ratio 40%.

This simple example thus shows that the particle methods should be favored at least whenever the target distribution is believed to contain multiple separate modes. With a unimodal distribution, MCMC methods might perform better due to their ability to explore a single mode. Basically, the efficiency of the Metropolis sampler could be improved by using simulated tempering techniques or differing the variance of the proposal distribution from time to time so that the chain would occasionally make bigger jumps, enabling the chain to visit both the modes in a reasonable computational time. However, this kind of ad hoc solution will generally fail to work as we can always separate the modes so far apart that the jumps are not big enough. With PMC method, on the other hand, the distance of the modes is irrelevant. Also, having to obey the conditions of Markov chains limits the applicability of MCMC methods. PMC method works especially well when the properties of the model can be utilized, like in the object matching problem studied in this thesis. As to the computational burden, the implementation of the PMC sampler is somewhat slower than that of the Metropolis algorithm because basically each iteration of the PMC sampler corresponds (CPU-wise) to a whole Metropolis algorithm if the number of particles equals the length of the chain. However, the advantage of the particle methods is the independence of the particles, as they interact only in the resampling step. This means that the implementation of the algorithm can be made parallel which can speed up the computations drastically.

Chapter 4

Incremental object matching methods

4.1 Introduction

An incremental (or online) object matching method has never been presented in the literature. However, online learning has been a subject of many studies. For example, tracking problems must naturally be approached with online learning (Doucet et al., 2001). Neural networks have been adapted to learn online (Saad, 1998). A Bayesian approach to online learning was given by Opper (1996) who approximates the posterior with a Gaussian distribution whose parameters are updated after each observation. Online learning has also been considered in the context of active learning (Monteleoni and Kääriäinen, 2007). Active learning is a kind of supervised learning scheme in which the aim is to minimize the amount of human data labelling as it is time consuming and expensive in many problems. Active learners try to infer which of the unlabelled data is most informative so that having a label for these data would most facilitate the learning process. If the queried data is much smaller in size than the available unlabelled data, reductions in time and cost are achieved. In online active learning, the decision about whether to query for the data label is made after each observation (Monteleoni and Kääriäinen, 2007).

Incremental object matching is problematic in a sense that the observations (images) contain also background; hence, the piece of information that is essential for the model must first be extracted before updating the model. Matching an object in a video sequence can be thought of suffering from the same difficulties. However, while the dynamics of the object can be taken into account when, for instance, tracking a person in a video, such a dynamical model is absent when processing images with arbitrary timestamps.

This chapter presents first the theoretical framework for matching the corresponding points recursively using Bayesian methodology. After introducing the theory, three practical implementations of the 'ideal' model are given. The methods differ in their complexities and in their abilities to handle different kinds of

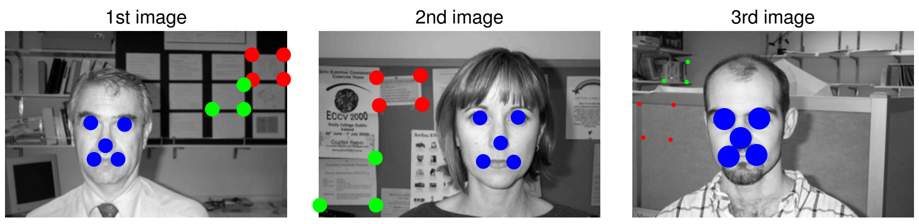


Figure 4.1: An artificial illustration of the Bayesian recursive object matching. The locations of three node sets are depicted with dots, each with different color. The size of the dots is proportional to the posterior probability of the node set. After the first two images, the three hypotheses about the common object are equally likely. Observing the third image causes the hypothesis of the facial node set to surpass the other two. In real experiments, the node set sizes are much larger.

object transformations. The most basic method is introduced in Section 4.3 which is extended in Section 4.4. The most sophisticated implementation, which is named full method, is presented in Section 4.5. The method of 4.3 was published in (Toivanen and Lampinen, 2009a), the method of 4.4 was published in (Toivanen and Lampinen, 2009b) and the method of 4.5 was published in (Toivanen and Lampinen, 2009c) and (Toivanen and Lampinen, 2010).

4.2 Bayesian approach to incremental object matching

In this section, a treatment of the incremental object matching problem in Bayesian terms is given. The problem can be formulated as follows: the model is given a sequence of images, one at the time, and each image contains an instance of the same object (or object class). The task is to incrementally learn a representation of the object, that is, to learn the object model. The used object model is based on local feature points so the learning requires the feature points to be found. The problem can thus be reformulated as having to match the feature points in the images, in other words, to locate the corresponding points. The present object model, which is learned from the images processed so far, is used in matching the next image in the sequence after which the object model is updated.

4.2.1 The Bayesian model

In this work, the feature points are called nodes. The nodes form a node set which is to be matched in the images. The following notation is adopted for images and node locations in them. The tth image in the sequence, which is considered as the current test image, is denoted with $\mathcal{I}_t$ and the locations of the nodes in the image is denoted with $\mathbf{x}_t$. By also adopting a notation $\mathbf{x}_{1:t} \equiv \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_t\}$, the posterior distribution of the location of the nodes can be computed recursively:

$$\begin{aligned} p(\mathbf{x}_{1:t}|\mathcal{I}_{1:t}) &= p(\mathbf{x}_t|\mathbf{x}_{1:t-1},\mathcal{I}_{1:t})p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t}) \\ &= \frac{p(\mathcal{I}_t|\mathbf{x}_{1:t},\mathcal{I}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{1:t-1},\mathcal{I}_{1:t-1})}{p(\mathcal{I}_t|\mathbf{x}_{1:t-1},\mathcal{I}_{1:t-1})}\, p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t}) \\ &\propto p(\mathcal{I}_t|\mathbf{x}_{1:t},\mathcal{I}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{1:t-1})p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t-1})\ , \end{aligned} \tag{4.1}$$

where $p(\mathcal{I}_t|\mathbf{x}_{1:t},\mathcal{I}_{1:t-1})$ is the image likelihood, $p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$ is the prior distribution, and $p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t-1})$ is the posterior distribution of $\mathbf{x}_{1:t-1}$, which is independent of $\mathcal{I}_t$. The (unnormalized) posterior distribution is thus the product of the previous posterior, current prior and current likelihood terms, and is a distribution over all the possible node sets. Figure 4.1 should clarify the issue. The posterior distribution is given only in an unnormalized form as the normalization constant — which scales the distribution so that it integrates to unity — is the integral of the unnormalized posterior over all the possible node configurations and is clearly impossible to compute. The prior distribution $p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$ is chosen to be a Gaussian distribution in a mean free and scale free coordinate system. Depending on the implementation, orientation invariance can be added. It should be noted that although the likelihood seems to depend on the processed images, $\mathcal{I}_{1:t-1}$, the dependence is actually only on the Gabor responses at the matched nodes, as explained in this section. The explicit dependence on every processed pixel is merely a notational matter. So, the previous images need not be stored in memory — otherwise we would have gained nothing compared with the memory requirements of the batch approach!

The likelihoods of the nodes are assumed to be independent of each other. The total likelihood is thus a product of the node likelihoods:

$$p(\mathcal{I}_t|\mathbf{x}_{1:t},\mathcal{I}_{1:t-1}) = \prod_{n=1}^{d} p(\mathcal{I}_t|x^n_{1:t},\mathcal{I}_{1:t-1})\ , \tag{4.2}$$

where n indexes the nodes of the node set, whose size is d, and $x_{1:t}$ indicates the locations of a node in images $1,...,t$ (to distinguish between the node set and a single node, a non-boldface symbol is used for this vector). Basically, the node likelihoods are not fully independent since they are computed from the same image pixels. The closer two nodes are, the higher is their correlation. Modelling the covariance between the node appearances is, however, beyond the scope of this thesis. Besides, the 'naiive Bayes' assumption (independent likelihoods) lightens the sampling so it will be used, in line with (Tamminen, 2005) and (Kalliomäki, 2007). From now on, the node index is dropped and the interest is on an individual node likelihood.

In the example of Figure 4.1, the facial node set consists only object nodes, that is, nodes that are part of the face. In the methods, presented later in this chapter, the node set often includes also other kinds of nodes which are called background nodes. This is illustrated in Figure 4.2. Hence, all the nodes are not necessarily

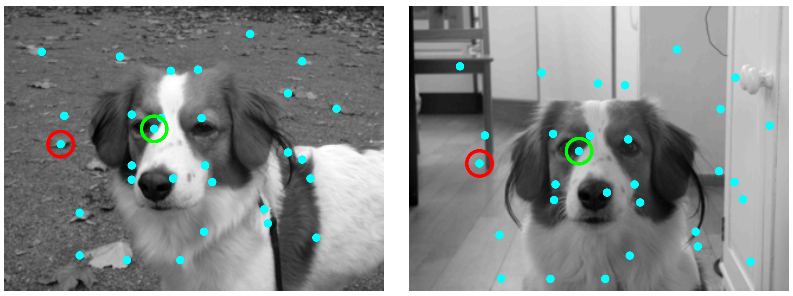


Figure 4.2: An example of matching a node set that includes also background nodes. In the right panel, the dots show a Monte Carlo estimate of the posterior mean of the locations of the points that have most correspondence with the points of the left image, with a condition that the points must have similar spatial configuration. An example of an object node, that is, a 'real' corresponding point, is depicted with green circle whereas an example of a background node that is dissociated from the object is illustrated with red circle.

corresponding points of the object. Figure 4.2 also educationally reveals how the background of the object complicates the matching of the object nodes: two nodes on top of the head are matched in wrong locations because the Gabor similarities are low at the correct location due to the different backgrounds.

When matching node sets with object and background nodes, the system must be able to formally identify them so that only object nodes are associated with the common object. This requires the concept of association to be adopted. The association is indicated with A and its complement with $\overline{A}$, meaning that the node is not associated (dissociated) with the common object. Because it is unknown to the system whether the node in question should be associated or not, the unknown association status is summed out of the likelihood, akin to integrating out nuisance parameters:

$$\begin{aligned} p(\mathcal{I}_t|x_{1:t},\mathcal{I}_{1:t-1}) =& p(\mathcal{I}_t|x_{1:t},\mathcal{I}_{1:t-1},A_t)P(A_t|x_{1:t-1},\mathcal{I}_{1:t-1})+ \\ & p(\mathcal{I}_t|x_{1:t},\mathcal{I}_{1:t-1},\overline{A}_t)P(\overline{A}_t|x_{1:t-1},\mathcal{I}_{1:t-1}) \; . \end{aligned} \tag{4.3}$$

The likelihood of a node is thus a weighted mixture of the associative and dissociative likelihoods. The weights are the prior probabilities for association which are computed recursively as

$$\begin{aligned} P(A_t|x_{1:t-1},\mathcal{I}_{1:t-1}) = & \frac{1}{t-1}P(A_{t-1}|x_{1:t-1},\mathcal{I}_{1:t-1}) \\ & + \frac{t-2}{t-1}P(A_{t-1}|x_{1:t-2},\mathcal{I}_{1:t-2}) \; , \end{aligned} \tag{4.4}$$

that is, a prior association probability is a weighted mixture of the posterior and prior association probabilities of the previous image. Bayes' formula gives the posterior probability for associating the node in image t, given the node locations $x_{1:t}$:

$$P(A_t|x_{1:t}, \mathcal{I}_{1:t}) = \frac{p(\mathcal{I}_t|x_{1:t}, \mathcal{I}_{1:t-1}, A_t)P(A_t|x_{1:t}, \mathcal{I}_{1:t-1})}{p(\mathcal{I}_t|x_{1:t}, \mathcal{I}_{1:t-1})} \ . \tag{4.5}$$

The prior probability for dissociation is the complement of the prior association probability: $P(\overline{A}_t|x_{1:t}, \mathcal{I}_{1:t-1}) = 1 - P(A_t|x_{1:t}, \mathcal{I}_{1:t-1})$.

4.2.2 Associative likelihood

The likelihood, conditioned on that the node is associated with the object, is defined as an un-weighted mixture of likelihood kernels of the processed images:

$$p(\mathcal{I}_t|x_{1:t}, \mathcal{I}_{1:t-1}, A_t) = \sum_{j=1}^{t-1} p(\mathcal{I}_t|x_t, x_j, \mathcal{I}_j, A_t) \ . \tag{4.6}$$

Here x_j refers to the node location in image j. The locations of the nodes in each matched image are thus used as reference in forming the likelihood for the following image. Using a mixture likelihood enables features with different appearances to be matched. For instance, for a node representing a chin of a face, some of the kernels might model a chin with a beard and some a hairless chin.

The associative likelihood kernels are non-linearly mapped from the phase-sensitive similarities (2.6):

$$p(\mathcal{I}_t|x_t, x_j, \mathcal{I}_j, A_t) = \exp(\beta S_p(J(x_t), J(x_j))) \ , \tag{4.7}$$

where β controls the steepness of the mapping, and $J(x_t)$ and $J(x_j)$ are the Gabor jets extracted at the test image location x_t and the reference image location x_j. So, for computing the likelihood kernels, only the reference Gabor jets $J(x_j)$ need to be stored and the actual images can be erased. Equation (4.7) is, actually, a Gaussian distribution of the (complex) normalized filter jet at x_t whose mean is the normalized filter jet at x_j and whose covariance is diagonal with a common variance $1/\beta$. This can be seen by denoting the normalized jets as $\mathcal{J}_t = J(x_t)/\|J(x_t)\|$ and $\mathcal{J}_j = J(x_j)/\|J(x_j)\|$ and computing their $L2$ -norm distance:

$$\|\mathcal{J}_t - \mathcal{J}_j\|^2 = (\mathcal{J}_t - \mathcal{J}_j)^H(\mathcal{J}_t - \mathcal{J}_j) = 2 - 2\,\mathrm{Re}\{\mathcal{J}_t^H \mathcal{J}_j\} = 2(1 - S_p) \tag{4.8}$$

Taking the probability of observing $\mathcal{J}_t$ to be a Gaussian distribution with mean $\mathcal{J}_j$ and covariance $\beta^{-1}\mathbf{I}$, where $\mathbf{I}$ is a unit matrix whose size equals the length of the filter jets, leads to

$$p(\mathcal{J}_t|\mathcal{J}_j, \beta^{-1}\mathbf{I}) \propto e^{-\beta\|\mathcal{J}_t - \mathcal{J}_j\|^2/2} \propto e^{\beta S_p} \ . \tag{4.9}$$

With lots of annotated training data, an optimal value for β could be estimated for each node using (4.9). Since the incremental matching begins with just one image, β is the same for each node and is estimated with other means, discussed in Section 4.7.

4.2.3 Dissociative likelihood

The dissociative likelihood, $p(\mathcal{I}_t|x_{1:t}, \mathcal{I}_{1:t-1}, \overline{A}_t)$, is a difficult subject. Taken literally, it is the probability of observing a certain image detail in a test image, given that the image detail is not to be associated with the reference details. Looking at formulas (4.5) and (4.3), it can be seen that this value, multiplied by $P(\overline{A}_t|x_{1:t-1}, \mathcal{I}_{1:t-1})$, sets a level for how similar the Gabor responses have to be with the reference responses in order for the node to receive a high association probability. With very low value, each node is associated with the common object, while in the other extreme none is associated.

Basically, the mixture likelihood $p(\mathcal{I}_t|x_{1:t}, \mathcal{I}_{1:t-1})$, defined in equation (4.3), only measures how rare or common the observed feature $\mathcal{I}_t(x_t)$ is. This can be demonstrated with a simple example. Let N_s denote the number that certain feature occurs in all the objects of the world and N_f the number of different features in the world. For simplicity, all the possible features are taken to be equally common so N_s is a constant. Let us consider the case $t = 2$ so that there is one reference feature for each node. Also, the location parameter can be dropped by referring to $\mathcal{I}_2(x_2)$ with $\mathcal{F}_2$ and to $\mathcal{I}_1(x_1)$ with $\mathcal{F}_1$. The likelihood function is thus $p(\mathcal{F}_2|\ \mathcal{F}_1)$. The association A can be reformulated as "features $\mathcal{F}_1$ and $\mathcal{F}_2$ depict the same image detail of the same object". The prior probability of A is $1/(N_s \times N_f)$ because there are $N_s \times N_f$ number of features that can be observed. If $\mathcal{F}_2$ is extracted from the correct location the likelihood can be divided in parts as follows:

$$p(\mathcal{F}_2 = \mathcal{F}_2^c|\ \mathcal{F}_1) = 1 \times \frac{1}{N_s N_f} + \frac{N_s - 1}{N_s N_f - 1} \times \frac{N_s N_f - 1}{N_s N_f} = \frac{1}{N_f}\ , \tag{4.10}$$

where the terms correspond to the terms in equation 4.3 and $\mathcal{F}_2^c$ denotes the correct feature. In the opposite case where $\mathcal{F}_2$ represents a different image detail than $\mathcal{F}_1$ the mixture likelihood takes the form

$$p(\mathcal{F}_2 = \mathcal{F}_2^f|\ \mathcal{F}_1) = 0 \times \frac{1}{N_s N_f} + \frac{N_s}{N_s N_f - 1} \times \frac{N_s N_f - 1}{N_s N_f} = \frac{1}{N_f}\ , \tag{4.11}$$

where $\mathcal{F}_2^f$ denotes a false (different than $\mathcal{F}_1$) feature. In both cases the value of the likelihood is — in accordance with intuition — the frequency of the observed feature, $1/N_f$.

Hence, it seems that the likelihood is constant; depending on the observed feature, it just spreads differently in associative and dissociative parts. What is missing in the notation, is the dependence on the assumptions which the model takes, or the information that the model is given. This information can be written as "there is the same object in $\mathcal{I}_1$ and $\mathcal{I}_2$". This piece of information changes the prior assumptions a lot from what they were in the previous example. The second image is not just any image but an image containing the same object which appears in the first image. The prior association probabilities are therefore much higher than one divided by the number of features that can be observed (which is a huge number). If the object is assumed to fill about half of each image, the prior

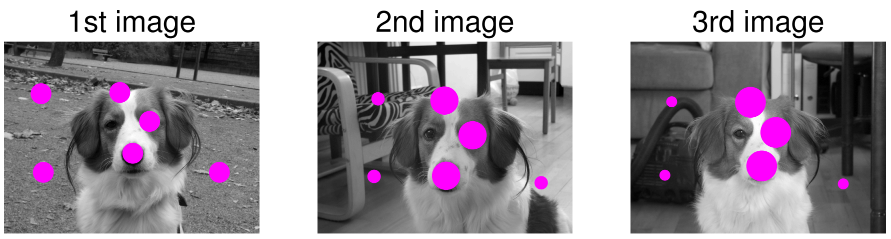


Figure 4.3: An artificial example of the evaluation of a fixed node set in a sequence of three images. The dots mark the mean of the posterior distribution of the node locations and the size of the dots reveals the association probability (large dots are associated with the object).

association probabilities should be initially fixed to half: $P(A_2|x_1, \mathcal{I}_1) = 0.5$ (or, actually, the posterior association probabilities in the first image, $P(A_1|x_1, \mathcal{I}_1)$, are set to half). With this setting, the probability of observing an object feature is, or should be, multifold to the probability of observing a background feature.

Basically, the dissociative likelihood should be modeled with a distribution as in the statistical methodology the likelihood model should be a generative model. This means that it should be possible to generate samples of the likelihood distribution, for example, to predict future data. The prior distribution of the studied method is a generative model because it is a Gaussian model from which it is straightforward to produce samples which would predict the spatial arrangement of the nodes in a new image. However, the likelihood model used in this study is not a generative distribution in a sense that it does not explain the appearance of the whole image as it only models the local appearance around the nodes, the number of which is much less than the number of image pixels. Sullivan et al. (2001) point out that images contain statistical information about where the object is and where it is not; therefore in their Bayesian object localization method they model the object and background with separate distributions. Their method is, however, much simpler than the one presented here for which it is probably impossible to obtain a detailed generative background model. Hence, the dissociative likelihood will be a uniform distribution whose value is denoted with κ. Such an approach is not novel. For instance, Koeser et al. (2006) model the distribution of the occluded pixels with a uniform density. Also in target tracking and surveillance, a constant value for no-detection likelihood is typically used (Bar-Shalom and Li, 1995). Adding a positive constant with the associative likelihood ensures that the total likelihood — being the product of the individual likelihoods — is always non-zero. This makes it possible to match node sets that include background nodes. Setting the dissociative likelihood parameter is discussed in Section 4.7.

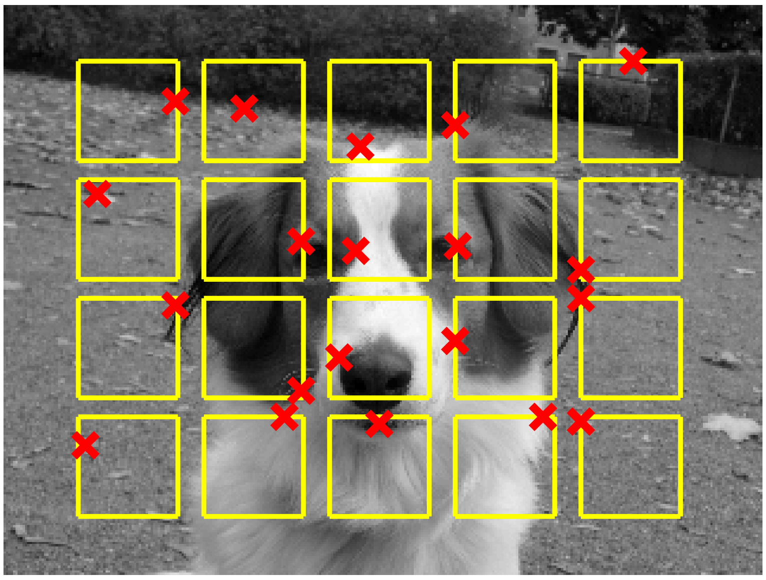


Figure 4.4: An example of the automatically selected nodes in the starting image of the sequence.

4.3 Basic method

This section describes the most basic method for learning the object model. In the method, only a single fixed node set is considered which is placed in the starting image of the sequence using a simple automatic procedure which aims to place the nodes in ’interesting’ locations, distributed approximately evenly in the image. The node set contains object nodes as well as background nodes. In the following images, the posterior distribution of the node set location is formed. The node set is located by sampling the posterior with a sequential Monte Carlo implementation. The prior association probabilities of the object nodes increase to unity along the sequence. If a node is located on an image detail whose appearance has many modes, or which is occluded in some images, the increase of its association probability is slower. The image details of the background nodes typically differ in each image so their association probability tends to zero along the sequence. A representation of the object can be formed anytime as soon as it becomes clear which of the nodes are object nodes. In Figure 4.3, an illustrative example of matching incrementally a set of six nodes in a sequence of three images is shown. The method has some invariance against scale but basically no invariance against orientation changes. In addition, it is preferable to have the object located near the center as the implementation disallows the background nodes to be located outside the image.

4.3.1 Node selection

A simple automatic procedure is used to select the nodes in the first image. The image is divided into small non-overlapping rectangular windows. In each window, a pixel is chosen which maximizes the sum of the magnitudes of the complex Gabor filter responses. An example of the selected nodes in the starting image is illustrated in Figure 4.4. The gaps between the windows prevent many nodes from being selected at the same image detail (at neighboring pixels) which may happen if the windows touch each other. The shape of the selected node set is considered as the reference shape for the subsequent images. The node set in the starting image is thus fixed. It should be noted that the rectangular windows are used only for placing the nodes in the starting image and do not set any limits on where to search for the nodes in the upcoming images.

