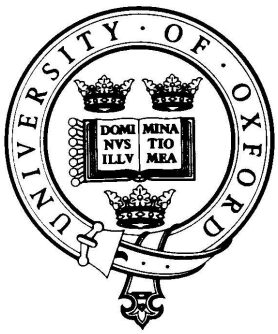
Visual Object Category Recognition

Robotics Research Group
Department of Engineering Science
University of Oxford



Supervisors:
Professor Andrew Zisserman
Professor Pietro Perona

Robert Fergus
New College

December 2, 2005

Abstract

We investigate two generative probabilistic models for category-level object recognition. Both schemes are designed to learn categories with a minimum of supervision, requiring only a set of images known to contain the target category from a similar viewpoint. In both methods, learning is translation and scale-invariant; does not require alignment or correspondence between the training images, and is robust to clutter and occlusion. The schemes are also robust to heavy contamination of the training set with unrelated images, enabling them to learn directly from the output of Internet Image Search engines.

In the first approach, category models are probabilistic constellations of parts, and their parameters are estimated by maximizing the likelihood of the training data. The appearance of the parts, as well as their mutual position, relative scale and probability of detection are explicitly represented. Recognition takes place in two stages. First, a feature-finder identifies promising locations for the model's parts. Second, the category model is used to compare the likelihood that the observed features are generated by the category model, or are generated by background clutter.

The second approach is a visual adaptation of "bag of words" models used to extract topics from a text corpus. We extend the approach to incorporate spatial information in a scale and translation-invariant manner. The model represents each image as a joint histogram of visual word occurrences and their locations, relative to a latent reference frame of the object(s). The model is entirely discrete, making no assumptions of uni-modality or the like. The parameters of the multi-component model are estimated in a maximum likelihood fashion over the training data. In recognition, the relative weighting of the different model components is computed along with the model reference frame with the highest likelihood, enabling the localization of object instances.

The flexible nature of both models is shown by experiments on 28 datasets containing 12 diverse object categories, including geometrically constrained categories (e.g. faces, cars) and flexible objects (such as animals). The different datasets give a thorough evaluation of both methods in classification, categorization, localization and learning from contaminated data.

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Robert Fergus, New College

To my parents

Acknowledgements

I would like to thank my two advisers: Professor Andrew Zisserman at Oxford and Professor Pietro Perona at Caltech for their guidance, patience and advice. This thesis has been a thrilling and rewarding experience thanks to them.

I also thank Fei-Fei Li being my main collaborator over the last 5 years, both in classes and research.

Many other people who have offered advice and guidance with my work for which I am very grateful (in alphabetical order): Andrew Blake, Mark Everingham, David Forsyth, Alex Holub, Dan Huttenlocher, Michael Isard, Jitendra Malik, Silvio Savarese, Josef Sivic, Frederik Schaffalitzky.

I would also like to thank all the people in the Vision Labs at both Caltech and Oxford for making them such interesting places to be.

I thank the various sources of funding I have had over the years: the Caltech CNSE, the UK EPSRC, EC project CogViSys and the PASCAL project.

Agnes deserves special thanks for being so supportive of my efforts and so understanding of the endless deadlines.

A final thanks must go to Pietro, Markus Weber and Max Welling who supervised me as a summer student while I was an undergraduate, sparking my interest in computer vision and object recognition.

Contents

Chapter 1

Introduction

1.1 Objective

Our goal is to be able to identify object categories within images. Despite the concerted effort of researchers over a number of decades this objective has remained, for the most part, unsolved. Although reasonably successful attempts have been made for certain classes of objects, such as human faces, no satisfactory methods exist that work with any category of object.

It is important to make the distinction between the recognition of specific instances of objects and classes of objects. For example, in Figure 1.1(a) & (b) we see two images of the Eiffel Tower and a Ferrari Enzo respectively. Despite changes in the background and viewpoint of the image, they are still the same object. In contrast, in Figure 1.1(c) & (d) show two instances of motorbikes and houses respectively. In both cases, not only does the background and viewpoint change between the two examples, but the object itself is different. However, each class is described by a single noun, reflecting the fact that *visual consistency* exists between the examples of each class.