4.3.2 Posterior distribution of the node locations

As in the basic method only one node set is considered, it can be thought that $p(\mathbf{x}_1|\mathcal{I}_1)$ is a delta distribution at the selected node set. In addition, the likelihood is (basically, see the next subsection) independent on the previous node locations and the prior distribution depends only on the node set of the starting image. Hence, the marginal posterior distribution of image t is simply

$$p(\mathbf{x}_t|\mathcal{I}_{1:t},\mathbf{x}_{1:t-1}) = p(\mathbf{x}_t|\mathcal{I}_{1:t},\mathbf{x}_1) \propto p(\mathcal{I}_t|\mathbf{x}_t,\mathcal{I}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_1) \ , \tag{4.12}$$

where $p(\mathcal{I}_t|\mathbf{x}_t,\mathcal{I}_{1:t-1})$ is the likelihood and $p(\mathbf{x}_t|\mathbf{x}_1)$ is the prior distribution for the location of the node set.

4.3.3 Likelihood model

The likelihood model is basically the same as that presented in Subsection 4.2.1. However, the formulation is somewhat different as there is only one node set, and in the used notation the likelihood is independent on the previous node locations. The likelihood of an individual node of the node set is

$$\begin{aligned} p(\mathcal{I}_t|x_t,\mathcal{I}_{1:t-1}) = {} & p(\mathcal{I}_t|x_t,\mathcal{I}_{1:t-1},A_t)P(A_t|\mathcal{I}_{1:t-1}) \\ & + p(\mathcal{I}_t|x_t,\mathcal{I}_{1:t-1},\overline{A}_t)P(\overline{A}_t|\mathcal{I}_{1:t-1}) \ , \end{aligned} \tag{4.13}$$

where $P(A_t|\mathcal{I}_{1:t-1})$ is the prior association probability:

$$P(A_t|\mathcal{I}_{1:t-1}) = \frac{1}{t-1}P(A_{t-1}|\mathcal{I}_{1:t-1}) + \frac{t-2}{t-1}P(A_{t-1}|\mathcal{I}_{1:t-2}) \ . \tag{4.14}$$

To compute the posterior probability for associating the node in image t, $P(A_t|\mathcal{I}_{1:t})$, the posterior association probability of the node at image location x_t, which resembles equation (4.5), is presented first:

$$P(A_t|x_t,\mathcal{I}_{1:t}) = \frac{p(\mathcal{I}_t|x_t,\mathcal{I}_{1:t-1},A_t)P(A_t|\mathcal{I}_{1:t-1})}{p(\mathcal{I}_t|x_t,\mathcal{I}_{1:t-1})} \ . \tag{4.15}$$

This is integrated over the posterior distribution of $\mathbf{x}_t$ to get the posterior association probability of a node in image t,

$$P(A_t|\mathcal{I}_{1:t}) = \int P(A_t|x_t, \mathcal{I}_{1:t}) p(\mathbf{x}_t|\mathcal{I}_{1:t}) d\mathbf{x}_t \ , \tag{4.16}$$

where

$$p(\mathbf{x}_t|\mathcal{I}_{1:t}) = \int p(\mathbf{x}_t|\mathcal{I}_{1:t}, \mathbf{x}_1) p(\mathbf{x}_1|\ \mathcal{I}_{1:t}) d\mathbf{x}_1 = p(\mathbf{x}_t|\mathcal{I}_{1:t}, \mathbf{x}_1) \tag{4.17}$$

as the nodes in the first image, $\mathbf{x}_1$, are fixed having no distribution (or they can be thought of obeying a delta distribution at $\mathbf{x}_1$). Note that the integration in (4.16) is performed over the posterior distribution of the location of the whole node set.

Next, the associative likelihood term is integrated over the distribution of $J_{1:t-1}$ which denotes the Gabor responses of the node in images $1, ..., t-1$:

$$\begin{aligned} p(\mathcal{I}_t|x_t, \mathcal{I}_{1:t-1}, A_t) &= \int p(\mathcal{I}_t|x_t, \mathcal{I}_{1:t-1}, A_t, J_{1:t-1}) p(J_{1:t-1}|\mathcal{I}_{1:t-1}) dJ_{1:t-1} \\ &\approx p(\mathcal{I}_t|x_t, A_t, \hat{J}_{1:t-1}) \ , \end{aligned} \tag{4.18}$$

where the dependence on $\mathcal{I}_{1:t-1}$ has been dropped and a point estimate for $J_{1:t-1}$ has been plugged in. This means that the posterior distribution of $J_{1:t-1}$ — which is independent of x_t and A_t — is taken to be a delta function at the point estimate, being for image k:

$$\hat{J}_k = \frac{\int J(x_k) P(A_k|x_k, \mathcal{I}_{1:k}) p(\mathbf{x}_k|\mathcal{I}_{1:k}, \mathbf{x}_1) d\mathbf{x}_k}{\int P(A_k|x_k, \mathcal{I}_{1:k}) p(\mathbf{x}_k|\mathcal{I}_{1:k}, \mathbf{x}_1) d\mathbf{x}_k} \tag{4.19}$$

For the first image, the Gabor responses $\hat{J}_1$ are naturally evaluated at the selected nodes. The associative likelihood is taken to be a mixture of likelihood kernels which are one for each processed image so that the appearance of each matched node is a reference appearance for the node in the test image:

$$p(\mathcal{I}_t|x_t, A_t, \hat{J}_{1:t-1}) = \sum_{k=1}^{t-1} p(\mathcal{I}_t|x_t, A_t, \hat{J}_k) \ . \tag{4.20}$$

The likelihood kernels are built from the similarity measures (2.6), as in (4.7):

$$p(\mathcal{I}_t|x_t, A_t, \hat{J}_k) = \exp(\beta S_p(J(x_t), \hat{J}_k)) \ . \tag{4.21}$$

4.3.4 Prior model

The mean shape of the node set in a test image, a priori to observing the image, is assumed to be the same as in the starting image, after put into the same location and scale, with independent Gaussian deviations on the nodes. The prior distribution,

$p(\mathbf{x}_t|\mathbf{x}_1)$, is therefore defined as a Gaussian distribution in a translation and scale free space:

$$p(\mathbf{x}_t|\mathbf{x}_1) = \mathcal{N}\left(\frac{\mathbf{x}_t - E[\mathbf{x}_t]}{s(\mathbf{x}_t, \mathbf{x}_1)} \mid \mathbf{x}_1 - E[\mathbf{x}_1], \sigma^2\mathbf{I}\right) , \tag{4.22}$$

where $\mathbf{I}$ denotes a unit matrix, $E[\mathbf{x}]$ is the mean value of $\mathbf{x}$, σ^2 is the pre-fixed variance of the nodes and $s(\mathbf{x}_t, \mathbf{x}_1)$ is the scale of the node set $\mathbf{x}_t$ w.r.t. the node set $\mathbf{x}_1$. The scale is computed from the node locations:

$$s(\mathbf{x}_t, \mathbf{x}_1) = \frac{\sqrt{\sigma_u(\mathbf{x}_t)^2 + \sigma_v(\mathbf{x}_t)^2}}{\sqrt{\sigma_u(\mathbf{x}_1)^2 + \sigma_v(\mathbf{x}_1)^2}} , \tag{4.23}$$

where $\sigma_u(\mathbf{x})$ is the standard deviation of the horizontal components of the nodes in $\mathbf{x}$ and $\sigma_v(\mathbf{x})$ is that of the vertical components. Since the same Gabor filters are used with each scale, the likelihood values change with large scales. Therefore the performance of the system decreases if the size difference between the object instances in different images is large (say, more than one and a half).

4.3.5 Sequential Monte Carlo sampling

In the SMC implementation, all the data are available from the start and the parameters (i.e. the coordinates of the nodes) are updated sequentially. The posterior distribution needs therefore to be defined in a conditional form so that it is possible to sample the ith node, given the locations of the already sampled nodes $\backslash i$. Since the node likelihoods are independent, it is enough to express the prior distribution in a conditional form which is straightforward for a Gaussian distribution with a diagonal covariance matrix. For clarity, the image number index is dropped so $\mathbf{x} \equiv \mathbf{x}_t$ and $\mathbf{x}' \equiv \mathbf{x}_1$ denote the test and reference node sets:

$$p(x_i|\mathbf{x}_{\backslash i}, \mathbf{x}') = \mathcal{N}\left(\frac{x_i - E[\mathbf{x}_{\backslash i}]}{s(\mathbf{x}_{\backslash i}, \mathbf{x}'_{\backslash i})} \mid x'_i - E[\mathbf{x}'_{\backslash i}], \sigma^2\mathbf{I}\right) . \tag{4.24}$$

The SMC implementation is illustrated in Algorithm 3. Apart from the first component, the sampling order of the particles is sampled deterministically according to the prior association probabilities so that the nodes that probably associate with the object tend to be matched first. Each particle thus samples the nodes in different order. The first component of the particles is matched from the likelihood. For the following components, the proposal distribution is a mixture of likelihood and prior terms, with the mixture coefficient φ heuristically determined on the basis of the node association probability. The purpose is that the associative nodes are sampled from the likelihood and the dissociative nodes from the prior. In the algorithm, only local area R around the prior mean is used. During the process, the particles containing high likelihood nodes will survive the resampling and thus the partially matched node sets will contain different subsets of nodes that probably associate with the object.

```
1. Initialization, $m = 1$
  for $j$ = 1 to $N$ do
    - Assign indices of first components for each particle, $\mathbf{J}_1^{(j)} = j \mod d$
    - Sample $x_i^* \sim p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, A_t)$ using $i = \mathbf{J}_1^{(j)}$
    - Set $\theta_1^{(j)} = x_i^*$ and $w_1^{(j)} = 1$
  end for
  - Set $m = 2$
2. Importance sampling
  for $j$ = 1 to $N$ do
    - Assign $\mathbf{J}_m^{(j)}$ from $1, ..., d$ according to the prior association probabilities
    $P(\mathbf{A}_t|\mathcal{I}_{1:t-1})$ so that $\mathbf{J}_m^{(j)} \neq \mathbf{J}_{1:m-1}^{(j)}$
    - Assign $i = \mathbf{J}_m^{(j)}$ and $\mathbf{x}_{\setminus i} = \theta_{1:m-1}^{(j)}$
    - Sample $x_i^* \sim q(x_i|\mathbf{x}_{\setminus i}, \mathbf{x}', \mathcal{I}) = \varphi \frac{p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1})}{\sum_{x_i \in R} p(\mathcal{I}_t|x_t, \mathcal{I}_{1:t-1})} + (1-\varphi) \frac{p(x_i|\mathbf{x}_{\setminus i}, \mathbf{x}')}{\sum_{x_i \in R} p(x_i|\mathbf{x}_{\setminus i}, \mathbf{x}')}$
    where $\varphi = 1 - \exp\left(1 - \frac{\sum_{x_i \in R} [p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1})]}{\sum_{x_i \in R} p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, \overline{A}_i) P(\overline{A}_i|\mathcal{I}_{1:t-1})}\right)$
    - Set $\tilde{\theta}_{1:m}^{(j)} = \{\theta_{1:m-1}^{(j)}, x_i^*\}$
    - Set $\tilde{w}_m^{(j)} = w_{m-1}^{(j)} \frac{p(x_i^*|\mathbf{x}_{\setminus i}, \mathbf{x}', \mathcal{I}_{1:t})}{q(x_i^*|\mathbf{x}_{\setminus i}, \mathbf{x}', \mathcal{I}_{1:t})}$
  end for
3. Reducing the particle number
  - Denote the current number of particles as $N' = N$
  - Eliminate M particles with lowest weights, where $M/N \ll 1$
  - Set the new number of particles to be $N = N' - M$
4. Resampling with Langevin MOVE step
  - Resample with replacement $N$ particles $(\theta_{1:m}^{(j)}, j = 1, ..., N)$ from the set
  $(\tilde{\theta}_{1:m}^{(j)}, j = 1, ..., N)$ according to the importance weights $(\tilde{w}_{1:m}^{(j)})^{\alpha}$
  for $j$ = 1 to $N$ do
    - For $l = 1, ..., m$, assign $i = \mathbf{J}_l^{(j)}$ and set $E_l = p(\mathcal{I}|x_i, \mathcal{I}_{1:t-1})$ with proba-
    bility $P(A_t|\theta_l^{(j)}, \mathcal{I}_{1:t})$, otherwise set $E_l = p(x_i|\theta_{1:l-1,l+1:m}^{(j)}, \mathbf{x}')$
    - For $n = 1 : N_{iter}$, assign $\setminus i = \mathbf{J}_{1:m}^{(j)}$ and
    sample $\theta_{1:m}^{(j)} \sim \theta_{1:m}^{(j)} - \frac{\epsilon^2}{2} \frac{\partial E_{1:m}}{\partial x_{\setminus i}} (\theta_{1:m}^{(j)}) + \epsilon \mathcal{N}(0, 1)$
  end for
  - Set $w_m^{(j)} = (\tilde{w}_m^{(j)})^{(1-\alpha)}$
  - If $m < d$ set $m = m + 1$ and go to step 2
```

Algorithm 3: The sequential Monte Carlo implementation. The number of particles and nodes are N and d, and $\boldsymbol{\theta}$ denotes the parameter vector. Also, following notation is adopted: $\mathbf{x} \equiv \mathbf{x}_t$, $\mathbf{x}' \equiv \mathbf{x}_1$.

Compared with the basic SMC algorithm, some modifications have been made that take into account the static nature of the problem. The particles are resampled according to w^α, $\alpha < 1$, in order to keep the particle weights of the previous nodes and to decrease the variance of the weights. After sampling each node, a Langevin Monte Carlo (Neal, 1993) step is applied to improve the sampled parameter values. At the end of sampling, the particles are sampled with an additional Gibbs sampling step due to its mode exploring capabilities. Also, it is desired to save computation time without impoverishing the results. Since the hypothesis of the correct mode is assumed to get stronger as more components are matched, the particle number is reduced along the sampling procedure by removing some proportion of particles with the lowest weights before resampling.

4.3.6 The algorithmic form

```
1. Gabor transform the starting image $\mathcal{I}_1$
2. Select the nodes in the starting image and store the Gabor responses at each node. Set $t = 2$
3. Gabor transform the next image $\mathcal{I}_t$
4. Compute the likelihood as explained in Subsection 4.3.3
5. Sample the image with SMC Algorithm 3
6. Estimate and store the posterior association probabilities of the nodes with equations (4.15) and (4.16)
7. Compute the mean Gabor responses with equation (4.19) and store them
8. Set $t = t + 1$ and go to step 3
```

Algorithm 4: The incremental algorithm for matching the corresponding points with the basic method.

Algorithm 4 shows pseudo code instructions for matching the corresponding points with the basic method.

4.4 Extensions to the basic method

The method presented in this section is an extended version of the basic version of the previous section. The main differences are the increased invariance against translation, scale and orientation changes, a different sampling method as the posterior distribution is sampled with population Monte Carlo algorithm, the modification of the node set during the procedure as the nodes with low association

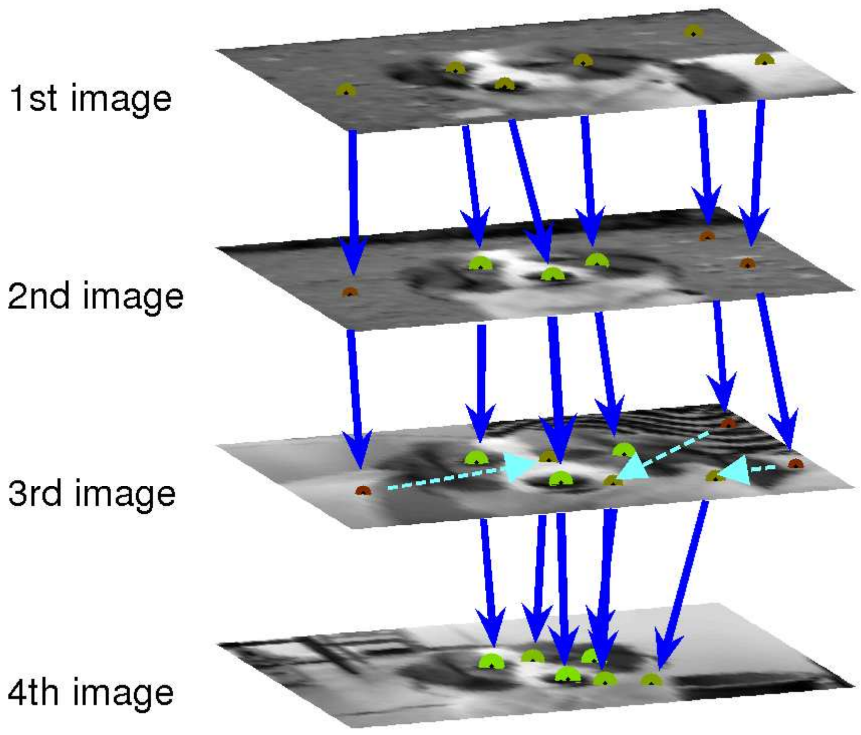


Figure 4.5: A schematic illustration of the method. In the first image, six nodes are selected (yellow dots) which are matched in the subsequent images (solid blue arrows and dots). The node association probability is high for large green dots and low for small red dots. In the third image, three nodes with low association probabilities are replaced by new nodes (dashed arrows) that are selected inside the convex hull of the current node set.

probabilities are eliminated and new nodes laid on locations that more probably cover the object, and the updating of the node variances. A schematic illustration of the method is presented in Figure 4.5.

4.4.1 Node selection

The nodes are selected in the starting image in a similar way as in the basic method, except that there is more emphasis on placing the nodes as far from each other as possible. The starting image is divided into windows in which the sum of the norm of a vector of the complex Gabor filter response magnitudes and the Gaussian distribution whose mean is the midpoint of the window, is maximized. As a result, the nodes are spread fairly evenly in the image, with high information content.

4.4.2 Posterior distribution

In this method, the scale and orientation of the node set in the current test image w.r.t. the reference node set are treated as the parameters of the model. They are denoted with s and φ and can be considered as nuisance parameters that can be integrated out of the posterior distribution of the current node set:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{x}_{1:t-1},\mathcal{I}_{1:t}) &= \int p(\mathbf{x}_t, s, \varphi|\mathbf{x}_{1:t-1},\mathcal{I}_{1:t})\, ds\, d\varphi \\ &= \frac{1}{p(\mathcal{I}_t \mid \mathcal{I}_{1:t-1}, \mathbf{x}_{1:t-1})} \int p(\mathcal{I}_t|\mathbf{x}_{1:t},\mathcal{I}_{1:t-1}, s, \varphi) p(\mathbf{x}_t, s, \varphi|\mathbf{x}_{1:t-1})\, ds\, d\varphi \\ &\propto \int p(\mathcal{I}_t|\mathbf{x}_t,\mathcal{I}_{1:t-1}, s, \varphi) p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, s, \varphi) p(s, \varphi)\, ds\, d\varphi\ , \end{aligned} \quad (4.25)$$

where $p(s, \varphi) \propto 1$ is a non-informative prior distribution for scale and rotation and the likelihood is again independent on the node locations of the processed images $\mathbf{x}_{1:t-1}$. In principle, the prior distribution depends on $\mathbf{x}_{1:t-1}$ but as explained in subsection 4.4.4, only a point estimate for $\mathbf{x}_{1:t-1}$ will be used.

4.4.3 Likelihood model

The likelihood model is the same as the one explained in Subsection 4.3.3 apart from the dependence on scale and orientation. Hence, the likelihood for image t is $p(\mathcal{I}_t|\mathbf{x}_t,\mathcal{I}_{1:t-1}, s, \varphi)$. For computing the value of the likelihood with an arbitrary scale and orientation, the Gabor jets are estimated by interpolating four neighboring filter responses as described in Subsection 2.5.5. Also, unlike in the basic method, the nodes are allowed to locate outside the image which increases the tolerance against translation. The associative likelihood of a node lying outside the image area is set to zero.

4.4.4 Prior model

The prior distribution for image t is extended from the basic prior model to include the possibility of orientation change:

$$p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, s, \varphi) = \mathcal{N}(\mathbf{x}_t|E[\mathbf{x}_t] + sR(\varphi)(\hat{\mathbf{x}}_{1:t-1} - E[\hat{\mathbf{x}}_{1:t-1}]),\ s^2 C(\varphi))\ . \quad (4.26)$$

In the equation, $\hat{\mathbf{x}}_{1:t-1}$ is the reference node set which is computed from $\mathbf{x}_{1:t-1}$ and is given in formula (4.28), $R(\varphi)$ is the rotation matrix for orientation angle φ:

$$R(\varphi) = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{pmatrix}\ , \quad (4.27)$$

and $C(\varphi)$ is the covariance matrix in which the horizontal and vertical components of the nodes are rotated with the rotation matrix.

The reference shape $\hat{\mathbf{x}}_{1:t-1}$ is a mixture of the mean free shapes of the processed images, weighted by their posterior association probabilities which for node i in image k is $P(A_k^i|\mathcal{I}_{1:k})$. The reference location of node i is thus computed as

$$\hat{x}^i_{1:t-1} = \frac{\sum_{k=1}^{t-1} P(A_k^i|\mathcal{I}_{1:k})(E[x_k^i] - M_{x,k})}{\sum_{k=1}^{t-1} P(A_k^i|\mathcal{I}_{1:k})} , \tag{4.28}$$

where $E[x_k^i]$ is the location of x_k^i, averaged over the posterior distribution of it. It should be noted that $\hat{x}^i_{1:t-1}$ and $E[x_k^i]$ are vectors of length two as they contain horizontal (u) and vertical (v) components. In the equation, $M_{x,k}$ denotes the midpoints of the matched images which are computed in such a way that they minimize the weighted variance of the horizontal and vertical components of the distances of the nodes to the midpoints. For the horizontal component u the variance is

$$E_u = \frac{\sum_i \sum_k V_k^i \left((\Delta u)_k^i - \frac{\sum_{k'} V_{k'}^i (\Delta u)_{k'}^i}{\sum_{k'} V_{k'}^i} \right)^2}{\sum_{k'} V_{k'}^i} , \tag{4.29}$$

where i indexes the nodes, k indexes the images, $V_k^i \equiv P(A_k^i|\mathcal{I}_{1:k})$ is the posterior association probability of node i in image k, and $(\Delta u)_k^i \equiv u_k^i - m_{u,k}$ is the horizontal coordinate of $E[x_k^i]$ in $m_{u,k}$ mean coordinate system. The error function E_u is minimized by differentiating it w.r.t. $M_u = \{m_{u,1}, m_{u,2}, ..., m_{u,t-1}\}$ and setting the result to zero. By denoting with V a matrix with elements V_k^i and with U a matrix with elements u_k^i, the solution is

$$\begin{aligned}
M_u =& (Z)^{-1} B_u \ , \quad \text{where} && (4.30)\\
Z =& \mathrm{d}\left[\left(\frac{V}{VI_{T' \times T'}} \right)^T I_{N' \times 1} \right] - 2 \left(\frac{V}{(VI_{T' \times T'})^2} \right)^T V \\
&+ \left(\frac{V}{(VI_{T' \times T'})^3} \right)^T \mathrm{d}[VI_{T' \times 1}] V \quad \text{and} && (4.31)\\
B_u =& \mathrm{D}\left[\left(\frac{V}{VI_{T' \times T'}} \right)^T U \right] - 2 \left(\frac{V}{(VI_{T' \times T'})^2} \right)^T \mathrm{D}[VU^T] \\
&+ \left(\frac{V}{(VI_{T' \times T'})^3} \right)^T \mathrm{d}[VI_{T' \times 1}] \mathrm{D}[VU^T] \ . && (4.32)
\end{aligned}$$

Here d$[w]$ means constructing a diagonal matrix of vector w, D$[W]$ means taking the main diagonal of matrix W, $I_{a \times b}$ is an $a \times b$ -sized matrix of ones, the number of processed images is $T' \equiv t - 1$, and N' is the size of the node set. The divisions and exponentiations are made element-wise.