This thesis develops methods that can automatically learn the visual consistency between exemplars of any object class, training a model that may be used to recognize novel instances in query images. The emphasis is on minimising the amount of human assistance required to learn a model of each class — an essential requirement if systems are to scale to thousands of classes.

(a) (b) (c) (d)

Figure 1.1: (a) & (b) Two instances of two specific objects (the Eiffel Tower and the Ferrari Enzo). (c) & (d) Two examples of two classes of object (motorbikes and houses).

1.2 Motivation

A person can recognize visually more than 10,000 categories of objects [12]. For humans, visual recognition is fast, effortless and robust with respect to viewpoint, lighting conditions, occlusion and clutter. Moreover, learning new categories requires minimal supervision and a handful of exemplars. Achieving this level of performance in a machine would enable a great number of useful applications.

Thanks to the low cost of digital cameras and video recorders, visual data may be cheaply captured and stored in electronic form, convenient for processing by computers. The problem is that computers cannot see very well, at the present time: they are unable to interpret the coloured pixels in the images into the higher level representations as humans do. Describing an image by its contents is in many cases far more useful than the original pixel representation, which is why our visual system is so good at recognition. There are many applications of machine recognition in our information-rich age.

- Video search — Many TV stations stream onto the Internet 24 hours a day, 356 days a year. This gives a huge body of data that at present cannot be searched unless someone manually watches it and annotates it with text labels. If automated techniques existed, then one could easily search the video data, for example to find your favourite show or actor.

- Web search — Internet image search engines are currently the only way to find images on the Internet. Their poor performance is due to their use of the image filename or surrounding HTML rather than the actual image content. The natural way to find images is to search visually. As it is difficult to specify a visual query directly, the user might select a few examples, similar in nature to the desired image. From these a visual model would be trained that could then be used to perform the search. Such a scheme would require the ability to learn a model quickly from a few examples.

- Searching image databases — Many people have thousands of home photos on their computers which are only loosely organised. Without manual annotation, the images cannot be searched to find all instances of the family dog, for example. Many companies also have large archives of images which they wish to search.

- Online dating — Online dating websites only have the facility to search for people based on information manually entered by people (e.g. hair colour, eye colour). A more useful form of search would utilize the photo placed online by each person subscribing to the website. For example, on finding an attractive person, you could search for people that looked like him or her. This requires building models of people that you (or people with similar tastes to you) find attractive.

- Security — Looking for people or vehicles in video streams from security cameras. Many major cities have thousands of close-circuit TV cameras whose footage must be manually scrutinised. Automated surveillance systems do not get tired or distracted unlike humans who currently monitor such footage.

- Airport baggage screening — People currently examine airplane passenger luggage using x-ray machines. The large number of bags examined per hour and the difficulty in interpreting the x-ray images mean the false negative rate is high. Recognition systems designed to find knives and guns might be able to assist the human operator, boosting safety.

- Car safety systems — Automobile manufactures are interested in making their cars aware of other cars or people close by and about to enter the path of the vehicle. This would

enable the car to alert the driver or to take evasive action by itself, so lowering accident rates.

While there are many applications for recognition, no practical solutions exist. This is due to the difficulties inherent in the problem, which we now examine.

1.3 Challenges

Successful approaches to object recognition must address a variety of problems:

- Changes of aspect. Different views of an object can look very different, as shown in figure 1.2.

- Changes of viewpoint. Objects can also be subject to in-plane transformations (translation, rotation, scaling, skews) and out-of-plane transformations (foreshortenings) that change their appearance. However, some viewpoints may be more likely than others (i.e. motorbikes are rarely vertically orientated) and this prior knowledge may be exploited.

- Illumination differences. A change in the lighting of the object will change the pixel values in the image. The change could be a shift or scaling of the pixel values or, if the light source changes position, a non-linear transformation, complicated by shadows falling on the object. The images in figure 1.3 illustrate examples of drastic changes in lighting.

- Background clutter. In the majority of images it is rare for the object to be cleanly segmented from the background. More typically the background of the image contains many other objects (other than the one of interest), which distract from the object itself. The images in figures 1.2 and 1.3 have cluttered backgrounds.

- Occlusion. Some parts of the object may be obscured by another object, as illustrated by the monkeys in figure 1.4. Additionally, as the aspect changes, one part of the object may hide another. This is known as self-occlusion.

- Intra-class variation. As in the car example of figure 1.5(a), the category itself can have a large degree of visual variability. The variability can take various forms: in the geometry, appearance, texture and so on. Also, one instance of an object may have features that

are missing on another (e.g. the radiator grille on the cars of figure 1.5(a)). Examples of these different forms can also be seen in figure 3.1. This is the main problem addressed in this thesis.

Figure 1.2: Variation due to changes in aspect.

Figure 1.3: Variation in appearance due to a change in illumination

Figure 1.4: Some examples of occlusion

What we mean by an object category also requires definition. There are two types: *visual* and *functional*. In the former, the objects are related by some kind of visual consistency, be it in the outline or the appearance of the object. In contrast, a functional category is one that is related by its purpose, for chairs, as shown in Figure 1.5. Functional categories cannot be modelled using visual information alone and since the methods in this thesis use only visual cues, we limit ourselves to visual categories.

(a) (b)

Figure 1.5: (a) Some examples from a visual category. (b) Some examples from a functional category. Note the wide variation in appearance of the functional category.

1.4 Definition of vocabulary

Recognition is a multi-faceted problem. Given an image, we could ask: is a certain object class present in an image or not? If so, where are the instance(s) located? Alternatively, which objects classes are present in the image? Many terms related to recognition are used in a somewhat loose manner in the literature so to avoid confusion we give definitions of the terms used in the thesis:

- The words *class* and *category* are used interchangeably, referring to a visually consistent set of objects (as opposed to a functionally consistent set).
- Classification (of image) - the task of deciding if at least one instance of an object class is present within an image or not (object present/absent).
- Categorization - the task of deciding what class the image belongs to out of many possible classes, for example when computing a confusion table. Thus while classification is a two state problem, categorization is an n state problem, n being the number of classes.
- Localization - the task of specifying the location within the image of all instances of an object. In this thesis bounding rectangles will be used, as opposed to pixel-level segmentations.
- Detection - the same as localization, in keeping with the use of the term in the face recognition literature (e.g. the face detector of Viola and Jones).

- Recognition - this term is used generically to refer to the problem as a whole (i.e. any of the set of classification, categorization or localization).

1.5 Contribution

In this thesis we develop and discuss two probabilistic models for an object category which can be learnt from a set of training images of instances of that category, requiring only weak supervision. Each model represents a single visual aspect of the object category (e.g. the side view of a car) — although both models can be extended to multiple views. The training images are required to contain a single instance of the category with a common visual aspect and orientation. These requirements are the extent of the weak supervision. The instances do not need to be aligned (e.g. centred) or scale normalized or put in correspondence, and the images may contain clutter (i.e. foreground segmentation is not required). Additionally, neither approach makes use of colour information meaning the schemes cannot rely on crude colour cues when performing recognition. We now introduce the two approaches.

1.5.1 The Constellation Model

The first approach developed in this thesis is an extension to the *constellation model* of Burl *et al.* [21] and Weber *et al.* [111]. This is a "parts and structure" model, as introduced by Fischler and Elschlager [43] in 1973. The model consists of a number of parts whose appearance and relative location are similar across instances. For example, in figure 1.6, two otherwise different cars have three similar parts, highlighted by the coloured circles. Note the similarity both in the appearance of the parts and their relative positions.

The proposed model explicitly accounts for appearance of the parts; their relative location; relative scale; the presence/absence of features due to occlusion and detector errors and image clutter, all in a probabilistic manner. Efficient methods allows optimal matching between the model and the image, enabling correspondences between different instances to be established. Both learning and recognition can be performed in a scale-invariant manner.