The node variances (that is, the diagonal elements of the covariance matrix C) are updated after each match, in order to allow the system to learn the level of rigidity of the object being matched. Having an inverse-gamma prior distribution on the variance leads to an analytical form for the conditional posterior distribution

(Gelman et al., 2004). Furthermore, using the mode of the posterior distribution and setting the observed variance in horizontal (u) direction v_u^i to the point estimate at the posterior mean, the prior node variance of node i is updated as

$$(\sigma_{u,prior}^i)_t^2 = \frac{\nu_0 + t - 1}{\nu_0 + t + 1} \frac{\nu_0\sigma_0^2 + (t-1)v_u^i}{\nu_0 + t - 1} , \tag{4.33}$$

where ν_0 and σ_0 are the hyperparameters of the prior distribution of $(\sigma_{prior})_2$, that is, the prior node standard deviation in the second image. The vertical prior node variances are computed in a similar fashion. In addition, in order to decrease the importance of the starting image the scale and orientation are averaged over the matched images so that, for instance, the scale in the test image is defined not w.r.t. the node set size in the starting image but w.r.t. the average node set size.

4.4.5 Relocating the nodes

With more object nodes and fewer background nodes the object model improves, making it easier to match the node set in the next image. Therefore, the node set is modified by replacing the nodes whose prior association probability goes below a threshold with new nodes so that the total number of the nodes stays constant. Because in the starting image the nodes are distributed over the whole image, the new nodes are positioned inside the convex hull (Sonka et al., 1999) of the existing nodes so that the object is approached 'from outside' by gradually shrinking the area of the node set. For non-convex objects, this approach leads to a trial end error type procedure as a new node may not get correspondence in the subsequent images and is again moved to a new position. The new nodes are positioned by maximizing the information content while also penalizing the vicinity of other nodes, as in selecting the nodes in the starting image. For these new nodes, the prior association probabilities are set to half, the previous posterior association probabilities are set to zero, the prior node variances are set to the initial values, and the previous kernels in the likelihood mixtures are removed. To prevent some image from having too much control over the new node locations, an upper limit can be set on the number of the modifications.

4.4.6 Sampling

For representing the distributions and computing the necessary integrals, the posterior distribution is sampled with population Monte Carlo method. As described in Section 3.3, in PMC the proposal distributions may differ between particles so the variance of the proposal distributions is let to decrease for associable nodes. The first round of the PMC implementation is special as the particle weight is taken to be $w^{1/3}$ to prevent degeneration — theoretically this does not produce samples of the posterior but acts as an (other) initialization for the second round. The PMC implementation is presented in greater detail in Algorithm 5, where indexing for different particles is omitted for the sake of clarity so steps 2 and 3 are

```
1. Initialize
   - Pre-compute a large filter bank consisting of five scales $\sqrt{2}\pi/\{2, 4, 8, 16, 32\}$ and 12 orientations $\{0, \pi/6, ..., 11\pi/6\}$
   - Initialize the particles as explained in Subsection 4.4.7
   - Set $h = 1$.
2. Sample each particle from the proposal distributions
   - Compute the posterior association probabilities with (4.15) and set $T_i = P(A_t|x_t, \mathcal{I}_{1:t})$
   - Set $q(\log[s]|s^{h-1}) = \mathcal{N}(\log[s^{h-1}], \sigma_s^2)$, where $\sigma_s = \log[1.06]/(1 + E[T])$
   - Set $q(\varphi|\varphi^{h-1}) = \mathcal{N}(\varphi^{h-1}, \sigma_\varphi^2)$, where $\sigma_\varphi = 2/(1 + E[T])$ (in radians)
   - Sample scale and angle: $\log[s^*] \sim q(\log[s]|s^{h-1})$ , $\varphi^* \sim q(\varphi|\varphi^{h-1})$
   for $i$ = 1 to $N_{\text{nodes}}$ do
      - Set $q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*) = \mathcal{N}(x_i|m_\mathbf{x} + s^* R(\varphi^*)(x_i^{h-1} - E[\mathbf{x}^{h-1}]),\ R(\varphi)C_i)$, where $m_\mathbf{x}$ is the $\mathbf{T}$ weighted midpoint of $\mathbf{x}^{h-1}$ and $C_i = \begin{pmatrix} \sigma_{u_i}^2 & 0 \\ 0 & \sigma_{v_i}^2 \end{pmatrix}$, where $\sigma_{u_i} = s^*(\sigma_{u,prior}^i)^{\sqrt{2-T_i}}$ and $\sigma_{v_i} = s^*(\sigma_{v,prior}^i)^{\sqrt{2-T_i}}$
      - Set $q(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, s^*, \varphi^*) = P(A_i|\mathcal{I}_{1:t-1})\tilde{L}_i(x) + (1 - P(A_i|\mathcal{I}_{1:t-1}))\kappa$, where $\tilde{L}_i(x)$ is the (pre-computed) likelihood, with the scale and orientation closest to $s^*$ and $\varphi^*$
      - Set $Z_i = \sum_{x_i \in A} q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*) q(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, s^*, \varphi^*)$, where the size of the search window $A$ is $[3\ \ 3]C_i$ pixels around the prior mean
      - Set $q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t}) = \frac{1}{Z_i} q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*) q(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, s^*, \varphi^*)$
      - Sample location (numerically): $x_i^* \sim q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t})$
   end for
3. Compute the particle weights for each particle
   - Compute $w = \frac{p(\mathcal{I}_t|\mathbf{x}^*, \mathcal{I}_{1:t-1}, s^*, \varphi^*) p(\mathbf{x}^*|\mathbf{x}_{1:t-1}, s^*, \varphi^*)}{[\prod_{i=1}^{N_{\text{nodes}}} q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t})] q(\log[s^*]|s^{h-1}) q(\varphi^*|\varphi^{h-1})}$, where the value of the likelihood has been computed by interpolating between four neighboring filter responses, according to $s^*$ and $\varphi^*$.
   - If $h = 1$, set the particle weight to $w^{1/3}$, otherwise set the weight to $w$
4. Resample and move the particles
   - Resample the particles with replacement according to the particle weights using deterministic resampling
   - Set the components of the particles $\{\mathbf{x}^h, s^h, \varphi^h\}$ to the resampled values
   - Move each particle with Langevin Monte Carlo sampling step
   - If $h > 1$ and the variance of $\mathbf{x}^t$ is below a threshold, or if $h = h_{\text{max}}$, finish. Otherwise, set $h = h + 1$ and go to step 2
```

Algorithm 5: The population Monte Carlo implementation. See text for details.

performed individually for each particle. Also, the subindex of the current image t has been dropped. The various parameter values, used in the implementation of (Toivanen and Lampinen, 2009b) and in the experiments of this thesis, are inserted in the algorithm. The number of the best scores in the initialization step, and hence the number of particles, was chosen to be 150 to have a sensible computational time. The hypothesis is that the correct approximative match is included in these 150 best scores.

4.4.7 Particle initialization

The PMC particles are initialized by utilizing the global move step of the Elastic Bunch Graph Matching method (Wiskott et al., 1997), in which an approximative location of the object is found by sparsely scanning the test image with a rigid reference node set and at the same time measuring the average node similarity. In this implementation, the test image is scanned using five different scales $s = 2^{\{-0.8,-0.4,0,0.4,0.8\}}$ and three orientations $\varphi = \{-10°, 0°, +10°\}$; hence the scanning is done $5 \times 3 = 15$ times with different scale / orientation combinations. The decision of choosing only a small amount of orientation angles is based on the fact that the global move step is sensitive to small orientation changes. The goal is to increase invariance against small changes, rather than to obtain full orientation invariance which would, of course, lengthen the computation times drastically and possibly lead to problems with memory. The scales were chosen to balance a need to cover a relatively large range of different scale changes and a need to use small enough interval between different scales so as to avoid missing the target. The interval of the successive matches in the horizontal and vertical directions was chosen to be $1/80$ times the amount of pixels in the corresponding direction, being for a conventional 240×320 -sized image 3 pixels in the vertical and 4 pixels in the horizontal direction. For each match, the similarities are computed using Gabor jets consisting of three frequencies and six orientations, by interpolating the filter responses with four neighboring filters of the pre-computed large filter bank. Each match is given a score, defined as the mean value of the most similar 25 % of the nodes, using the phase-insensitive similarity. The 150 best scores (matches) are chosen and the particles are initialized with the corresponding locations. The initialized values are assigned to $\mathbf{x}^0$, s^0 and φ^0.

4.4.8 The algorithmic form

The incremental object matching method with improvements presented in this section is summarized in Algorithm 6.

4.5 Full method

Section 4.3 presented a basic method for incremental object matching which was improved in Section 4.4. There is still room for further improvements. For in-

```
1. Actions for the starting image:
    (a) Gabor transform the starting image
    (b) Select the nodes in the starting image as explained in the text and store
        the Gabor responses at each node
2. Match the next image
    (a) Gabor transform the next image
    (b) Compute the likelihood as explained in Subsection 4.4.3
    (c) Match the node set with PMC (Algorithm 5)
3. Compute the parameter values
    (a) Estimate and store the posterior association probabilities of the nodes
        with (4.16)
    (b) Estimate and store the magnitudes and phases of the Gabor responses
        with (4.19)
    (c) Update the prior node variance with (4.33)
4. Modify the node set
    (a) For each node with $P(V_i) < P_{th}$, reposition the node as explained in
        Subsection 4.4.5
    (b) Update the reference shape with (4.28) and (4.30)
5. Go to step 2
```

Algorithm 6: The incremental algorithm for matching the corresponding points with the extended method.

stance, it may seem unnecessary to match a node set that contains background nodes. Modifying the node set, as presented in the previous section, is a way to eventually get rid of the background nodes. However, it may take dozens of images before the node set contains only object nodes. The method, presented in this section, aims to exclude any background nodes from the beginning of the image sequence. This is made possible by 'fully' utilizing the recursive formula for the joint posterior distribution for $\mathbf{x}_{1:t}$, presented in equation (4.1): the joint posterior for all the possible node sets, not just one fixed set, is considered. The purpose is thus to locate the most corresponding point sets in all the images instead of only locating in the upcoming images the corresponding points of the node set that was fixed in the starting image.

The particle Monte Carlo is again a natural method for approaching the problem: Each particle, in principle, corresponds to a different node set and represents

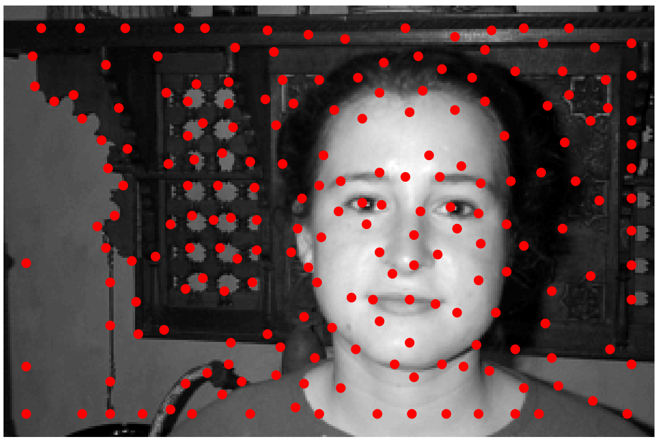


Figure 4.6: An example of the selected candidate nodes in a starting image.

a hypothesis about the common object in the images. This means that different hypotheses can be carried along the sequence (see Figure 4.1). Owing to the recursive formula of the posterior distribution, the posterior probabilities of the false hypotheses diminish when observing new images whereas the correct hypotheses (object nodes) strengthen. The number of different hypotheses — or in practice the number of Monte Carlo particles representing different modes in the parameter space — is controlled in the resampling stage. Also, the matching is made fully orientation invariant, without having memory problems or hopelessly long computation times. This method can be considered to be the most complex and sophisticated of the three methods and is expected to outperform the other methods on difficult images.

4.5.1 Selecting candidate nodes in the starting image

For computational reasons the starting image is downsampled by selecting N_p candidate nodes, N_p being typically few hundreds. The candidate nodes can be considered as forming a node pool from which the node sets are extracted. The candidate nodes are selected in visually interesting locations, such as corners and edges, by taking the local maxima of the sum of the absolute values of the Gabor filter responses, akin to the node selection schemes of the previous methods. The vicinity of the already selected nodes is penalized in a Gaussian way using such a variance that the selected nodes approximately cover every part of the image. Furthermore, the details in the image should be represented by at most one node so that there would not be many nodes on the same edge, for instance. This is realized by forming the phase-sensitive Gabor similarity of the selected node with the image and cropping out from the image the local maximum around the node before selecting the next node. It can be thought that the selected nodes are local maxima of the posterior distribution in the first image, $p(\mathbf{x}_1|\mathcal{I}_1)$, where the likelihood $p(\mathcal{I}_1|\mathbf{x}_1)$

is the sum of Gabor filters, and the prior assumption $p(\mathbf{x}_1)$ is that the nodes are spread at regular intervals. An example of the selected nodes is illustrated in Figure 4.6. An upper limit is set for the number of nodes used for the representation of the object, $N_{\max} < N_p$; the size of the node set can be anything between one and $N_{\max}$.

Discarding pixels in the starting image means that the distribution $p(\mathbf{x}_1|\mathcal{I}_1)$ becomes a delta function so that each possible node combination of the selected candidate nodes (with nodes up to $N_{\max}$) is equally likely and node sets including other nodes than the selected ones have zero probability. The marginal posterior distribution of the location of the node sets in the second image is thus

$$\begin{aligned} p(\mathbf{x}_2|\mathcal{I}_{1:2}) &= \int p(\mathbf{x}_{1:2}|\mathcal{I}_{1:2}) d\mathbf{x}_1 \\ &= \int p(\mathbf{x}_2|\mathbf{x}_1, \mathcal{I}_{1:2}) p(\mathbf{x}_1|\mathcal{I}_1) d\mathbf{x}_1 = \sum_s p(\mathbf{x}_2|\mathbf{x}_1^s, \mathcal{I}_{1:2}) \ , \end{aligned} \tag{4.34}$$

where the summing is over all the possible node sets.

4.5.2 Posterior distribution

In the basic method, presented in Section 4.3, the marginal posterior distribution of the node locations in the current image t was contributed by the past images only through the mean Gabor responses; when searching for the corresponding points in images $\mathcal{I}_{1:t}$, the distribution $p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t-1})$ had no effect on where to look for the points in image t. The extended method (Section 4.4) was an improvement in the sense that the prior model depended on the mean of the node locations $\mathbf{x}_{1:t-1}$.

In the full method, the current posterior distribution in image t depends on $\mathbf{x}_{1:t-1}$ and the distribution $p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t-1})$ must be taken into account when matching the image. As in the context of dynamic state-space models (Doucet et al., 2001), each new image is considered a novel observation which recursively updates the joint posterior distribution whose dimension grows after each observation (see Figure 4.1). Therefore, the joint posterior (4.1) is considered in this section. It is a distribution over all the possible node sets and is rewritten here for convenience:

$$p(\mathbf{x}_{1:t}|\mathcal{I}_{1:t}) \propto p(\mathcal{I}_t|\mathbf{x}_{1:t}, \mathcal{I}_{1:t-1}) p(\mathbf{x}_t|\mathbf{x}_{1:t-1}) p(\mathbf{x}_{1:t-1}|\mathcal{I}_{1:t-1}) \ . \tag{4.35}$$

Again, the scale and orientation parameters can be thought of being marginalized out of the posterior distribution of image t.

4.5.3 Likelihood model

The likelihood is built as explained in Section 4.2. To save computation time, the reference jets $J(x_j)$ used in the similarity measure $S_p(J(x_t), J(x_j))$, and thereby in the associative likelihood equation (4.7), are the same for all x_j (that correspond to the node under scrutiny) and this value is computed by averaging $J(x_j)$ over the product of the posterior probability of $\mathbf{x}_j$ and the posterior association probability of the node.

The similarities are computed with Gabor jets having three different frequencies ($2^{-\{2,2.5,3\}}\pi$) and ten orientations ($\{0,...,9\}\pi/10$). Since the proposed framework is orientation and scale invariant, the Gabor responses of the test image are computed with an extended filter bank that contains two additional frequencies on both ends and additional conjugated filter responses so that the likelihoods can be estimated from the neighboring filter responses (see Section 2.5).

4.5.4 Prior model

The prior model is the same as in 4.4.4, except that the posterior association probability depends on the node locations of the past images. Also, the reference shape does not use a point estimate for the past node locations. The reference location of node i is thus

$$\hat{x}^i_{1:t-1} = \frac{\sum_{k=1}^{t-1} P(A^i_k | x^i_{1:k}, \mathcal{I}_{1:k})(x^i_k - M_{x,k})}{\sum_{k=1}^{t-1} P(A^i_k | x^i_{1:k}, \mathcal{I}_{1:k})} , \tag{4.36}$$

where $M_{x,k}$ is computed with equation (4.30).

4.5.5 Particle filtering

The nodes in the test image t are sampled with PMC-like sampling. The algorithm resembles the PMC implementation of Section 4.4 but here the node locations of the particles at the processed images are taken into account. The particle weights are evaluated recursively:

$$w^{(n)}_{1:t} \propto w^{(n)}_{1:t-1} \frac{p(\mathcal{I}_t | \mathbf{x}^{(n)}_{1:t}, \mathcal{I}_{1:t-1}) p(\mathbf{x}^{(n)}_t | \mathbf{x}^{(n)}_{1:t-1})}{q(\mathbf{x}^{(n)}_t | \mathbf{x}^{(n)}_{1:t-1}, \mathcal{I}_{1:t})} , \tag{4.37}$$

where n indexes a particle. The detailed form of the particle Monte Carlo implementation is given in Algorithm 7, where the indexing of the particles have been dropped for clarity and the notation x_i refers to the ith node of the node set $\mathbf{x}_t$. The initialized particles contain components $\{\mathbf{x}^0_{1:t}, s^0, \varphi^0\}$, where $\mathbf{x}^0_{1:t-1}$ are the node locations in the processed images.

The algorithm includes two iterations. The first iteration is mainly intended for getting rid of the worst initializations while in the second iteration the survived hypotheses are improved. The node sets of the particles are modified during the second iteration by removing some nodes whose prior association probability gets too low. The new nodes are selected among such candidate nodes (see Subsection 4.5.6) that are located near the existing nodes. Also, if the number of nodes is below the allowed maximum N_{max}, the node set is expanded by selecting some of the candidate nodes that are located near the convex hull of the existing nodes.

```
1. Initialize
    - Pre-compute responses of the test image with a large filter bank consisting of 7 frequencies $2^{-\{1,1.5,...,4\}}\pi$ and 20 orientations $\{0, \pi/10, ..., 19\pi/10\}$
    - Initialize the particles as explained in Subsection 4.5.6
    - Set $h = 1$.
2. Sample each particle from the proposal distributions
    - If $h = 2$ compute the posterior association probabilities with (4.5) and set $T_i = P(A_t|x_t, \mathcal{I}_{1:t})$, otherwise set $T_i = 0.5 \ \forall i$
    - Set $q(\log[s]|s^{h-1}) = \mathcal{N}(\log[s^{h-1}], \sigma_s^2)$, where $\sigma_s = 2\log[1.1]/(1 + E[T])$
    - Set $q(\varphi|\varphi^{h-1}) = \mathcal{N}(\varphi^{h-1}, \sigma_\varphi^2)$, where $\sigma_\varphi = 10/(1 + E[T])$ (in radians)
    - Sample scale and angle: $\log[s^*] \sim q(\log[s]|s^{h-1})$ , $\varphi^* \sim q(\varphi|\varphi^{h-1})$
    - If $h = 2$ modify the node set as explained in the text
    for $i$ = 1 to $N_{\text{nodes}}$ do
        - Set $q(x_i|\mathbf{x}_t^{h-1}, s^*, \varphi^*) = \mathcal{N}(x_i|m_\mathbf{x} + s^* R(\varphi^*)(x_i^{h-1} - E[\mathbf{x}_t^{h-1}]),\ R(\varphi^*)c_i)$, where $m_\mathbf{x}$ is the $\mathbf{T}$ weighted midpoint of $\mathbf{x}^{h-1}$ and $c_i = \begin{pmatrix} \sigma_{u_i}^2 & 0 \\ 0 & \sigma_{v_i}^2 \end{pmatrix}$, where $\sigma_{u_i} = s^*(\sigma^i_{u,prior})^{\sqrt{2-T_i}}$ and $\sigma_{v_i} = s^*(\sigma^i_{v,prior})^{\sqrt{2-T_i}}$
        - Set $q(\mathcal{I}_t|x_i, \mathbf{x}_{1:t-1}^{h-1}, \mathcal{I}_{1:t-1}, s^*, \varphi^*) = P(A_i)\tilde{L}_i(x) + (1 - P(A_i))\kappa$, where $P(A_i)$ is the prior association probability (4.4) and $\tilde{L}_i(x)$ is the likelihood, with the scale and orientation closest to $s^*$ and $\varphi^*$
        - Set $Z_i = \sum_{x_i \in A} q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*) q(\mathcal{I}_t|x_i, \mathbf{x}_{1:t-1}^{h-1}, \mathcal{I}_{1:t-1}, s^*, \varphi^*)$, where the size of the search window $A$ is $[3\ \ 3]c_i$ pixels around the prior mean
        - Set $q(x_i|\mathbf{x}_{1:t-1}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t}) = \frac{1}{Z_i} q(x_i|\mathbf{x}^{h-1}, s^*, \varphi^*) q(\mathcal{I}_t|x_i, \mathbf{x}_{1:t-1}^{h-1}, \mathcal{I}_{1:t-1}, s^*, \varphi^*)$
        - Sample location (numerically): $x_i^* \sim q(x_i|\mathbf{x}_{1:t-1}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t})$
    end for
3. Compute the particle weights for each particle
    - Compute $w = \frac{p(\mathcal{I}_t|\mathbf{x}^*, \mathbf{x}_{1:t-1}^{h-1}, \mathcal{I}_{1:t-1}, s^*, \varphi^*) p(\mathbf{x}^*|\mathbf{x}_{1:t-1}^{h-1}, s^*, \varphi^*)}{[\prod_{i=1}^{N_{\text{nodes}}} q(x_i|\mathbf{x}_{1:t-1}^{h-1}, s^*, \varphi^*, \mathcal{I}_{1:t})] q(\log[s^*]|s^{h-1}) q(\varphi^*|\varphi^{h-1})}$, where the value of the likelihood has been computed by interpolating between four neighboring filter responses, according to $s^*$ and $\varphi^*$.
    - If $h = 1$, set the particle weight to $\tilde{w} = w$, otherwise set $\tilde{w} = w_{1:t-1}\ w$
4. Resample
    - Resample the particles with replacement according to $\tilde{w}^\alpha$
    - Set the components of the particles $\{\mathbf{x}_{1:t}^h, s^h, \varphi^h\}$ to the resampled values
    - If $h = 1$ set $h = h + 1$ and go to step 2. Otherwise set the particle weight to $w_{1:t} = \tilde{w}^{1-\alpha}$ and finish.
```

Algorithm 7: The particle Monte Carlo implementation. See text for details.

After the first few images of the sequence, the particle representation may contain many different hypotheses about the common object. Figure 4.1 can also be used to depict this; the posterior distribution is represented with three particles, whose weights are proportional to the size of the dots. Sooner or later the weaker hypotheses pass away owing to the resampling and after many images all the particles share the same ancestor.

4.5.6 Initializing the particles

Probably the most problematic part of the sampling scheme is the initialization of the particles, as the object can be located anywhere in the test image with arbitrary scale and orientation. When matching the second image, the particles are initialized in the following manner. For each candidate node, selected in the starting image, the similarity with the second image is computed, using all the possible scales and orientations (there are 3 and 20 of them). The maximum over the image and over all the scales and orientations is chosen. This gives so-called key node which sets the location, scale and orientation for one particle. Each particle has its own key node so the number of particles is also N_p.

When applying the PMC algorithm in the second image, the node sets of the particles are expanded around the key node by picking in the starting image the closest $N_{\max}$ number of nodes among the N_p candidate nodes. These are matched in the second image using the proposal distributions. For each matched node location, the posterior association probability is computed and nodes having this value less than half are discarded from the particle set. Due to the resampling, only node sets with high visual correspondence between the first two images survive. The sampling algorithm is thus somewhat different in the second image than in the further images.

For initializing the particles in the subsequent images a similar procedure is used. The mean value of the Gabor responses of each matched node is estimated from the particles. The key node of each particle is the one with the largest prior association probability. For degenerated particles, different key nodes are used, being those with the largest association probabilities. Some particles may share the same key nodes. Maximum similarity between the key nodes and the next image is again computed to initialize the particles. At this stage, also new reference nodes are selected in the image if the number of different matched nodes is below N_p so that there are always N_p number of candidate nodes available when modifying the current node sets.

The particle initialization is based on the desire that at least one of the key nodes is correct, giving a correct location, scale and orientation. Typically there are many correct key nodes and after the resampling of PMC these hypotheses survive. However, it may happen that none of the key nodes is correct, in which case the image is doomed to be incorrectly matched.

4.5.7 Kernel selection

As more images are processed, the number of kernels in the mixture likelihood increases. This increases the computation time. The idea behind having many kernels in the likelihood is based on a desire of being able to model multiple appearances for a same node. However, there is usually an upper limit for the number of different appearances that certain part of an object possesses. For instance, a human being does or does not wear glasses, resulting in two different appearances for an eye node. Of course the appearance of an eye varies from subject to subject but there probably is redundancy in a mixture representation where all the kernels represent a bare eye of different subjects. Besides, as the area covered by the kernels in the Gabor jet space increases, also background pixels receive higher likelihood values which may more easily lead to false matches.

In order to decrease the redundancy and to compress the object model, a forward selection algorithm (Orr, 1996) is applied to select only the most meaningful kernels from the mixture. In the algorithm, a new kernel representation is constructed which at the beginning does not contain any kernels. The kernel that most decreases the training error is added into the representation. Then, second kernel is added using the same reasoning. In the glass example, the first two kernels would probably be selected in the two groups (with / without glasses) and they would represent the most typical appearances, meaning that they would lie in the middle of the two groups. More kernels are added until the number of selected kernels exceeds a pre-defined threshold, or until some other error measure that is monitored during the process starts to increase. The error measure used here is the generalized cross-validation error (Craven and Wahba, 1978) which can be computed analytically. It estimates how well the mixture model would fit in new data by balancing the fit in the training data and the model complexity; in other words, it tries to solve the eternal dilemma of over / underfitting. Other similar error measures, such as the unbiased estimate of variance and the Bayesian information criterion, exist and could be used as well. Because the error measure based stopping rule is somewhat unreliable, a lower limit can be set on the number of kernels. In the experiments of this thesis, at least three kernels were always selected. After the kernel selection scheme is stopped, the representation should contain only statistically significant kernels, but no more.

4.5.8 Algorithmic form

The full incremental method for matching the objects is depicted in Algorithm 8. The small filter bank contains the frequencies $2^{-\{2,2.5,3\}}\pi$ and the orientations $\{0,...,9\}\pi/10$. The frequencies and orientations in the large filter bank are $2^{-\{1,1.5,...,4\}}\pi$ and $\{0,\pi/10,...,19\pi/10\}$.

```
1. Actions for the starting image:
    (a) Gabor transform the starting image with the small filter bank
    (b) Select $N_p$ number of candidate nodes in the starting image as explained
        in Subsection 4.5.1 and store the Gabor responses at each node
2. Match the next image
    (a) Gabor transform the next image with the large filter bank
    (b) Compute the likelihood as explained in Subsection 4.5.3
    (c) Compress the kernel representation as explained in Subsection 4.5.7
    (d) Sample the posterior distribution of the image with Algorithm 7
3. Compute the parameter values
    (a) For each particle, compute and store the posterior association probabil-
        ities of the nodes with (4.5)
    (b) For each node, estimate and store the mean magnitudes and phases of
        the Gabor responses
    (c) For each particle, update the prior node variance with (4.33)
    (d) For each particle, update the reference shape with (4.36) and (4.30)
4. Go to step 2
```

Algorithm 8: The incremental algorithm for matching the corresponding points with the full method.

4.6 Comparison of the methods

4.6.1 On the similarities and differences of the methods

The Sections 4.3, 4.4, and 4.5 have presented three methods for incrementally matching the corresponding points of an unknown object in a sequence of images. The methods have much in common: The joint likelihood of a node set is a product of the independent node likelihoods, a node likelihood is a mixture of dissociative and associative likelihood which is formed from the phase-sensitive Gabor similarity, the prior is a Gaussian distribution with a diagonal covariance matrix, the prior node association probabilities are updated as a mean of the previous posterior association probabilities, and the posterior distribution of the node locations is sampled with particle Monte Carlo methods. The differences lie in the complexities of the methods. The basic method uses the same set of nodes along the sequence, with a constant variance in the prior distribution, and uses a SMC implementation where the nodes are matched one at a time from conditional distributions, conditioned on the already sampled nodes. The extended method modifies the node set by moving the dissociative nodes inside the convex hull of the current node set so that the area of the node set shrinks along the sequence until it covers only the object. Also, the node variances are updated and the reference node shape is computed as a mixture of previous node shapes. The matching utilizes PMC sampling where the particles are initialized by scanning the image with a rigid node set using few different scales and orientations to find the most probable locations of the object.

In the basic and extended methods, the marginal posterior distribution of the current image depends only on the point estimates of the previous images. A particle is thus a hypothesis of the node locations only in the current image and the particle representations need not be stored after having computed the point estimates. The full method utilizes the recursive formula of the joint posterior in its 'full extent' by considering each new image as an observation which updates the present knowledge about the common object. In the particle representation, each particle can be inferred as a hypothesis about the common object, containing the locations of the nodes and the scale and orientation of the node set in the processed images. The updating of the prior association probabilities, node variances and reference shapes is thus done individually for each particle. The particle Monte Carlo implementation resembles the PMC sampling of the extended method but the weights of the particles in the previous images contribute in computing the current weights. Also, the initialization is based on the maximum similarity values of single nodes instead of the whole node set. Furthermore, it is possible to include full orientation invariance without having a computationally too heavy model.

4.6.2 On the performance of the methods

It may seem that as the complexity of the methods increases from 4.3 to 4.5 also the performance of the systems increases. However, this may not always be true, depending on the nature of the images being matched. The basic method 4.3 assumes the object to appear at nearly constant location, scale and orientation. If this situation holds for the object instances in the images, the starting point is much better for the basic method that for the more complex methods. For the full method, the search space is significantly larger due to the possibility of arbitrary location, orientation and (almost arbitrary) scale which means that the correct mode of the posterior distribution takes a very small portion of the whole parameter space. This complicates the matching because it may either happen that the main mode of the posterior is not the correct mode or the sampling algorithm is unable to locate the globally largest mode. On the other hand, in the extended and full methods the particle initialization is separated from the actual sampling process, making it possible to increase the performance of those methods by using better initialization routines.

4.6.3 On the robustnesses of the methods

The methods differ also in their robustness, that is, the ability to handle statistically abnormal observations. The basic method is expected to be the most robust because its node set is constant so that the prior distribution of the node locations is the same in each image. A mismatched image presents only a small problem in matching upcoming images, especially if the mismatch occurs after a few successful matches so that there are enough likelihood kernels that represent the correct appearances. Actually, the basic method is even expected to handle sequences containing background images (images without the object instance). In addition, although it is in principle unnecessary to carry the background nodes in the node set, they on the other hand provide information about where the object is not.

The extended method is less robust due to the modification of the node set and node variances. If a node set is modified in a mismatched image, the relocated nodes are inevitably selected in the background and matching the later images gets more difficult. For the extended and full methods, each mismatched node also distorts the value of the node variance. The quality of the first two images is most important for the full method since the node sets are established based on these images; if all the particles miss the object, it is almost impossible to find the object in the upcoming images as the node set is then required to hit the target by change. Also, only the most similar nodes are included in the node set which may cause problems. For example, if the object is a human face and the other of the first two faces have sunglasses, the eye nodes are probably not selected although they should be part of the object representation with two different appearances (with / without sunglasses). Although the modification scheme of the sampling algorithm is there to have the missed nodes included later in the sequence, it is unlikely that the node representation will ever cover each part of the object. Hence, there is a danger of overfitting the particle set in the first two images. The level of overfitting, however, can be diminished by increasing the maximum number of nodes and lowering the threshold in the initial node selection scheme, but swinging the pendulum too far the opposite way may also lead to mismatches. As is typical for statistical methods, there is a balance between over- and underfitting that wants to be found. Finally, it can be concluded that the order of the images is vital for each method but less vital for simpler methods.

4.7 Parameter estimation

The three main parameters of the methods, presented in Sections 4.3, 4.4 and 4.5, are the steepness parameter of the likelihood function (β) , the dissociative likelihood (κ) and the Gaussian deviation from the reference shape (σ) which reflects the elasticity of the object. Although the extended and full methods aim to learn the shape parameter σ during the process, they still need an initial value to be used in the second image. This section considers a balance between the likelihood steepness and the shape variance as well as presents a heuristic procedure for estimating

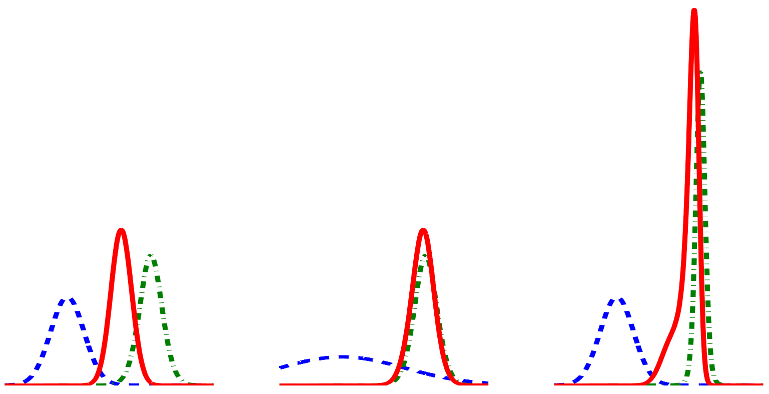


Figure 4.7: An illustrative example of the relation of the standard deviation parameter of a prior function (dashed blue line) and the steepness parameter of a likelihood function (dash-dotted green line). In the leftmost plot, the parameters are small and the unnormalized posterior distribution (solid red line) is between the two distributions. In the middle plot, the prior variance is large enough to let the posterior distribution essentially equal the likelihood function, whereas in the rightmost plot, it is due to the high likelihood steepness that the posterior is again approximately at the same location as the likelihood.

the dissociative likelihood from the data.

4.7.1 Balancing β and σ

The standard deviation of the Gaussian shape model should be large enough to allow instances of objects with different shapes to be matched. In practice this means that the marginal posterior distribution of a node location should have a global maximum at correct location where there — hopefully — is a local likelihood peak. On the other hand, too large variance causes false likelihood peaks to dominate if they happen to be large by magnitude, and in the limit $\sigma \rightarrow \infty$ the global maximum of the total likelihood is the maximum of the posterior. The value of the shape parameter is thus vital to achieve robust object matching.

A 'too' small value of σ can still lead to the correct maximum of the marginal posterior of a node if the steepness of the likelihood is large enough. This simple phenomenon is illustrated in Figure 4.7, in which the parameters are modified in two different ways so that the mode of the posterior would coincide with the mode of the likelihood. The task is, therefore, to find an optimal balance for the shape and steepness parameters. The balance can be estimated by noting that the value of the marginal posterior should be bigger at the correct location x_i^c than at the location expected by the reference shape x_i^r. Let us consider the second image of the sequence and an object node whose prior association probability is taken to be one. By adopting the notation $\mathbf{x} \equiv \mathbf{x}_2, \mathbf{x}' \equiv \mathbf{x}_1, \mathcal{I} \equiv \mathcal{I}_2$ and $\mathcal{I}' \equiv \mathcal{I}_1$ the inequality is

$$\begin{aligned}
& p(x_i^c|\mathbf{x}_{\setminus i},\mathbf{x}',\mathcal{I},\mathcal{I}') > p(x_i^r|\mathbf{x}_{\setminus i},\mathbf{x}',\mathcal{I},\mathcal{I}') \\
\Leftrightarrow\ & p(\mathcal{I}|x_i^c,x_i',\mathcal{I}',V_i)p(x_i^c|\mathbf{x}_{\setminus i},\mathbf{x}') > p(\mathcal{I}|x_i^r,x_i',\mathcal{I}',V_i)p(x_i^r|\mathbf{x}_{\setminus i},\mathbf{x}') \\
\Leftrightarrow\ & \exp\left[\beta S_p(J(x_i^c),J(x_i')) - \frac{(x_i^c-x_i^r)^2}{2\sigma^2}\right] > \exp\left[\beta S_p(J(x_i^r),J(x_i'))\right]
\end{aligned}$$

$$\Leftrightarrow\ \beta\sigma^2 > \frac{(x_i^c-x_i^r)^2}{2(S_p(J(x_i^c),J(x_i')) - S_p(J(x_i^r),J(x_i')))} \qquad (4.38)$$

which shows the reciprocal dependence on the steepness parameter and the variance of the shape model. The larger the distance between the expected and correct location, and the smaller the similarity difference between the two, the larger the product of the parameters must be. Having the prior association probability less than unity leads to an analytically intractable equation but it can be inferred that the right hand side of the inequality (4.38) increases (slightly, depending on the value of $S_p(J(x_i^c),J(x_i'))$), thereby demanding higher value for $\beta\sigma^2$.

Equation (4.38) illustrates the limit where the correct location is as probable posterior-wise as the reference location — an open question is how much more probable the correct location should be. An arbitrary large value for $\beta\sigma^2$ makes the shape model useless as noted before. On the other hand, if the correct location is more probable than the reference location, the maximum of the posterior is probably still in the middle of the correct and reference locations. Equation (4.38) is exploited to validate the parameters in Chapter 5.

4.7.2 Setting the threshold parameter κ

Especially for the performance of the basic method 4.3 and the extended method 4.4 the value of the threshold parameter κ has a great effect as it should be set to such a value that the nodes are associated correctly. Also, it affects the matching of the node set as the joint likelihood is a product of independent node likelihoods; if the similarity of (any of) the background nodes of the node set at the correct location is very low, a too small value for κ leads to the main mode of the posterior being in a false position where each node fits somehow. On the other hand, when the value of κ increases, the mixture likelihood becomes approximately a constant which again leads to false matches. This is illustrated in Figure 4.8 which considers an artificial example where the reference node set consists of three object nodes and three background nodes. In the test image, the object nodes fit well at the correct location and the background nodes possess zero likelihood values — this is as expected. However, a false location is also imagined, representing a local maximum of the posterior in which all the nodes fit somehow — this, too, is common in object matching. Having a (very) low value for κ causes the false likelihood peak to dominate but then again the ratio decreases towards unity when κ

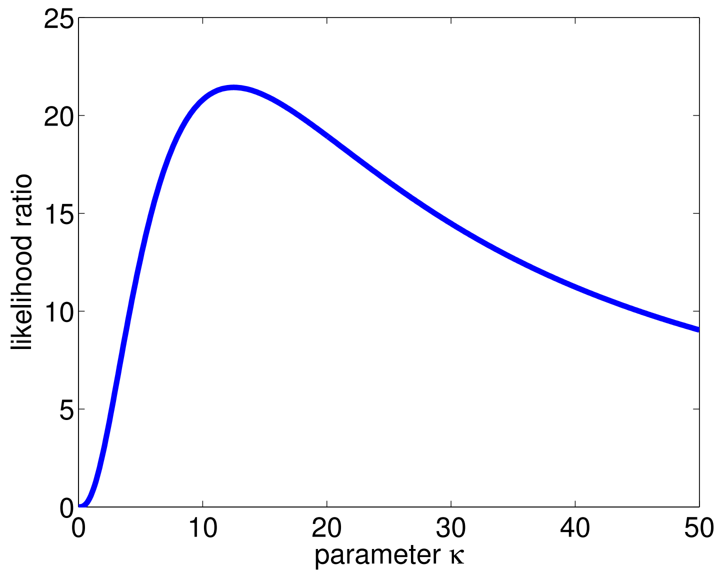


Figure 4.8: An artificial example of the ratio of joint likelihoods of a node set, consisting of three object nodes and three background nodes, at correct and false locations. The prior association probabilities are set to half. For the correct location, the object likelihoods are $\{100, 100, 100\}$ and the background likelihoods $\{0, 0, 0\}$. The likelihood values at the false location are all set to 10.

approaches infinity and an intermediate value gives the highest ratio. Because the matching is easiest with high ratios of the posterior masses, fixing κ is an essential problem. For the full method, the value of κ is perhaps less meaningful as the node set is supposed to include object nodes only; however, a positive value for κ enables occluded objects or objects with abnormal appearance to be matched.

The fixing of κ is based on decision theory according to which an expected utility of making a certain decision is a weighted average of the utilities for different outcomes with the decision, weights being the probabilities for those outcomes (Bishop, 2006). The optimal decision is the one that maximizes the expected utility. An analytical formula for κ can be obtained by considering the expected utility of the optimal decision for node association in which the object nodes, and only them, are associated as object nodes. This expected utility is maximized w.r.t. the parameter κ while keeping parameters β and σ fixed.

Let us assume that the utility is one when making the correct association and zero for the opposite case. By taking the negation of the expected utility and by letting A_i be the association status of the ith node of image t we end up with the (sum) association error:

$$E_s(\beta, \kappa, \sigma) = -\sum_{i \in \mathcal{F}} P(A_i|\mathcal{I}_{1:t}) - \sum_{i \in \mathcal{B}} [1 - P(A_i|\mathcal{I}_{1:t})] \ , \tag{4.39}$$

where $\mathcal{F}$ denotes the object nodes and $\mathcal{B}$ the background nodes. The smallest possible value of E_s is the total number of nodes with a negative sign. Another option would be to investigate a product based error

$$E_p(\beta, \kappa, \sigma) = -\prod_{i \in \mathcal{F}} P(A_i|\mathcal{I}_{1:t}) \prod_{i \in \mathcal{B}} [1 - P(A_i|\mathcal{I}_{1:t})] \tag{4.40}$$

which is directly interpretable as the negative of the total probability of having the object nodes, and only them, to be associated with the object. Using the product of individual association errors is highly sensitive to single errors as any of the object nodes surely associated as a background node, or vice versa, leads to the total error being zero. It is more reasonable to use an error measure that depends on the average error than the worse-case error and as the aim of this subsection is to find some practical (rather than theoretically flawless) rules for setting the parameters, the sum error (4.39) will be used from now on.

Equation (4.39) deals with the posterior node association probabilities from which the node locations are integrated out. To estimate the marginal association probabilities, $\mathbf{x}_t$ is divided into two subsets, x_i and $\mathbf{x}_{-\mathbf{i}}$, where the latter contains all the indices other than i so that $p(\mathbf{x}|\mathbf{x}', \mathcal{I}_{1:t}) = p(x_i|\mathbf{x}_{-\mathbf{i}}, \mathbf{x}', \mathcal{I}_{1:t}) p(\mathbf{x}_{-\mathbf{i}}|\mathbf{x}', \mathcal{I}_{1:t})$. Then, let the distribution $p(\mathbf{x}_{-\mathbf{i}}|\mathbf{x}', \mathcal{I}_{1:t})$ be a delta function at its mean value, $\hat{\mathbf{x}}_{-\mathbf{i}}$. This assumption ignores the posterior variance but with large β the posterior is essentially unimodal so the variance in the distribution of $\mathbf{x}_{-\mathbf{i}}$ has little effect on the marginal posterior distribution $p(x_i|\mathbf{x}_{-\mathbf{i}}, \mathbf{x}', \mathcal{I}_{1:t})$. The integral in equation (4.16) then reduces to an integration over the ith component x_i:

$$\begin{aligned} P(A_i|\mathcal{I}_{1:t}) &= \int p(A_i|\mathbf{x}, \mathcal{I}_{1:t}) p(x_i|\mathbf{x}_{-\mathbf{i}}, \mathcal{I}_{1:t}) p(\mathbf{x}_{-\mathbf{i}}|\mathcal{I}_{1:t}) d\mathbf{x} && (4.41) \\ &\simeq \frac{\int P(A_i) p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, A_i) p(x_i|\hat{\mathbf{x}}_{-\mathbf{i}}) dx_i}{\int P(A_i) p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, A_i) p(x_i|\hat{\mathbf{x}}_{-\mathbf{i}}) dx_i + (1 - P(A_i))\kappa} && (4.42) \\ &= \frac{P(A_i)\gamma_i}{P(A_i)\gamma_i + (1 - P(A_i))\kappa}\, , && (4.43) \end{aligned}$$

where $P(A_i)$ is the prior association probability and γ_i denotes the integral of the product of the likelihood and shape function which can be thought of as a smoothed local average of the likelihood around the mean of the Gaussian distribution:

$$\gamma_i \equiv \int p(\mathcal{I}_t|x_i, \mathcal{I}_{1:t-1}, A_i) p(x_i|\hat{\mathbf{x}}_{-\mathbf{i}}) dx_i\, . \tag{4.44}$$

With this notation equation (4.39) is rewritten as

$$E_s(\beta, \kappa, \sigma) = -\sum_{i \in \mathcal{F}} \frac{P(A_i)\gamma_i}{P(A_i)\gamma_i + P(\overline{A_i})\kappa} - \sum_{i \in \mathcal{B}} \frac{P(\overline{A_i})\kappa}{P(A_i)\gamma_i + P(\overline{A_i})\kappa} \tag{4.45}$$

Let us next calculate a value for κ that minimizes the error (4.45). For arbitrary γ values the problem is unattractive to solve analytically. However, (4.45) can be minimized — in the usual way by differentiating it with respect to κ and setting the results to zero — with relatively little effort if all the γ values and prior association probabilities of object nodes are assumed to have the same value, and likewise for the background nodes. Naturally, this assumption is not in line with reality as there is some variation in the masses of the likelihood peaks; that variation must be ignored in this analysis. Let γ_F and γ_B denote the γ values of object and background nodes and let P_F and P_B denote the prior association probabilities of object and background nodes. The value of κ that minimizes E_s is then

$$\tilde{\kappa} \approx \sqrt{\frac{N_B P_B P_F \gamma_F \gamma_B}{N_F(1-P_F)(1-P_B)}} , \tag{4.46}$$

where N_F and N_B denote the number of object and background nodes and the approximation holds when $\beta \gtrsim 1$ and $\gamma_F > \gamma_B$. By setting the prior association probabilities to half the above estimate reduces to

$$\tilde{\kappa} = \sqrt{\frac{N_B}{N_F}\gamma_F \gamma_B} . \tag{4.47}$$

Interestingly, having minimized E_p under the same assumptions as above would also have lead to estimate (4.47).

It is worthwhile to investigate the logarithm of 4.47:

$$\log(\tilde{\kappa}) = \frac{1}{2}\left[\log\left(\frac{N_B}{N_F}\right) + \log(\gamma_F) + \log(\gamma_B)\right] . \tag{4.48}$$

The result seems reasonable: with an equal number of object and background nodes, the logarithm of an optimal threshold is the mean of the logarithms of background and object γ values, being the most optimal way to separate the nodes whose γ values are spread into two groups with zero in-group variance. When the number of object and background nodes differ, a small bias is added.

The formula 4.47 is implemented in the methods 4.3 - 4.5 in different ways. In the basic method, the second image is matched with a constant value $\kappa = \exp(0.5\beta)$. After the match, the identities of object and background nodes are estimated by assessing the nodes whose similarities exceed the mean similarity value to be object nodes. The mean values of both groups are taken and inserted into equation 4.47. It is believed that integrating over the prior distribution has little effect when using high value for β, as is used in the experiments. $\tilde{\kappa}$ is computed for each sample and the mean is taken. This estimation for κ is used for matching the third image from which the optimal κ is again estimated. For the fourth image, the median value of the two estimations is used, and so worth. For the extended method, the first PMC iteration always uses value $\kappa = \exp(0.5\beta)$ whereas the following iterations estimate $\tilde{\kappa}$ from the γ values at the sampled locations. The observed γ values are split in two groups based on the mean logarithm and the median values are used for γ_F and γ_B. For the third and subsequent images, the mean of the so far estimated κ values are taken. With the full method, a value $\kappa = \exp(0.1\beta)$ is used in the second image. In the later images, $\tilde{\kappa}$ is simply taken to be the square root of the median of the γ values, averaged over the samples.

Chapter 5

Evaluation of performance

5.1 Introduction

The purpose of the methods, presented in this thesis, is to locate the corresponding points of an object by sequentially handling the images in which the object appears. The promises of the methods can be demonstrated in two different tasks. In the first one, the pixel-wise matching errors between the true corresponding points and the points proposed by the methods are evaluated. The second task measures the ability of the methods to detect whether there is an instance of the learned object in an unseen test image or not.

In this chapter, the experimental results of the methods are given by measuring both the matching errors and detection errors. The errors are compared with other published methods. Before presenting the numerical results, some qualitative results are illustrated in order to give an intuition of the performance of the methods. Also, the used image databases and the parameter values are presented. The three different methods of Sections 4.3, 4.4, and 4.5 will be — somewhat unimaginatively — referred to as M1, M2, and M3.

5.2 Data sets and matching details

To assess the performance of the proposed methods, six data sets were used, four of which are publicly available while the two others consist of private digital camera images. The public databases are the IMM-DTU database (Stegmann, 2002), the Bio-Id database (Jesorsky et al., 2001), the Caltech faces database (Fergus et al., 2003), and the butterfly database (Lazebnik et al., 2004). The digital camera databases contain images of a dog and a similar traffic sign. The DTU images and the dog and sign images are used in evaluating the matching errors, whereas the other images, together with the dog images, are used in detection tasks. Illustrative examples and the sizes of each database are given in the Appendix. In the matching experiments, image sizes were 240×320 and in the detection experiments the vertical size was downscaled to 200 for memory reasons.

Table 5.1: The percentiles for the positive values of λ (equation (5.1)) for the Bio-Id database and Bio-Id* subset, which consists of images from the Bio-Id having the same person.

Database	25 %	50 %	75 %
Bio-Id	19	39	89
Bio-Id*	12	22	51

The Euclidean matching errors of DTU are compared with three published batch methods that have reported the matching errors for these images. They are the Elastic Bunch Graph Matching method of Wiskott et al. (1997), Active Appearance Model implementation of Stegmann (2002) and the Bayesian Object Matching method of Tamminen and Lampinen (2006). In addition to being batch methods, these methods utilize manual annotations to learn the object model, so the situation is much more difficult for the proposed methods. As Rolf Würtz has kindly provided the code for the EBGM method, its performance could be tested also with the other two matching databases, dogs and signs. The detection results of the butterfly images are compared with the batch method of Lazebnik et al. (2004) and the detection results of Caltech face images are compared with several other published methods.

5.3 Estimating $\beta\sigma^2$

In Subsection 4.7.1, it was deduced that in order to achieve good performance, the product of the likelihood steepness parameter β and the variance of the shape model σ^2 should exceed a threshold. The inequality 4.38 is rewritten here for convenience, using slightly different notation:

$$\beta\sigma^2 > \frac{(x_c - x_r)^2}{2(S(J(x_c), J(x')) - S(J(x_r), J(x')))} \equiv \lambda \,, \tag{5.1}$$

where x' is the location of a node in the starting image, x_c is the correct location in the second image, and x_r is the reference location in the second image, that is, the mean of the prior distribution (the case with many training images is more complicated).

In order to exploit the inequality (5.1) the value of its right hand side, denoted as λ, is estimated using the Bio-Id database. The images on the database possess manual annotations on 20 fiducial locations of a human face (these annotations were not utilized in the actual matching tasks). Random pairs of images were drawn from the database and the 20 nodes of the first image were fitted to the second image using the Procrustes method. The correct locations in the second image are taken to be the annotated locations and the reference locations are taken to be the transfered locations. The phase-sensitive similarities and distances were computed and inserted in equation (5.1). This was repeated 100 times. The statistical summaries of positive λ values are given in Table 5.1. Negative λ values are a result of having a poorer Gabor similarity at the annotated location than at the reference location; such nodes pairs are useless for this analysis since they are inevitably matched inaccurately. In addition to using all the Bio-Id images, a subset of the database was extracted that consists of images of the same certain person for which the same analysis was carried out. The subset is denoted as Bio-Id* and the corresponding entries are also found in Table 5.1. In line with intuition, the values are smaller for the images containing the same person as the deformation of the object is smaller and as the differences in the Gabor similarities are higher due to the similarities between the annotations being close to one.

The inequality (5.1) and the values of Table 5.1 should be interpreted as rough guides. One problem is the denominator $S(x_c, x') - S(x_r, x')$. The smaller it is, the larger the product $\beta\sigma^2$ must be to have the marginal posterior higher at the correct location than at the reference location. However, there is a point beyond which it is unnecessary to go; if the difference is, say, $1/1000$, it can be concluded that such a node will not match correctly unless the prior variance is unreasonably large. Therefore, the inference should be based only on the reasonable values of λ, such as the 75% smallest values. As was discussed in Subsection 4.7.1, an upper limit for $\beta\sigma^2$ exists but is unavailable. It should also be noted that the object instances of the Bio-Id database vary a lot, both in appearance and shape, whereas the object instances of the Bio-Id* database are almost identical, and the images that the proposed methods can handle are expected to fall in between the two extremes. Based on the available information and common sense, it can be concluded that the value of $\beta\sigma^2$ should clearly exceed one hundred but fixing it to more than one thousand would probably be an exaggeration. Also, as the approximations that were made in the methods were based on having a sharp-peaked likelihood, a combination of large β - small σ is preferred over the opposite case.

5.4 The parameter values

The parameters that probably have the most effect on the performance are the likelihood steepness parameter β, the deviation of the shape model σ and the dissociative likelihood κ. The relation between β and σ was discussed in Subsection 4.7.1 and Section 5.3, and a heuristic procedure for adopting κ using the observed data was presented in Subsection 4.7.2. In the experiments, the likelihood steepness was set to $\beta = 50$ in all the three methods. The node variance in M1 was set to 20 pixels, whereas the hyperparameters in M2 were set to $\nu = 5$ and $s^2 = 10$, corresponding to an initial node variance of 7.1 pixels, and the hyperparameters in M3 was set to $\nu = 15$ and $s^2 = 5$, resulting in an even tighter initial node variance of about 4.4 pixels. Since the node variance in M1 is not updated, it is possibly safer to fix it to a somewhat large value (for M1, $\beta\sigma^2 = 1000$). Also, because the M2 and especially M3 methods are less robust, it is better to use a rather rigid node set to prevent the 'wandering of the nodes' phenomenon, discussed later in this chapter.

The other parameters of the models are the number of nodes and the parameters of the Gabor filter bank and Monte Carlo sampling methods. The number of nodes for M1 and M2 was 30, being a result of using a grid size of 5×6 in the starting image. In M2, the threshold for removing the nodes was $P(A) = 0.24$ (preventing the node set to be modified yet in the second image) and the maximum number of nodes to be modified in a single image was set to six. In M3, the total number of nodes was $N_p = 150$ (except when otherwise noted) and the maximum number of nodes for a node set was $N_{max} = N_p/5$ (which is 30 for $N_p = 150$). The methods seem to be quite insensitive to the chosen Gabor filters, as long as there are 'enough' different orientations and the frequencies span a 'reasonable' range. The standard deviation of the Gaussian part of the Gabor filter in all three methods was set to π. In M1, 3 frequencies were used ($\sqrt{2}\pi/4$, $\sqrt{2}\pi/8$ and $\sqrt{2}\pi/16$) and 6 orientations $(0, \pi/6, ..., 5\pi/6)$. In M2 and M3, a larger set of base frequencies were used to interpolate the arbitrary scale. Five base frequencies were used in M2 ($\sqrt{2}\pi/[2, 4, 8, 16, 32]$) and seven in M3 ($\pi/2^{[1,1.5,...,3.5,4]}$). The orientations for M2 were the same as in M1. In M3 there were 10 different orientations ($[0, ..., 9]\pi/10$).

Finally, taking that the actual model parameters have been set to 'good' values, so that the main mode of the posterior is at the correct location, it is up to the sampling method to find this main mode, that is, to represent the posterior with good accuracy. In practice, this means a careful implementation of the particle Monte Carlo methods and a large enough number of particles. Detailed implementations of the sampling algorithms of the three models are presented in Chapter 4. In M1, the number of SMC particles was 400 in the beginning and it was reduced by one fourth at each iteration until it reached 50. The Langevin equation was iterated 10 times with leapfrog stepsize $\epsilon = 1$ and the exponent of the weight was $\alpha = 1/3$. In M2, the number of particles was 150 and the maximum number of PMC iterations was 5. In M3, the number of particles was $N_p = 150$ (unless otherwise noted) and the exponent of the weight was $\alpha = 1/\beta (= 1/50)$.

5.5 Qualitative results

Before giving numerical errors, some qualitative results of matching a sequence of eight dog images with each method are given in this section. Inspecting the results should facilitate to understand the performance and differences between the methods.

Figure 5.1 shows the matching results of the M1 method. The object nodes (located in nose, eyes etc.) seem to be quite accurately matched in each image and their association probabilities evolve to unity along the sequence, as can also be seen from the bottom left panel. For the nodes located near the edge of the object the evolving is less straightforward due to the affect of the background and the method is less sure whether these nodes are part of the object. Then again, the

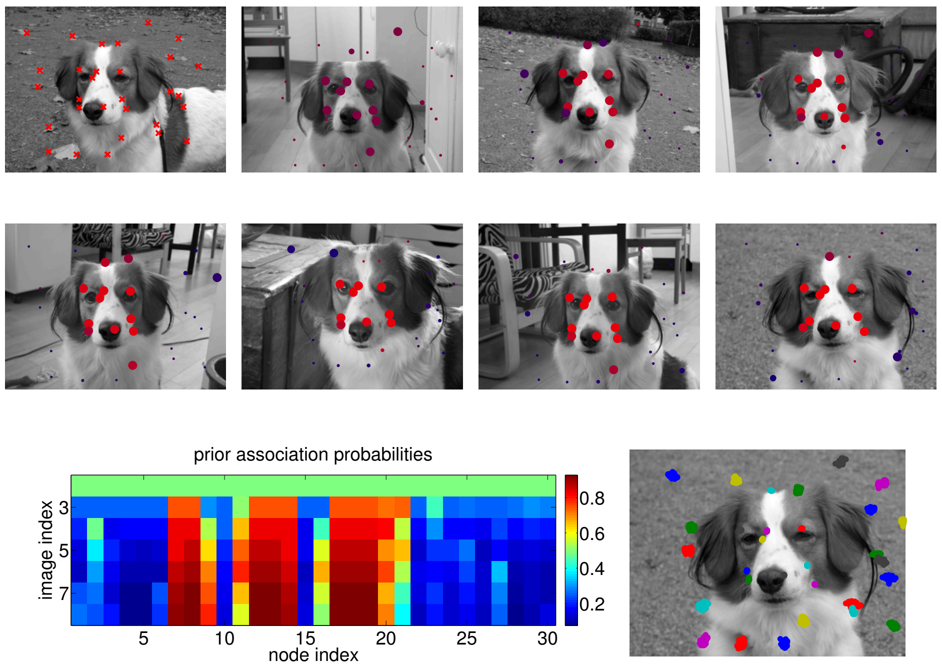


Figure 5.1: Matching results of the M1 method. Sequence proceeds from left to right and top to bottom. The dots depict the Monte Carlo estimate of the posterior mean of the node sets. The red values of the dots are proportional to the prior association probabilities of the nodes and the sizes of the dots are proportional to the posterior association probabilities of the nodes. The evolution of the prior association probabilities for each node and image are given at the bottom left panel and the SMC particles of the eighth image are shown on the bottom right panel.

background nodes are easily identified by the method. The panel on the bottom right showing the eighth image of the sequence with the SMC particles superimposed reveals how the uncertainty about the location of the nodes is much less for the object nodes than for the background nodes. An object model could be formed using only the nodes whose prior association probability exceeds, for instance, 0.9 in the end.

In Figure 5.2, the result of matching the eight dog images with the M2 method is shown. The object nodes are again accurately matched. The relocation of the nodes whose prior association probability goes under a threshold of 0.24 causes the number of object nodes to increase along the sequence and at the end of the sequence the proportion of the object nodes in the node set is larger than when using method M1. The matrix of prior association probabilities reveals how the probabilities typically start to grow after the relocation of the nodes, indicating that the nodes were reselected inside the object boundaries. The panel on the bottom right shows how the variances of the shape model update during the sequence. The values of the matrix are the Euclidean standard deviations ($\sigma = \sqrt{\sigma_x^2 + \sigma_y^2}$) of the

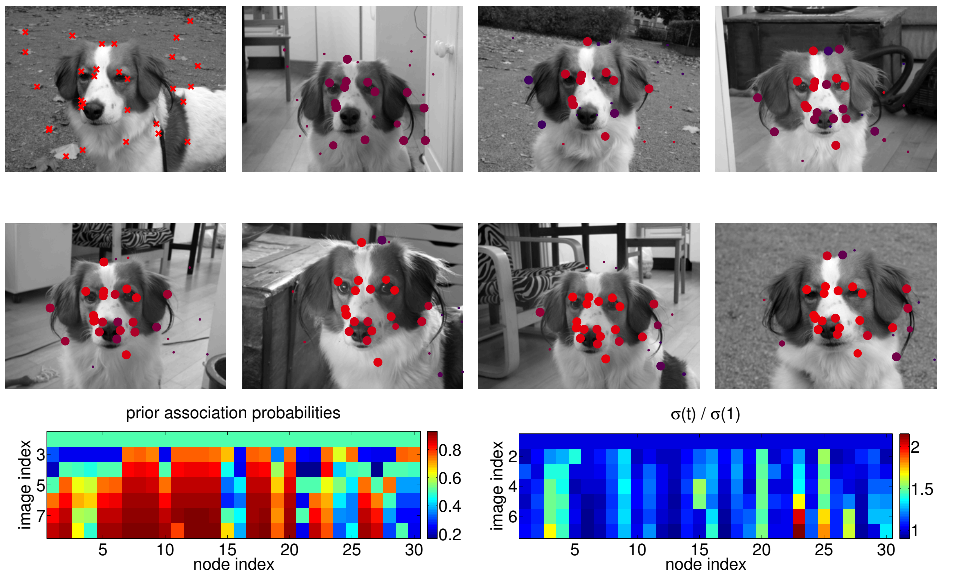


Figure 5.2: Matching results of the M2 method. See the caption of Figure 5.1 for details. Bottom right panel shows the progression of the standard deviations of the shape model, compared with the initial value.

tth image, divided by the initial deviation $\sigma(1)$, for each node. For most nodes, this figure stays close to unity. For the nodes on the edge of the object, there is more variance in their locations, again due to the disturbing background.

The matching results of the M3 method are illustrated in Figure 5.3. The method aims at establishing the correspondence already between the first two images and seems to succeed as there are no background nodes in the node set, apart from the few nodes above the head. Since each part of the object is clearly visible in each image, the posterior association probabilities are unities in each image, except for one of the nodes above the head in the seventh image, and the prior association probabilities are close to one in the end. The first two images of the sequence are shown at the bottom of the figure enlarged, showing also three distinct modes of the posterior distribution which the method has found. The particle representation contains three different reference node sets, which are shown in the left image with different colors and markers. For perceptional purposes, also the convex hulls of the node sets are plotted. In the right image, the average locations of the particles with the corresponding reference node sets are depicted in a similar fashion. The average weight of the node set shown in blue crosses is significantly larger than the weights of the other two node sets, which is why this node set in practice equals the mean value of the node locations, shown in the smaller panels. It is also this node set that alone survives the resampling after observing the third image.

In Figure 5.4 the images that were shown in Chapter 1 are matched with method M3. The nodes have been indexed with numbers to illustrate the identities of the

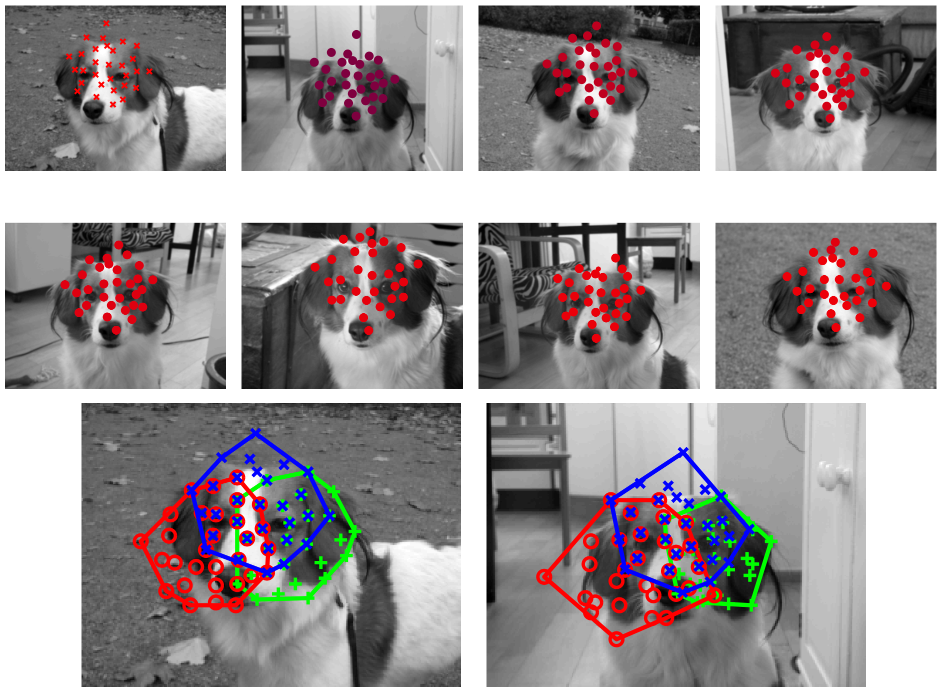


Figure 5.3: Matching results of the M3 method. See the caption of Figure 5.1 for details. The lowest panels show three distinct modes of the posterior distribution of the first two images with different colors.

nodes. Despite the rotation and scale differences of the object instances, the method has been able to locate the common object. This is verified in the right column by the extracted image areas which obviously are similar. Also, almost all the node association probabilities are 100%.

5.6 Matching errors

This section presents the matching errors, that is, the Euclidean distances between the matched locations and the manually annotated correct locations. For evaluating these errors, different image sequences of ten images were matched. The images were randomly chosen from the databases. The point-to-point matching errors were assessed by going through the matched sequences and annotating manually in the next image the correct locations of the corresponding points of the object nodes. For the M3 method, all the nodes are assumed to be object nodes, whereas for the M1 and M2 methods, the nodes whose prior association probability exceeds a threshold at the end of the sequence are interpreted as object nodes. The weighted Euclidean distances between the annotated locations and the locations set by the proposed methods are measured, weights being the posterior association probabilities of the nodes. Hence, the value of the threshold for selecting the object nodes

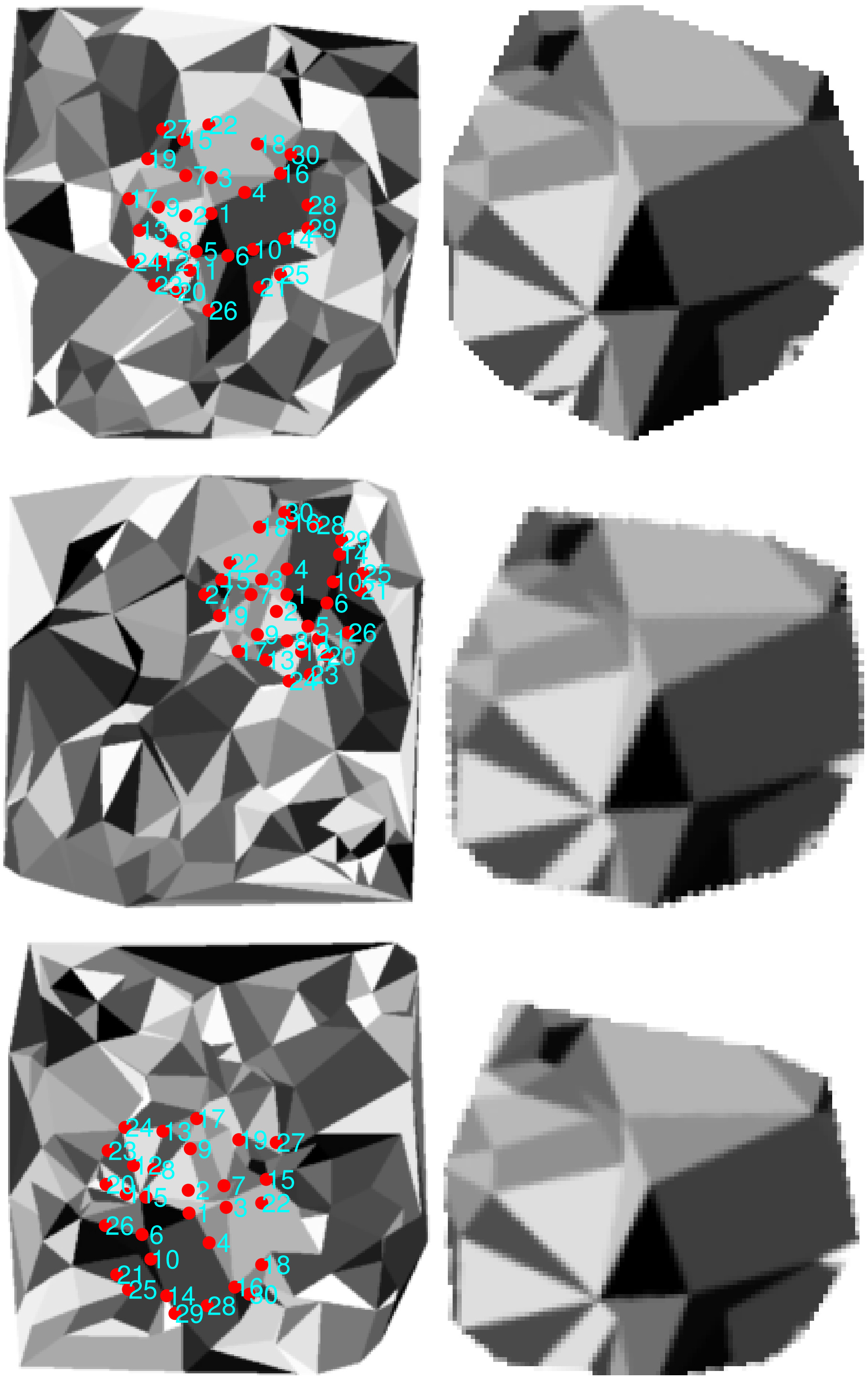


Figure 5.4: Left column: Matching results of the example images using method M3. Right column: The areas extracted from the corresponding images by cropping the area according to the convex hull of the matched points and rotating the area according to the mean posterior orientation angle.

Table 5.2: The statistics of the matching errors of 50 dog image sequences for different methods. N_{nodes} refers to the number of object nodes used to compute the error. For the EBGM method, N refers to the number of training images. The rightmost column reveals the average CPU time in minutes.

Method	Median	Mean	Std	$E[N_{nodes}]$	Time (min.)
M1	5.4	5.6	1.7	8.0	24.6
M2	4.2	5.7	4.8	7.6	23.1
M3	5.8	6.1	1.7	30	57.6
EBGM ($N = 1$)	7.1	16.5	22.2	12	
EBGM ($N = 5$)	3.2	5.3	11.3	12	
EBGM ($N = 9$)	3.2	7.7	16.1	12	

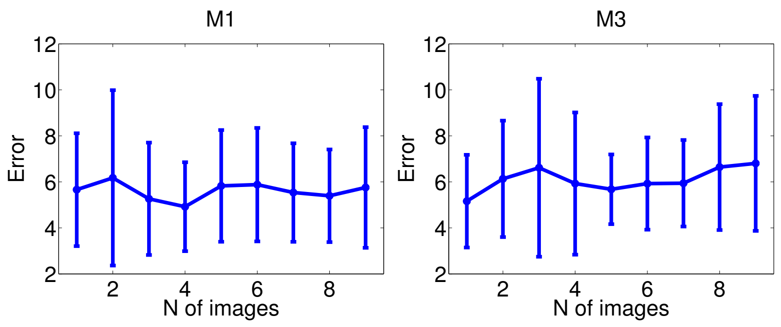


Figure 5.5: The mean errors and standard deviations for dog images as function of the number of processed images for the methods M1 and M3.

is not that important as the nodes typically have low posterior association probability when they are inaccurately matched. A relatively high threshold of 0.9 was selected in order to have a low number of object nodes and to reduce the manual work. The errors were averaged over the images in the sequence. Because the fallacious matches — that is, the matches that converged to an incorrect mode — have high impact on the mean values, the median values give more insight into the matching accuracies and the mean values and the standard deviations of the ability to find the correct mode. It should also be noted that a manual annotation is not precise and hence carries an (unknown) extra error into the analysis.

The results of 50 dog image sequences are given in Table 5.2. The images were the same for each method to yield a fair comparison. The studied methods seem to do an approximately equal job in finding the corresponding points. A closer look at the statistics indicates that the results of the M1 and M3 methods contain fewer mismatches than the method M2 which on the other hand matches more accurately when it finds the target. This is revealed by the fact that for methods M1 and M3 the median and mean values almost coincide and the standard deviations are small, whereas for method M2 the median is clearly less than the mean and the variance of the errors is large, referring to a skewed distribution. A reason

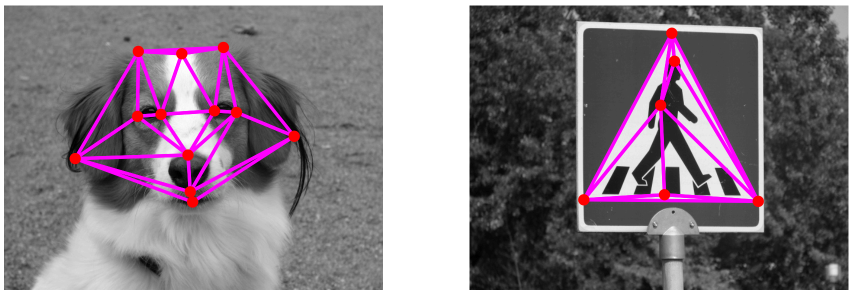


Figure 5.6: An example of the pre-annotations and edges of the dog and sign images used by the EBGM method.

for this might be that the global move based initialization step of M2 finds only an approximative position from which the convergence to the correct mode sometimes fails, even though some of the nodes may be correctly matched. The slightly better accuracy of the M2 method might be due to the Langevin Monte Carlo step that M2 utilizes at the end of the sampling procedure. Also, the number of manual annotations needed to evaluate the error for M3 was huge ($30 \times 9 \times 50 = 13500$ clicks of the mouse) and the boredom of the job possibly resulted in somewhat lower annotation precision which increases the errors. The computation time (in Matlab) of the method M3 is over twice as much as of the other two methods, due to the full rotation invariance and larger number of candidate nodes. On the other hand, the M3 method uses approximately four times as many object nodes as the other two methods and thus it can be thought of carrying more information of the object. The computational requirements are dealt more extensively in Section 5.8.

Figure 5.5 reveals how the matching errors of methods M1 and M3 behave during the sequence. The errors seem to stay at approximately constant values although for M3 the errors — somewhat surprisingly — even seem to increase a bit. However, the fact that the highest matching accuracy with M3 is at the second image is understandable since the method establishes the object nodes by finding the most similar points between the first two images — if some parts of the object have different appearances, they are not selected in the node set.

For the EBGM method, all the images were first annotated on the 12 landmark locations of the object, such as eyes and nose. The edges between the nodes, required for the method, were defined as the Delaunay triangulations of the annotated locations. An example of the annotations and triangulations are presented in the left panel of Figure 5.6. The orientations in the Gabor filter bank were $(0, \pi/6, ..., 5\pi/6)$, like in the M1 and M2 methods, and the frequencies were $\pi/4, \pi/8$, and $\pi/16$. These parameters were used by Tamminen (2005) who found them working well in his matching tests. Compared with the frequencies of methods M1 and M2, these frequencies are smaller by the amount of $\sqrt{2}$, and compared with the frequencies of method M3, the frequencies are the same except $\pi/16$ is replaced by $\pi/2^{2.5}$. According to the experience, these differences in the frequencies

Table 5.3: The statistics of the matching errors of 100 sign image sequences for different methods. N_{nodes} refers to the number of object nodes used to compute the error. For the EBGM method, N refers to the number of training images.

Method	Median	Mean	Std	$E[N_{nodes}]$
M2	1.7	3.2	11.5	2.8
EBGM ($N = 1$)	16.9	31.9	36.0	6
EBGM ($N = 5$)	11.6	20.1	26.8	6
EBGM ($N = 9$)	6.3	9.5	12.3	6

should be small enough to have a high influence on the similarities.

The object model was learned using different number of training images (1, 5 and 9) that were the same as in the sequences used by the proposed methods. The learned object model was then matched in the test image, being the following image in the sequence. EBGM method seems to perform at most as well as the proposed methods, despite that it trains the object model using N number of annotated training images simultaneously. When EBGM finds the correct location, it seems to match accurately but this happens more rarely than with the proposed methods. Even with nine training images the EBGM method is sometimes unable to locate the object in a test image. Hence, it can be concluded that all the methods studied in this thesis perform well for the dog database. EBGM method is implemented in C language making it thus much faster (the CPU times were not recorded) compared with the Matlab implementations of the proposed methods.

The traffic sign images were matched with M2 due to its good accuracy and better invariance w.r.t. scale and transformation compared with M1 (evaluating the errors for M3 is too exhaustive to be done for another database). The results of matching 100 sequences with M2 and EBGM are tabulated in Table 5.3. The extremely small median value is again an indication of the precise matching capabilities of M2. The pre-annotations used by EBGM are shown in the right panel of Figure 5.6. Although the sign images are easy to match as the traffic signs are stiff objects with little deformations, EBGM often fails to locate the object. It might be that six nodes is too few for the EBGM method to achieve robust matching. Another reason could be the scale changes which might be too large for the basic version of EBGM.

The results of matching IMM-DTU images with M1 are given in Table 5.4. The point-to-point errors of M1 are larger than the reference errors, due to several reasons. M1 learns the object model recursively from un-annotated images starting from one image whereas the reference methods process simultaneously 36 images, with 58 pre-annotations in each, to learn the object model. The 58 pre-annotations are informative as they are located on easily defined landmarks — eyebrows, eyes, nose, mouth and jaw — whereas the object nodes of M1 are often on the forehead or top of the head, the exact location of which is more difficult to determine due to the variation in people's haircuts (see Figure 5.7). The background in the im-

Table 5.4: The matching errors of the IMM-DTU images for different methods. The number of image sequences matched by M1 was 50. N_{nodes} refers to the number of object nodes used to compute the error. BOM refers to the Bayesian Object Matching method of Tamminen and Lampinen (2006). The results of EBGM were taken from (Tamminen and Lampinen, 2006). The reference methods utilize 36 training images.

	M1	EBGM	AAM	BOM
Mean $\pm$ std	6.0 ± 4.1	3.08 ± 0.88	2.86 ± 0.61	2.76 ± 0.73
$E[N_{nodes}]$	9.2	58	58	58

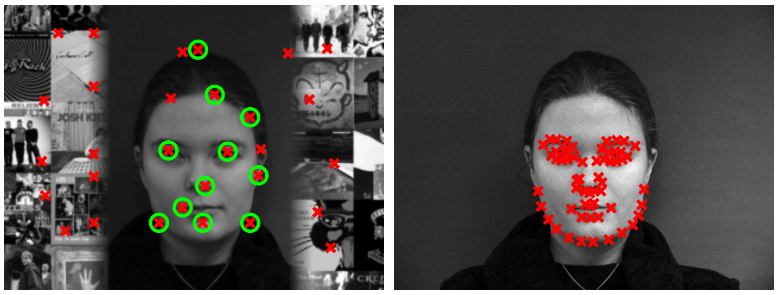


Figure 5.7: Left: the starting image of a sequence and the nodes selected by M1. The green circles indicate the object nodes that contribute to the error. Right: the pre-annotations used by the reference methods.

ages used by the reference methods is monochromatic whereas the backgrounds of the images used here were filled with parts of other images to complicate the matching. It should be noted that although for a human observer the added background textures form an evident pattern, the M1 method is based only on the node points with no segment analysis. Therefore, the regularly shaped background patterns give no advantage to the method; on the contrary, the background segment boundaries appearing in the same locations in each image complicate the matching as they produce false candidates for object nodes. Moreover, the reference experiments (except those of AAM) are conducted on the images whose size is double to that used by M1; due to the discarded information, the errors are likely to grow non-linearly with increasing image size (of course, halved errors for those methods are reported in Table 5.4). Hence, it can be concluded that the proposed method performs again well in locating the corresponding points of the unknown object in novel images.

5.7 Detection errors

The detection errors were computed by matching the learned object model in novel test images, half of which contain the object and half of which are random background images. The summary statistics of the errors are generated by using the receiver operating characteristic (ROC) curve. A ROC curve is obtained by varying a threshold for detection and for each threshold plotting the true positive rate versus the false positive rate. For a random classifier on average, the true positive rate equals the false positive rate for each threshold and the ROC curve is an ascending diagonal line, whereas a nicely performing classifier has a high true positive rate with a low false positive rate when the threshold is adjusted correctly. The detection was based on the average posterior association probability for methods M1 and M2 and on the MAP estimate for M3. For M1, only object nodes contributed on the association probability, defined as nodes whose prior association probability exceeded 0.5 in the image in question. These values were computed and stored for each matched object image of the sequences which contained 15 images. The number of matched sequences was 200 unless otherwise noted. The object model formed from 1, 3, 6, 10 and 15 processed images were matched in randomly chosen Caltech background images (Fergus et al., 2003). The ROC curves can thus be formed for these numbers of training images. Again, the three methods processed the same object images so as to exclude the variance caused by randomness in the chosen images.

Typical summary statistics are the area under the ROC curve (AUR) and equal error rate (EER). The AUR is equal to the probability that a randomly chosen object image is correctly detected against a randomly chosen background image whereas the EER is the true positive rate at the point where the rate of true positives equals one minus the rate of false positives, being thus the intercept of the ROC curve and a descending diagonal line of the graph. For a random classifier, both the detection figures are half, whilst a perfect classifier yields the values of unity — the presented methods are expected to perform in between the two extrema. Because the AUR and EER measure somewhat different characteristics, the mean of the AUR and EER values are used as a final single detection score for comparing the methods of this thesis within each other.

In Figure 5.8, the detection scores of the dog images for the three methods are depicted. For methods M1 and M2, the scores are low with a few training images but increase towards unity as more images are included in the sequence. On the contrary, the M3 method is able to separate the object and background images already with one training image. This is as expected, for various reasons. In the M2 method, the value of the non-associative likelihood, κ, is a mixture of the previous estimate and an estimate that is based on the current test image. Hence, when matching a background image with only one training image κ is adapted to a low value as the method tries to find something common in the images. With more training images available the estimate based on the current test image has less meaning and eventually, in a long sequence, the value of κ adjusts itself to a value that is suitable for the object in question. Also, the portion of object nodes increases along the sequence which alleviates the matching and increases the average posterior association probability. The M3 method aims to include only object nodes in the node set and the score is based on the MAP estimate of the

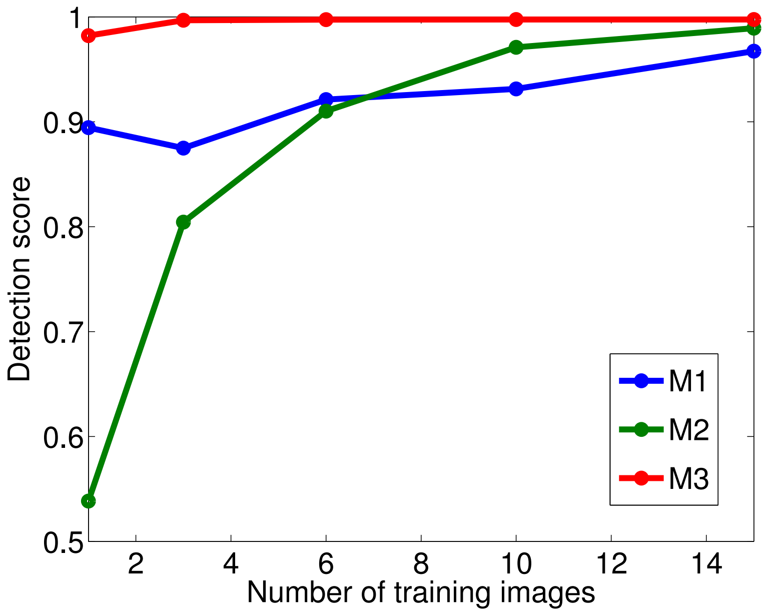


Figure 5.8: The ROC scores for dogs.

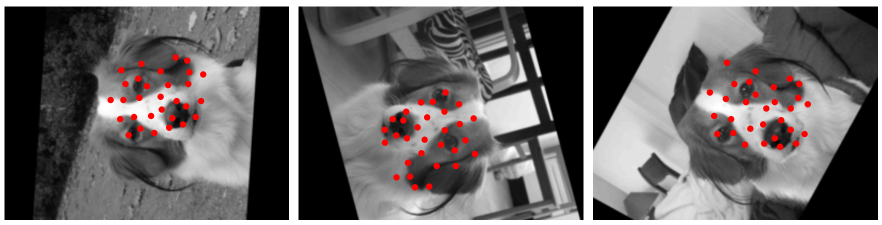


Figure 5.9: The corresponding points of the rotated dog images matched by M3.

probability of the corresponding points, which is why the method does a good job in separating the object and background images already from one training image.

In order to quantitatively demonstrate the orientation invariance of the M3 method, a test was made in which the dog images were arbitrarily rotated before matching. Three images, matched by M3, are illustrated in Figure 5.9. The ROC scores are given in Figure 5.10, showing how the M3 method yields scores (almost) as high as with the non-rotated images. M1 and M2 perform also surprisingly well, probably due to the following reason. Each different rotation ('different' here means an angle difference of more than, say, 20 degrees) basically appears as a different object. As M2 and especially M1 are robust methods, they are, in principle, able to simultaneously represent many objects. The more images processed, the higher is the probability that the rotation angle in the test image is approximately the same as in one of the processed images. Thus, the ROC scores increase along the image sequence, at least up to a certain point.

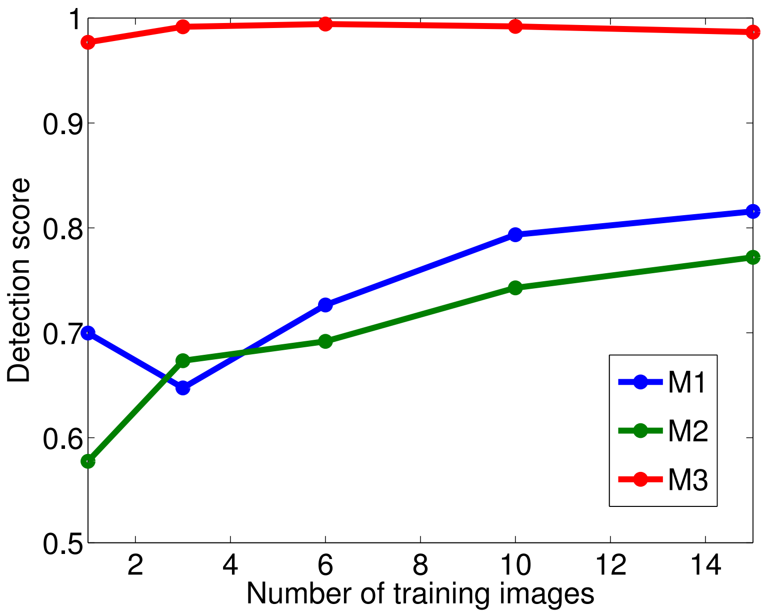


Figure 5.10: The ROC scores for rotated dogs.

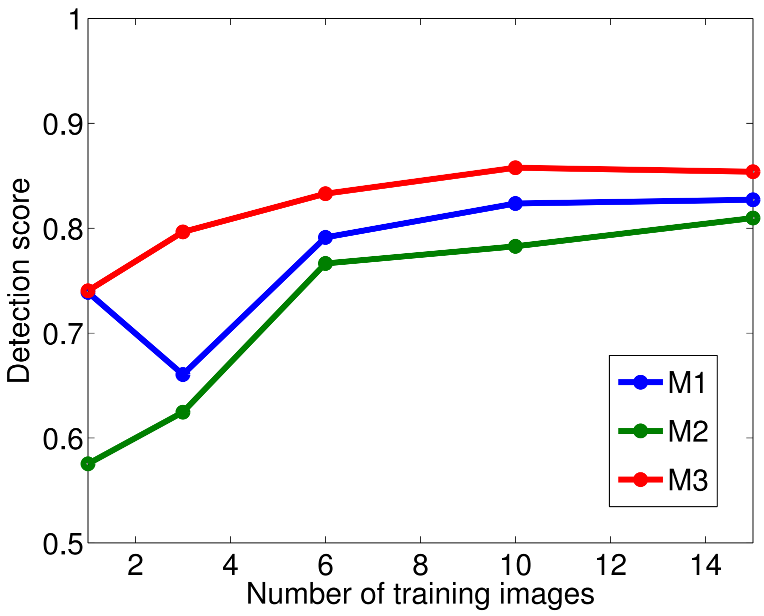


Figure 5.11: The ROC scores for Caltech faces.

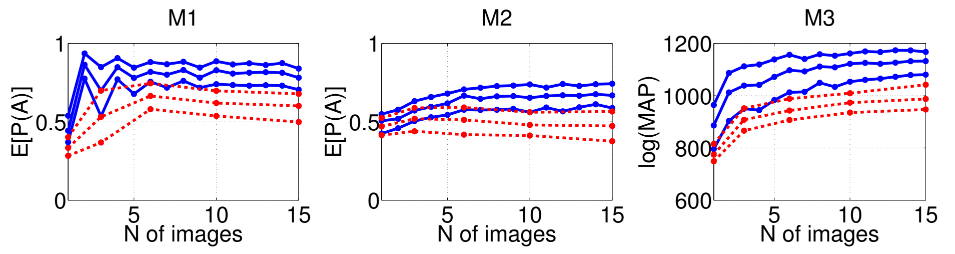


Figure 5.12: The quantiles of the values on which the detection of each method is based for the Caltech face images. Blue solid lines give the quantiles ($25\%, 50\%$ and 75%) of the object test images and red cross lines of the background test images (available only for $N = 1, 3, 6, 10, 15$).

The detection scores for the Caltech face images are illustrated in Figure 5.11. The M3 method again outperforms the other two methods, however, with smaller difference than in the dog images. The object appears in the same orientation and usually in the middle in each image and incorporating this prior knowledge in M3 (which is implicit in M1 and almost implicit in M2) would increase the scores (this is shown later in this chapter). It is somewhat surprising that M1 outperforms M2. In Figure 5.12, the association probabilities of M1 and M2, and the MAP estimates of M3, are depicted for each number of training images. The median value based adaptation of the non-associative likelihood parameter κ of M1 causes a regular fluctuation on the average posterior association probabilities at the beginning of the sequence. It seems that the adaptation method is not paying off until about five images and the constant value set at the beginning of the sequence ($\kappa = \exp(0.5\beta)$) happens to give smaller detection errors than the value adjusted after three processed images (note that there is no value for background test images with two, four or five training images). This explains the drop of the detection scores of M1 at three training images, being visible in each database but very apparent in Figure 5.11. The average posterior association probabilities of M2 for object and background images diverge along the sequence, leading to steadily increasing ROC scores. The MAP estimates of M3 increase with more processed images but so do those of the background images. This is probably due to the increasing number of likelihood kernels in the mixture; with more kernels, the number and masses of the false modes of the (unnormalized) posterior increase. Naturally this applies to M1 and M2 as well, but the full rotation invariance of M3 makes things worse in this sense. Besides, the detection of M1 and M2 is based on the association probabilities for which an optimal value of κ is estimated and the estimate improves along the sequence.

Figure 5.13 shows the detection scores for the Bio-Id face images which are more difficult than Caltech face images due to increased variations in lighting, expression and pose. Hence, it is surprising to see M1 perform so well on this database which is clearly too difficult for M2 and M3. However, the rotation

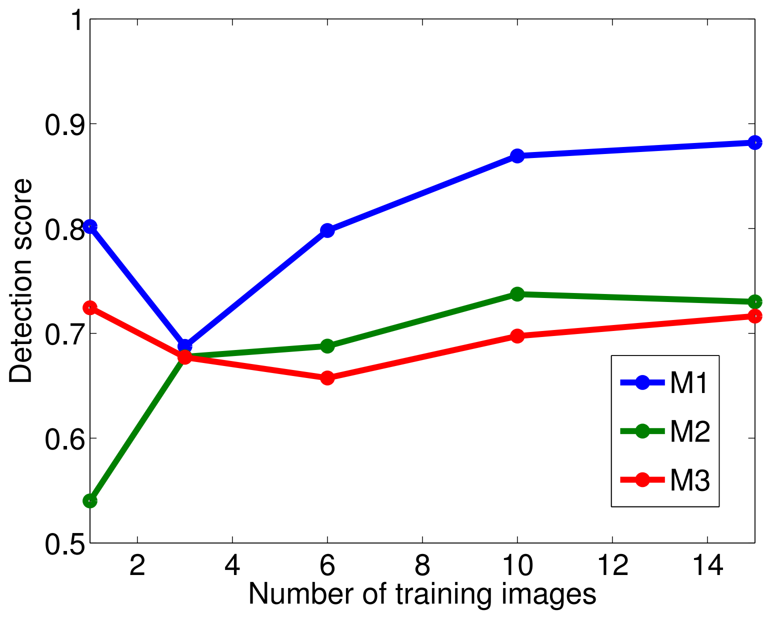


Figure 5.13: The ROC scores for Bio-Id faces.

changes of the faces are again small which facilitates the matching of M1. The failure of M2, and partly of M3, is probably an indication of the unsuccessful initialization of the particles; more samples would be needed to increase the scores.

In Figure 5.14, the detection scores for the peacock butterfly images are shown. The butterflies are photographed under arbitrary orientations and it is not a wonder that the M1 and M2 methods fail to learn the object model (although M1, again, outperforms a random classifier). On the contrary, M3 detects the butterflies moderately well in novel test images. To assess the question whether the number of sequences, 200, is enough for statistically reliable results, the ROC scores of the peacock butterfly images were computed for M3 using N_s number of sequences. N_s was varied from 1 to 200 and in Figure 5.15, the scores for each value of N_s are plotted. The values seem to change relatively little after 100 sequences although there is some fluctuation also when N_s approaches 200. Probably a much larger number of sequences, say 2000, would be needed for real convergence. However, this would lead to impractical computation times.

From the used databases, reference detection results are found for Caltech face images and butterfly images. According to the results, the M3 method seems to be the best method for detecting these images. In order to (hopefully) obtain better results, the face images were matched again with M3 by doubling the total number of nodes (and the Monte Carlo particle number) to $N_p = 300$ and hence also doubling the maximum size of a node set to $N_{max} = 60$. The experiment also serves as a comparative test of how much the detection scores change when using more nodes and particles. This time, both the AUR and EER scores are given. The

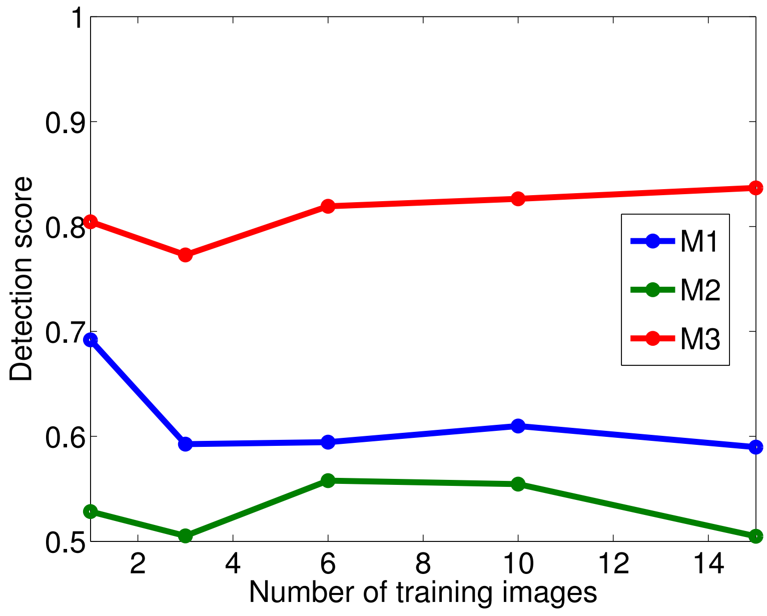


Figure 5.14: The ROC scores for peacock butterflies.

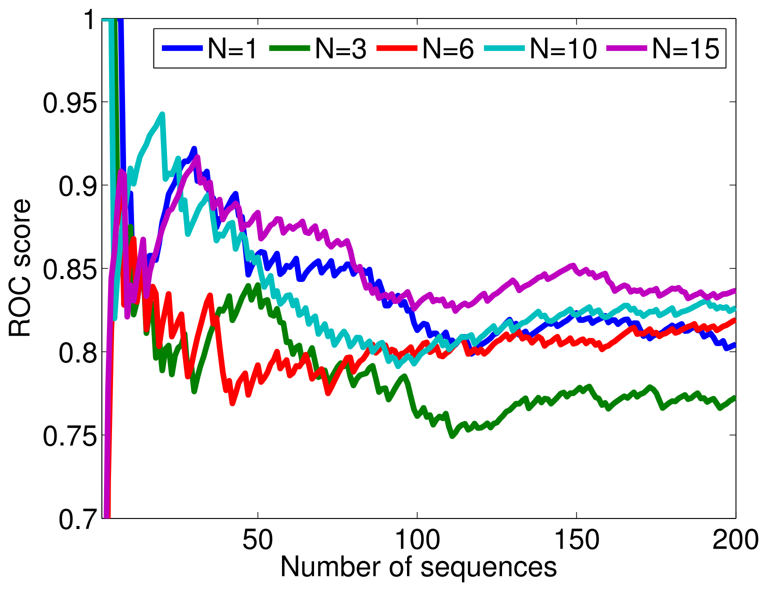


Figure 5.15: The ROC scores of the peacock butterflies of method M3 for different number of training images ($N = 1, 3, 6, 10, 15$), plotted against the number of sequences used in computing the score.

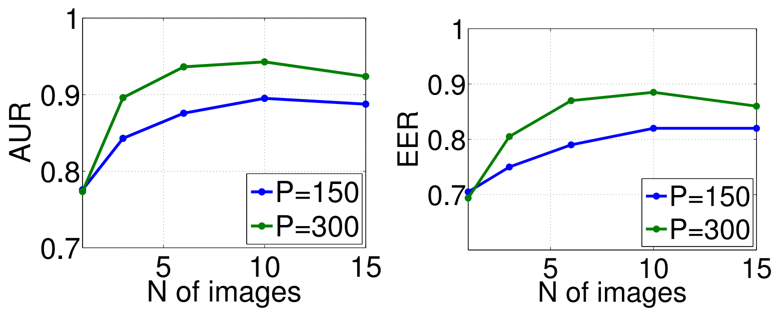


Figure 5.16: Areas under the ROC curve (AUR) and equal error rates (EER) of Caltech faces for M3, with two different number of particles.

Table 5.5: The equal error rates (EER) of Caltech faces for different methods, in percents. N refers to the number of images used for training the object model. All the reference methods are batch methods. Some values were visually estimated from a graph.

Method	N	EER	Notes
M3	1	69.4	rotation and scale invariant
M3	6	87.0	rotation and scale invariant
M3	1	85.5	no invariances
M3	6	96.0	no invariances
Weber et al. (2000)	100	93.5	
Fergus et al. (2003)	225	96.4	
Fei-Fei et al. (2003)	1	$\sim$88	Visually estimated from a graph
Fei-Fei et al. (2003)	4	$\sim$83	Visually estimated from a graph
Willamowski et al. (2004)	225	99.3	
Amores et al. (2005)	300	86.0	Utilize color information
Shotton et al. (2005)	50	96.5	Hand-segmented training samples
Wolf and Martin (2005)	1	$\sim$59	Visually estimated from a graph
Wolf and Martin (2005)	10	$\sim$89	Visually estimated from a graph
Opelt et al. (2006a)	60	100	
Lazebnik et al. (2006)	100	98.5	
Serre et al. (2007)	225	98.2	
Escalera et al. (2007)	90	97.7	Uses supervised training
Zhang et al. (2007)	225	100	
Holub et al. (2008)	100	91	Uses Caltech-101 faces

scores for using both $N_p = 150$ and $N_p = 300$ parameters are plotted in Figure 5.16. The model with 300 particles clearly dominates except with one training image. A possible cause is that irrespective of the number of nodes, the value of the posterior distribution at the correct location in the second image is not necessary large compared with a local maximum somewhere at the background test image,

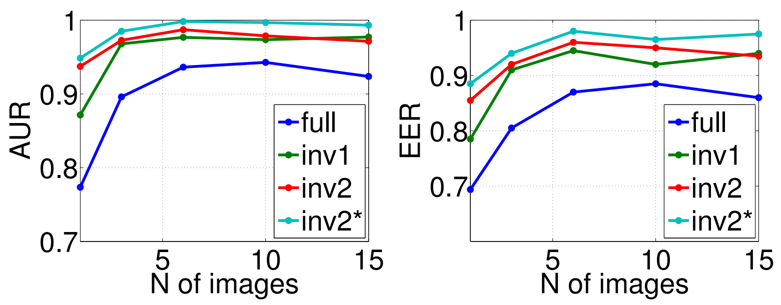


Figure 5.17: Areas under the ROC curve (AUR) and equal error rates (EER) of Caltech faces for different versions of the M3 method. 'Full' refers to the ordinary M3 method, in 'inv1' the rotation invariance has been dropped, in 'inv2' the rotation and scale invariances have been dropped, and 'inv2*' is the same as 'inv2' but the images of Caltech101-faces database were used, in which the 15 images with the face on a very small scale are excluded.

or may even be inferior to it, but with more training images, the difference is larger with more nodes in the node set.

The rotation and scale invariances are likely to worsen the detection results when the object instances appear in the same orientation and size. This was also noticed by Bay et al. (2006) who state that "the additional complexity of full affine-invariant features often has a negative impact on the robustness and does not pay off, unless really large viewpoint changes are to be expected". Hence, the Caltech faces were also matched with M3 by dropping the invariances which indeed improves the results as demonstrated in Figure 5.17. Also, the results of using Caltech101-faces (Fei-Fei et al., 2007) images are shown. The Caltech101-faces database is the same as the Caltech faces database used so far, but 15 images with the face on a very small scale have been excluded, making it thus a bit easier database to match. Using this database further improves the results.

Some EER values of Figure 5.17 are given in Table 5.5, together with values of various reference methods. There seems to be a trade-off between the number of training images and the detection rate although for instance the method of Opelt et al. (2006a) performs perfectly with just 60 training images. It should be noted that the figures are mean values and sometimes the variance between the differently learned object models can be significant. There are also remarkable differences in the computation times of the methods. All the methods use dozens or hundreds of training images except those of Fei-Fei et al. (2003) and Wolf and Martin (2005); comparing with these, the results of M3 are equal or even better when the invariances are dropped.

The results of the method of Fei-Fei et al. (2007), which is the only incremental object detection method found in the literature, can be compared with the Caltech101-face detection results of this work. They report the results of using 1, 3, 6 and 15 training images. However, there are two problems which complicate

Table 5.6: The areas under the ROC curve for the two incremental methods, in percents. N refers to the number of images used for training the object model. The exact results of Fei-Fei et al. (2007) are unavailable; see text for details.

Method	$N = 1$	$N = 3$	$N = 6$	$N = 15$
M3	92	99	99	99
Fei-Fei et al. (2007)	$\lesssim$86	$\lesssim$92	$\lesssim$91	$\approx$88

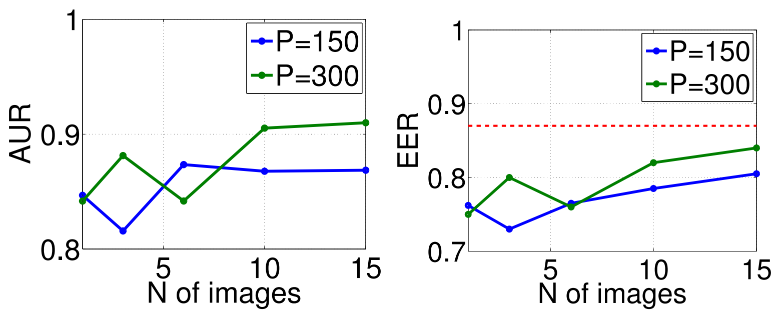


Figure 5.18: Areas under the ROC curve (AUR) and equal error rates (EER) of peacock butterflies for M3, with two different particle numbers. The images in the sequences were different for the two method. Red dashed line shows the results of the batch method of Lazebnik et al. (2004).

a fair comparison. First, Fei-Fei et al. (2007) report the results only graphically and in a bit unorthodox way, making it impossible to assess which of the 101 results corresponds to the face category. Hence, only an upper limit can be given, being the actual result only if it was the face category that performed best of all the 101 categories. An exception is the case with 15 training images, for which they give the category identities of the results. Second, as the results are only in a graphical form, the performance values must be estimated visually. The comparison is made in Table 5.6 which shows that M3 clearly outperforms the reference method. Because the method of (Fei-Fei et al., 2007) lacks rotation invariance, the comparison is made with the rotation variant (but scale invariant) version of M3. It should, however, be noted that the method of Fei-Fei et al. (2007) is (at least to some extent) able to detect objects with highly varying appearance and shape (such as airplanes and joshua-trees) that are depicted in varying view points. This generality probably comes at the cost of a decreased performance for a single object category, compared with a method which aims to detect relatively rigid objects portrayed in a somewhat constant viewpoint, like the presented method.

The seven butterfly databases were matched with the M3 method, using again the heavier parameter setting with $N_{max} = 60$ and $N_p = 300$. For the sake of computational cost, only 100 image sequences were matched for each butterfly type. Figure 5.18 shows the improvement of using more nodes and particles. In Figure 5.19, the EER values for the rest of the six butterfly categories are presented.

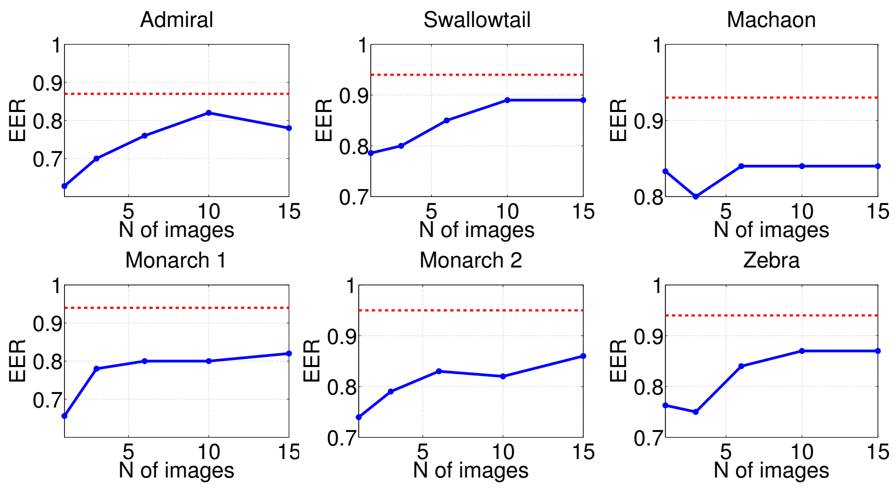


Figure 5.19: EER values of different butterflies using M3. Red dashed line shows the results of the batch method of Lazebnik et al. (2004).

The results of the reference method are out of reach for M3. However, the reference method (Lazebnik et al., 2004) is a batch method which needs 26 training images to form the object model. There is also a substantial amount of variation in the EER values as the results are expected to differ between experiments. The same is true for the reference method which apparently was tested with only one set of training images.

There is one notable weakness in the M3 method, namely that sometimes the results are worse with 15 training images than with 10. This is in contrast with the general principle of learning which states that having more training samples improves the performance of a pattern recognition system. However, one must remember that the nature of the learning is incremental and most of the information in the images is discarded after each match. Hence, the situation is very different compared with the batch learning and may lead to an unexpected behavior that is difficult to explain. One reason might be that the variance in the shape model increases along the sequence for some nodes whose location is ambiguous. A good example are the nodes near the edge of the object the appearance of which is affected by the background. A node set with larger variance in their prior distribution fits better in the background images which lowers the detection rates. It seems like an extremely difficult problem to incrementally learn the corresponding points whose relative location would be as rigid as possible. For a human face, this would usually mean only the eyes, nose and mouth nodes, as the appearances elsewhere in the face differ between individuals. For instance, if the persons in the first two images have similar haircuts, M3 includes the forehead nodes in the node set. If the following images contain persons with different hairstyles, these nodes start to 'wander' as they search for similar appearance and the search area increases after each image due to the update of the shape variance. Obviously, there is a limit for

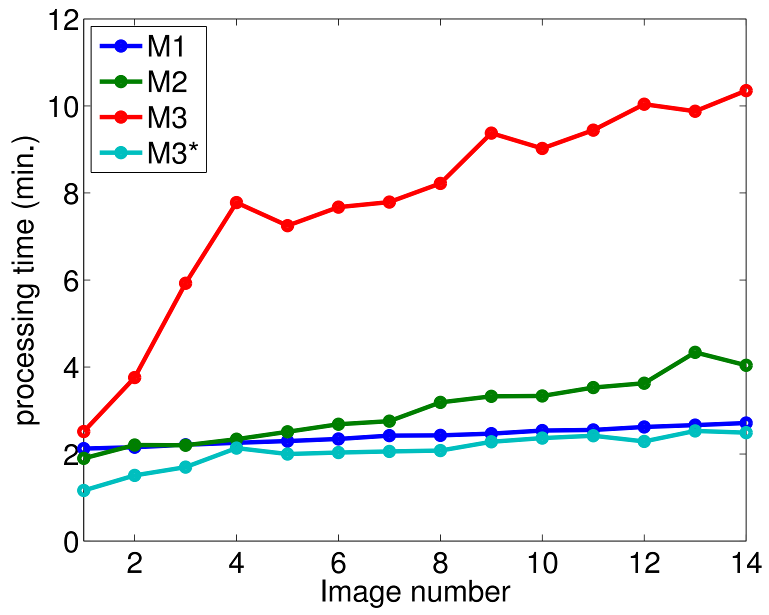


Figure 5.20: The median CPU times for the three methods needed for processing the next image. M3* refers to M3 without invariances.

the flexibility of the objects that can be matched with the proposed system. The more rigid the object, the easier its model is to learn incrementally. Having a variance on the shape model in the beginning which exceeds the ’true’ value is not dangerous as it adapts itself to the actual level of rigidity, unless it is so high in the beginning that the matching fails. In the method of Fei-Fei et al. (2003), same kind of phenomenon of having a weaker detection rate with more training images also sometimes happens although it is a batch method. However, they refused to comment on this behavior in any way. The incremental method of Fei-Fei et al. (2007) suffers from similar problems as their errors often increase, at least momentarily, with more training images. For some categories, they report similar results with 15 training images than with one training image and the batch version of their algorithm outperforms the incremental version when the number of training images exceeds six. According to them, the degradation of the incremental performance is because “the incremental method is much more sensitive to the quality of the training images and to the order in with they are presented to the algorithm. Therefore it is more likely to form suboptimal models. [...] Less information is carried along by the incremental algorithm from one learning epoch to the next, while the batch algorithm has all training images available at the same time”.

5.8 Computational requirements

The proposed methods are computationally quite heavy. It takes minutes, rather than seconds, to match one image using a totally unoptimized Matlab implementation on a regular desktop computer. The median computation times per each image of the 200 face image sequences are depicted in Figure 5.20. Investigating the mean computation times would be unreasonable because in the used computer grid system some individual runs may halt for a very long time. As further images are processed, matching the next image takes more time because the number of the likelihood kernels increases. For the M1 method this increase is subtle. Also, the computation times of M2 increase moderately along the sequence. On the contrary, processing the following image with M3 is typically much heavier than processing the previous image. This is because in M3, percentually much more time is spent in computing the similarity values for which the computation time grows linearly along the sequence. On the other hand, the scale and rotation variant version of M3 has the smallest processing times. Because this version also achieved the best results for Caltech faces, the invariances should definitely be left out when it is known a priori that the object appears at the same scale and pose in each image.

The major computational bottlenecks are the sampling, and with M2 and M3, also the initialization. The initialization step is proportional to the number of different scales and orientations with which the particles are initialized. The sampling time is proportional to the particle number and to the number of nodes. PMC sampling time is also proportional to the number of iterations. Luckily, these steps are parallelizable as in the initialization the best particle locations over different scales and orientations are independently searched and as the particles are sampled independently. A parallel implementation would thus reduce the computation time dramatically. Furthermore, implementing the algorithms in the GPU (graphical processing unit) environment would probably make the system real time.

Chapter 6

Conclusion

This thesis has presented probabilistic models and methods for incremental object matching which is an unexplored field of computer vision. The Bayesian approach was taken as it offers a theoretically consistent way for updating the object model, for separating the appearance and shape modeling, and for handling uncertainties that arise in natural images. Due to the modularity of the Bayesian framework, any part — appearance or shape — can be modified without affecting the other part. The dimension of the problem is rather high (dozens) which makes the conventional numerical integration methods unfeasible to use. On the other hand, the multimodal posterior distribution is unlikely to be well approximated at least with any unimodal simple distribution, and hence particle Monte Carlo methods were utilized. Especially the population Monte Carlo method is an appealing approach to solve the problem; multiple hypotheses about the largest posterior modes are made in terms of the particles, and the goodness of the hypotheses is revealed by the particle weights which are directly proportional to the value of the (unnormalized) posterior distribution, evaluated at the sampled points.

Three different methods are presented in this thesis, each of which is an implementation of the theoretical Bayesian framework for incremental object matching. In all the methods, the object is modeled as a set of feature points, called nodes, whose appearance is modeled with Gabor responses and whose shape is assumed to follow a Gaussian distribution. The methods differ in the approximations and 'shortcuts' made, and in the sampling methods. The simplest method uses the same set of node points with the same node variance in each image, and samples the images with a sequential Monte Carlo algorithm where the nodes are sampled one at a time and the resampling is based on the values of the marginal posterior distributions in different subspaces. The level of invariances in the simplest method is small. In the extended method, the node set is allowed to change by eliminating background nodes and setting new nodes, and the method is able to handle larger transformations. Also, the variances of the nodes are updated according to Bayes' rule, and the posterior is sampled with a population Monte Carlo method in which the particles are resampled according to the value of the full posterior. Finally, the most sophisticated method possesses full rotation invariance and takes (almost) full advantage of the Bayesian framework; the parameter space includes all the possible node sets in the images, not just the fixed node set as in the simpler methods. For practical reasons, the number of different possible nodes is limited, but the method is able to carry multiple hypotheses about the common object during the process. The sampling algorithm is a population Monte Carlo -like implementation.

The matching errors of the three proposed methods were compared using the dog image database and the results indicate a similar performance. Also, the methods are able to match the corresponding points (nearly) as accurately as other prominent feature point based matching methods. Because the reference methods utilize a large pre-annotated training set in contrast with the proposed methods which expand the training set, starting from just one image, and avoid using pre-annotations, the results can be considered as extremely promising.

Various databases were used in conducting the detection tests. The smallest detection errors are usually given by the most sophisticated method. However, for the Bio-Id face database the simplest method gives the best results. The experiments reveal that if an object appears in the same scale and orientation in the images, having invariances against these can degrade the results. The detection capabilities of the most sophisticated method are almost on a par with other published object detection methods, especially when the invariances are dropped from the method. Again, the reference methods are (usually) batch methods, and also, the reference methods do not match the corresponding points so they lack the precise object location information. In addition, the proposed method outperforms the only incremental object detection method found in the literature. It should be noted that the population Monte Carlo based methods of the thesis would probably benefit from better particle initialization schemes.

It can be concluded that the primary aim of the thesis — to develop probabilistic methods for incremental object matching — has been met. Some issues have been excluded from the study and deserve further research. The present appearance and shape models are probably not powerful and scalable enough to handle multiple classes and larger variations in the viewpoint. If the models are modified, an open question is how well the improved models would fit into the incremental framework, that is, how well does the principle of the incremental learning generalize to arbitrary complex models.

Appendix

Image databases

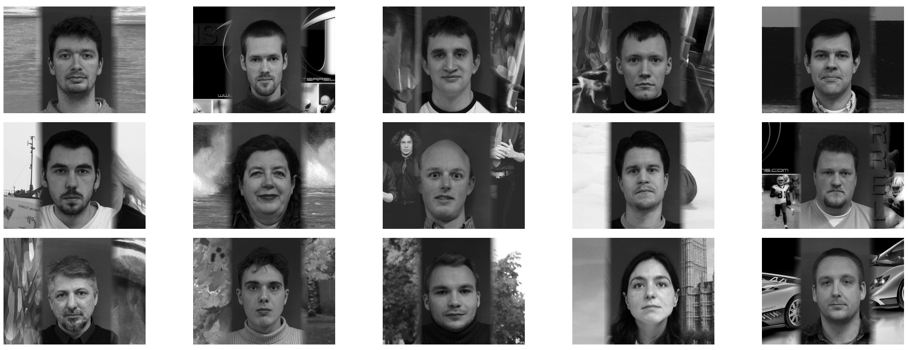


Figure 6.1: Examples of the DTU database. The database contains 37 images.

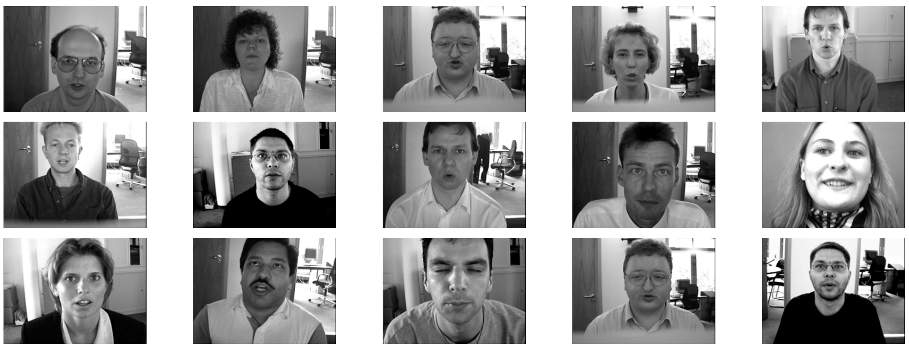


Figure 6.2: Examples of the Bio-Id database. The database contains 1521 images.

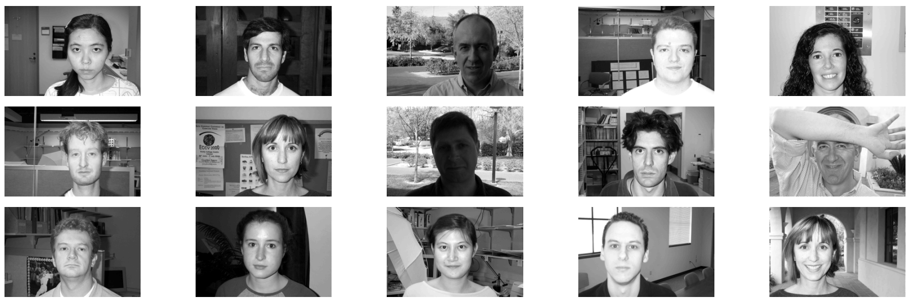


Figure 6.3: Examples of the Caltech faces database. The database contains 450 images.

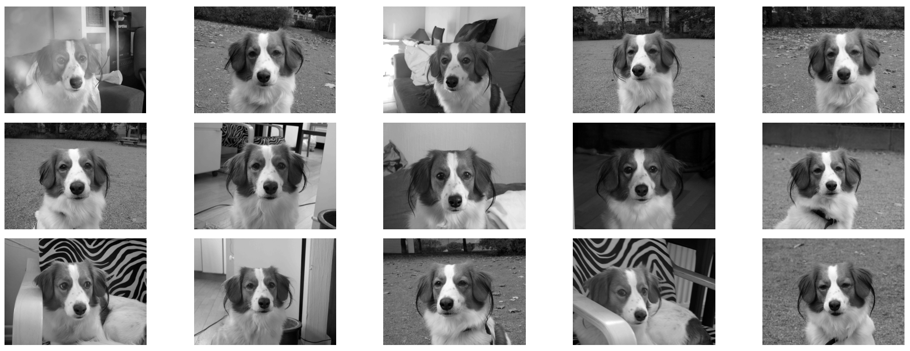


Figure 6.4: Examples of the dogs database. The database contains 64 images.

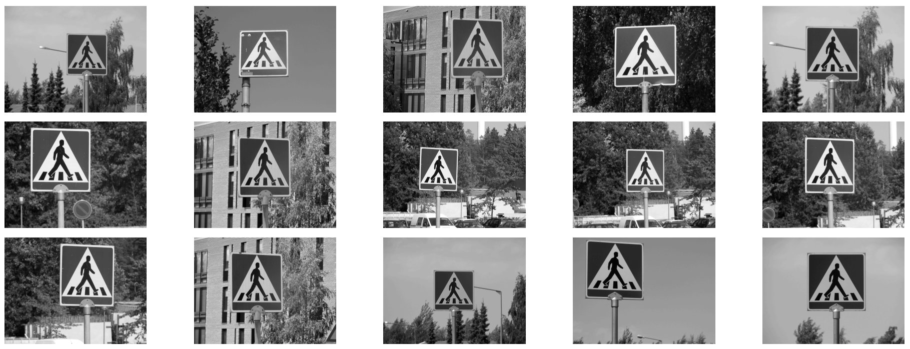


Figure 6.5: Examples of the traffic signs database. The database contains 35 images.

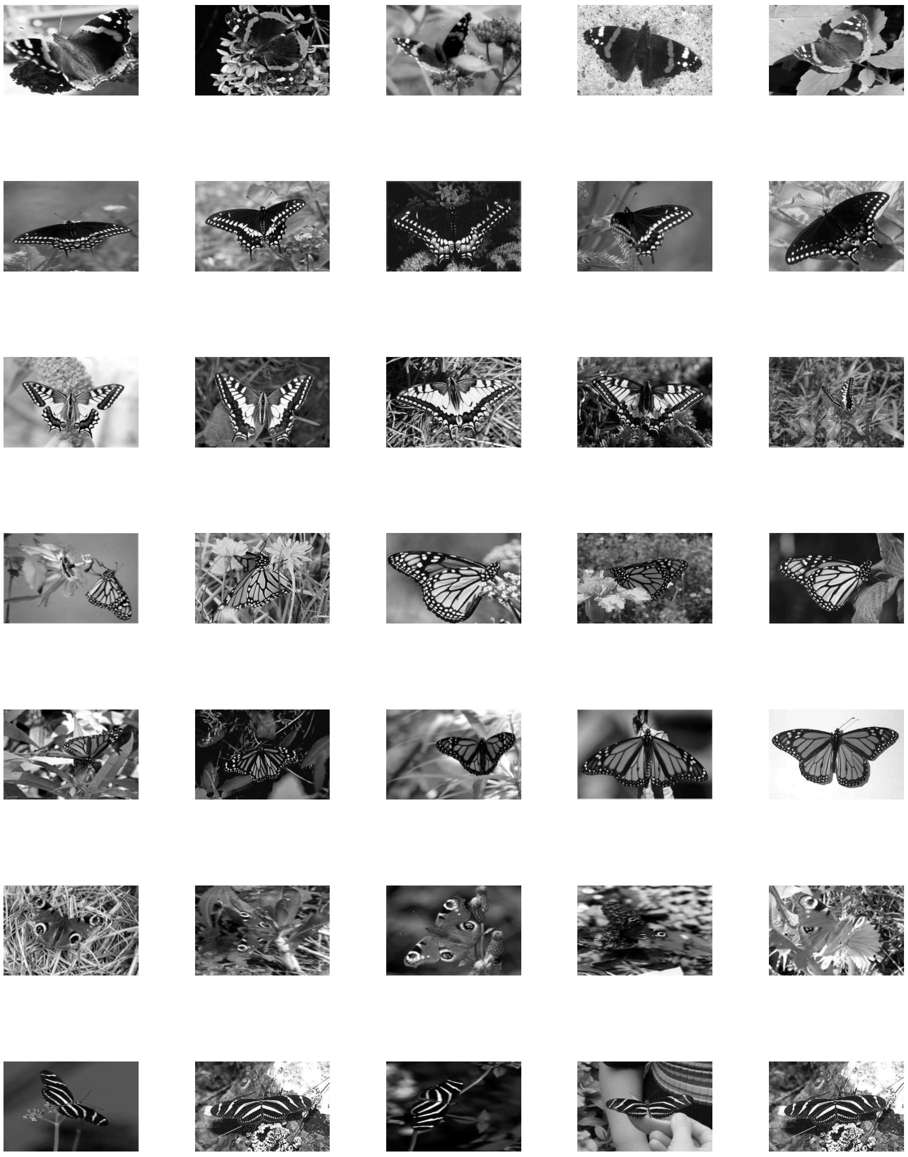


Figure 6.6: Examples of each of the seven butterfly classes.

Bibliography

Ahuja, N. and Todorovic, S. (2007). Learning the taxonomy and models of categories present in arbitrary images. In *Proc. of IEEE Intl. Conf. on Comp. Vis.*

Amores, J., Sebe, N., and Radeva, P. (2005). Fast spatial pattern discovery integrating boosting with constellations of contextual descriptors. In *Proc. CVPR*, volume 2, p. 769.

Ballard, D. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122.

Ballentine, L. E. (1998). *Quantum Mechanics*. World Scientific.

Bar-Shalom, Y. and Li, X. (1995). Multitarget-multisensor tracking: Principles and techniques. *Storrs, CT: University of Connecticut, 1995.*

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. *Lecture notes in computer science*, 3951:404–417.

Belhumeur, P., Hespanha, J., Kriegman, D., et al. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720.

Bernardo, J. and Smith, A. (1994). *Bayesian Theory*. John Wiley and Sons.

Berzuini, C. and Gilks, W. (2003). Particle filtering methods for dynamic and static Bayesian problems. In Green, P., Hjort, N., and Richardson, S., editors, *Highly Structured Stochastic Systems*, pp. 207–227. Oxford University Press.

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer New York.

Bolstad, W. (2004). *Introduction to Bayesian Statistics*. Wiley-IEEE.

Borenstein, E., Sharon, E., and Ullman, S. (2004). Combining top-down and bottom-up segmentation. In *Proceedings IEEE workshop on Perceptual Organization in Computer Vision, CVPR 2004*.

Brunelli, R. and Poggio, T. (1993). Face recognition: Features versus templates. *IEEE transactions on pattern analysis and machine intelligence*, 15(10):1042–1052.

Burl, M., Weber, M., and Perona, P. (1998). A probabilistic approach to object recognition using local photometry and global geometry. *Lecture notes in computer science*, 1407:628–641.

Cappe, O., Guillin, A., Marin, J., and Robert, C. (2004). Population Monte Carlo. *J. Comput. Graph. Statist*, 13(4):907–929.

Chatfield, C. and Collins, A. (1980). *Introduction to multivariate analysis*. CRC Press.

Cho, T. (2006). Object matching using generalized Hough transform and chamfer matching. *Lecture Notes in Computer Science*, 4099:1253–1257.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552.

Chum, O. and Zisserman, A. (2007). An exemplar model for learning object classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. Citeseer.

Cootes, T., Edwards, G., Taylor, C., et al. (2001). Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685.

Cox, G. (1995). Template matching and measures of match in image processing. Technical report, Department of Computer Science, University of Cape Town.

Crandall, D., Felzenszwalb, P., and Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, p. 8.

Craven, P. and Wahba, G. (1978). Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403.

Daugman, J. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A*, 2(7):1160–1169.

Daugman, J. (1988). Complete discrete 2-d Gabor transforms by neural networks for imageanalysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing]*, 36(7):1169–1179.

Doucet, A., De Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo methods in practice*. Springer Verlag.

Ersi, E. and Zelek, J. (2006). Local feature matching for face recognition. In *Computer and Robot Vision. The 3rd Canadian Conference on*, pp. 4–10.

Escalera, S., Pujol, O., and Radeva, P. (2007). Boosted Landmarks of Contextual Descriptors and Forest-ECOC: A novel framework to detect and classify objects in cluttered scenes. *Pattern Recognition Letters*, 28(13):1759–1768.

Fei-Fei, L., Fergus, R., and Perona, P. (2003). A Bayesian approach to unsupervised one-shot learning of object categories. In *Proc. ICCV*, pp. 1134–1141.

Fei-Fei, L., Fergus, R., and Perona, P. (2007). Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70.

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Proc. CVPR*, pp. 264–271.

Fergus, R., Perona, P., and Zisserman, A. (2006). A sparse object category model for efficient learning and complete recognition. *Lecture Notes in Computer Science*, 4170:443.

Gabor, D. (1946). Theory of communication: J. Inst. *Electr. Eng*, 93:429–457.

Gelman, A., Carlin, J., Stern, H., and Rubin, D. (2004). *Bayesian Data Analysis*. Chapman and Hall.

Gonzalez, R. and Woods, R. (1993). Digital Image Processing.

Goodall, C. (1991). Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 285–339.

Günther, M. and Würtz, R. (2009). Face detection and recognition using maximum likelihood classifiers on Gabor graphs. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(3):433–461.

Haykin, S. (2001). *Kalman filtering and neural networks*. Wiley-Interscience.

Holub, A., Welling, M., and Perona, P. (2008). Hybrid generative-discriminative visual categorization. *International Journal of Computer Vision*, 77(1):239–258.

Jain, A. (1989). *Fundamentals of digital image processing*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Jesorsky, O., Kirchberg, K., and Frischholz, R. (2001). Robust face detection using the Hausdorff distance. *Proceedings of the Third International Conference on Audio-and Video-Based Biometric Person Authentication*, pp. 90–95.

Jiang, T., Jurie, F., and Schmid, C. (2009). Learning shape prior models for object matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 848–855.

Jung, J., Lee, H., Lee, J., and Park, D. (2010). A novel template matching scheme for fast full-search boosted by an integral image. *IEEE Signal Processing Letters*, 17:83–86.

Jurie, F. and Dhome, M. (2002). Real time robust template matching. In *British Machine Vision Conference*, pp. 123–131.

Kalliomäki, I. (2007). *Probabilistic Methods for Pose-Invariant Recognition in Computer Vision*. PhD thesis, Helsinki University of Technology. Report B67.

Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE Computer Society.

Koeser, K., Perwass, C., and Sommer, G. (2006). Dense optic flow with a Bayesian occlusion model. *Lecture Notes in Computer Science*, 3667:127.

Lades, M., Vorbruggen, J., Buhmann, J., Lange, J., von der Malsburg, C., Würtz, R., and Konen, W. (1993). Distortion invariant object recognition in the dynamic linkarchitecture. *Computers, IEEE Transactions on*, 42(3):300–311.

Lazebnik, S., Schmid, C., and Ponce, J. (2004). Semi-local affine parts for object recognition. In *Proc. BMVC*, volume 2, pp. 959–968.

Lazebnik, S., Schmid, C., and Ponce, J. (2005). A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). A discriminative framework for texture and object recognition using local image features. *Lecture notes in computer science*, 4170:423–442.

Lee, T. (1996). Image representation using 2 D Gabor wavelets. *IEEE Transactions on pattern analysis and machine intelligence*, 18(10):959–971.

Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *ECCV workshop on statistical learning in computer vision*, pp. 17–32.

Leibe, B., Leonardis, A., and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1):259–289.

Leibe, B. and Schiele, B. (2003). Interleaved object categorization and segmentation. In *British machine vision conference*.

Lowe, D. (1999). Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, volume 2, pp. 1150–1157.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Mengersen, K. and Robert, C. (2003). IID sampling using self-avoiding population Monte Carlo: the pinball sampler. *Bayesian Statistics*, 7:277–292.

Metropolis, N., Rosenbluth, A., Rosenbluth, R., Teller, A., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Miao, X. and Rao, R. (2007). Learning the Lie groups of visual invariance. *Neural Computation*, 19(10):2665–2693.

Mikolajczyk, K., Leibe, B., and Schiele, B. (2006). Multiple object class detection with a generative model. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Volume 1*, pp. 26–36.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1615–1630.

Moghaddam, B., Wahid, W., and Pentland, A. (1998). Beyond eigenfaces: Probabilistic matching for face recognition. In *Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998. Proceedings*, pp. 30–35.

Monteleoni, C. and Kääriäinen, M. (2007). Practical online active learning for classification. *24th IEEE CVPR*, pp. 249–263.

Murphy-Chutorian, E., Aboutalib, S., and Triesch, J. (2005). Analysis of a biologically-inspired system for real-time object recognition. *Cognitive Science Online*, 3(2):1–14.

Neal, R. (1993). Probabilistic inference using Markov chain Monte Carlo methods. Technical report, Department of Computer Science, University of Toronto.

Nguyen, H., Worring, M., and Van Den Boomgaard, R. (2001). Occlusion robust adaptive template tracking. In *Proc. Int. Conf. Computer Vision*, pp. 678–683.

Opelt, A., Pinz, A., Fussenegger, M., and Auer, P. (2006a). Generic object recognition with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):416–431.

Opelt, A., Pinz, A., and Zisserman, A. (2006b). A boundary-fragment-model for object detection. *Lecture Notes in Computer Science*, 3952:575.

Opper, M. (1996). On-line versus off-line learning from random examples: General results. *Physical review letters*, 77(22):4671–4674.

Orr, M. (1996). Introduction to radial basis function networks. *Center for Cognitive Science, University of Edinburgh, Scotland*.

Palmer, S. (1999). *Vision science: Photons to phenomenology*. MIT press Cambridge, MA.

Pantofaru, C., Dorko, G., Schmid, C., and Hebert, M. (2006). Combining regions and patches for object class localization. In *Beyond Patches Workshop, CVPR*.

Pentland, A., Moghaddam, B., and Starner, T. (1994). View-based and modular eigenspaces for face recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, pp. 84–91.

Phillips, P., Moon, H., Rizvi, S., and Rauss, P. (2000). The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1090–1104.

Robert, C. and Casella, G. (2004). *Monte Carlo statistical methods*. Springer Verlag.

Rucklidge, W. (1997). Efficiently locating objects using the Hausdorff distance. *International Journal of Computer Vision*, 24(3):251–270.

Saad, D. (1998). *On-line learning in neural networks*. Cambridge University Press.

Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., and Poggio, T. (2007). Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426.

Shen, L. and Bai, L. (2006). A review on Gabor wavelets for face recognition. *Pattern Analysis & Applications*, 9(2):273–292.

Shotton, J., Blake, A., and Cipolla, R. (2005). Contour-based learning for object detection. In *Proc. ICCV*, volume 1, pp. 503–510.

Smith, P., Sinclair, D., Cipolla, R., and Wood, K. (1998). Effective corner matching. In *British machine vision conference*, pp. 545–556. Citeseer.

Sonka, M., Hlavac, V., and Boyle, R. (1999). *Image Processing, Analysis, and Machine Vision*. International Thomson Publishing.

Spiegelhalter, D., Abrams, K., and Myles, J. (2004). *Bayesian approaches to clinical trials and health-care evaluation*. Wiley.

Steger, C. (2002). Occlusion, clutter, and illumination invariant object recognition. *Proceedings of the Conference on Photogrammetric Computer Vision*, 34:345–350.

Stegmann, M. B. (2002). Analysis and segmentation of face images using point annotations and linear subspace techniques. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark.

Sullivan, J., Blake, A., Isard, M., and Maccormick, J. (2001). Bayesian object localisation in images. *International Journal of Computer Vision*, 44(2):111–135.

Tamminen, T. (2005). *Models and Methods for Bayesian Object Matching*. PhD thesis, Helsinki University of Technology. Report B52.

Tamminen, T. and Lampinen, J. (2006). Sequential Monte Carlo for Bayesian matching of objects with occlusions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:930–941.

Toivanen, M. and Lampinen, J. (2009a). Bayesian online learning of corresponding points of objects with sequential Monte Carlo. *International Journal of Computational Intelligence*, 5(4):318–324.

Toivanen, M. and Lampinen, J. (2009b). Incremental Bayesian learning of feature points from natural images. In *Proc. CVPR workshops*, pp. 39–46.

Toivanen, M. and Lampinen, J. (2009c). Incremental object matching with Bayesian methods and particle filters. In *Proc. Digital Image Computing: Techniques and Applications*, pp. 111–118.

Toivanen, M. and Lampinen, J. (2010). Incremental object matching and detection with Bayesian methods and particle filters. *Computer Vision special issue. To be published.*

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.

Wang, H., Suter, D., Schindler, K., and Shen, C. (2007). Adaptive object tracking based on an effective appearance filter. *IEEE transactions on pattern analysis and machine intelligence*, 29(9):1661.

Weber, M., Welling, M., and Perona, P. (2000). Unsupervised learning of models for recognition. *Lecture Notes in Computer Science*, 1842:18–32.

Willamowski, J., Arregui, D., Csurka, G., Dance, C., and Fan, L. (2004). Categorizing nine visual classes using local appearance descriptors. In *ICPR Workshop on Learning for Adaptable Visual Systems*.

Winn, J. and Jojic, N. (2005). Locus: Learning object classes with unsupervised segmentation. In *Proc. ICCV*, volume 1. Citeseer.

Wiskott, L., Fellous, J.-M., Kruger, N., and von der Malsburg, C. (1997). Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:775–779.

Wolf, L. and Martin, I. (2005). Robust boosting for learning from few examples. In *proc. CVPR*, pp. 359–364.

Würtz, R. (1997). Object recognition robust under translations, deformations, and changes in background. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):769–775.

Yan, S., He, X., Hu, Y., Zhang, H., Li, M., and Cheng, Q. (2004). Bayesian shape localization for face recognition using global and local textures. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):102–113.

Yang, J., Lu, W., and Waibel, A. (1997). Skin-color modeling and adaptation. *Lecture Notes in Computer Science*, pp. 687–694.

Yoon, Y., Lee, S., Chung, C., and Kim, S. (2008). An effective defect inspection system for polarized film images using image segmentation and template matching techniques. *Computers & Industrial Engineering*, 55(3):567–583.

Zhang, J., Marszalek, M., Lazebnik, S., and Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238.

Zhao, D., Liu, D., and Yang, Y. (2008). A new stereo matching method based on sub-pixel corner detection. In *Proceedings of the 2008 International Conference on Computer Science and Software Engineering-Volume 02*, pp. 899–902. IEEE Computer Society.