The input to the model consists of features which are either regions of pixels or curve segments. These two complementary types of feature allow both the appearance and geometry of an object to be represented, meaning a wide range of categories can be modelled. The

algorithm automatically chooses the most appropriate feature types for each category.

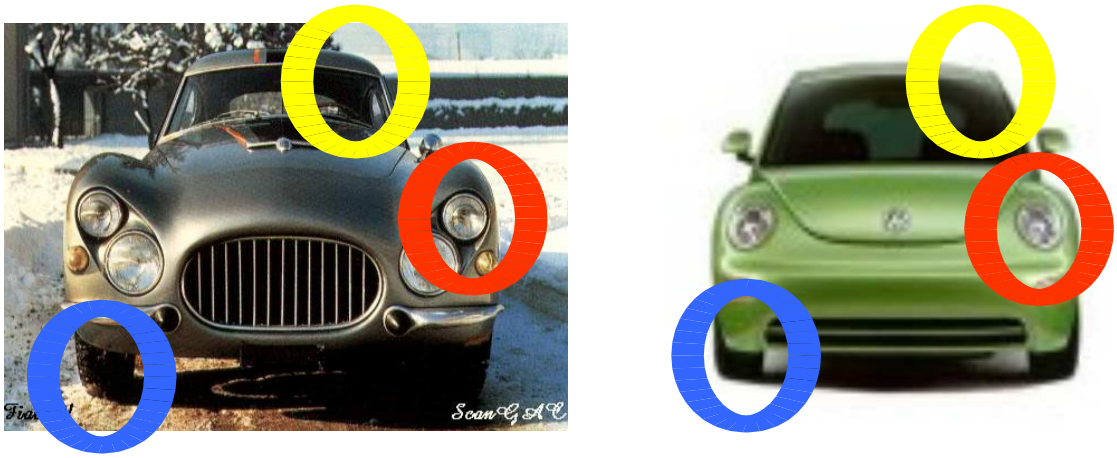


Figure 1.6: Two images containing cars that look dissimilar. The coloured circles indicate regions of both instances that are similar in visual appearance and relative location.

1.5.2 Translation and Scale-Invariant pLSA (TSI-pLSA)

The second approach is based on *probabilistic latent semantic analysis* (pLSA), a method from the text community [52] where this technique is used to extract coherent components from a collection of documents. This problem has many parallels with learning visual models with little or no supervision. Indeed, pLSA was applied to computer vision by Sivic *et al.* [100] and Fei-Fei and Perona [34]. However, both of these approaches make no use of location information in the image, relying solely on the appearance of a set of regions extracted from the image. We extend their work to include location information in a scale and translation invariant manner, calling the approach TSI-pLSA.

The model is able to learn in a totally unsupervised manner (as demonstrated by Sivic *et al.* [100]) and uses a discrete representation of both appearance and location, making no assumptions of Gaussianity or the like. Images are represented by histograms of vector-quantized regions, using many hundreds or thousands of them - in contrast to the sparse approach of the constellation model above.

1.6 Outline of the thesis

The structure of the thesis is as follows: In chapter 2 we review existing work in the field of recognition, detailing previous versions of the constellation model. We explain the operation of the feature detectors (pixel patches and curve segments) that produce the input to both models. Chapter 3 introduces the variety of datasets used in the thesis to assess both methods, each dataset being designed to test a different aspect of performance. In chapter 4 we give the form of the first approach, the constellation model, discussing the assumptions the model makes. The solid mathematical basis of the model will become apparent. We also explain how scale and translation invariance may be achieved. Finally, we review the improvements made to the constellation model over previous incarnations. In chapter 5 we give details how the constellation model is used in learning and recognition. We elaborate on the maximum-likelihood learning process, followed by a discussion of its behaviour. Chapter 6 evaluates the constellation model on the Caltech datasets. A series of diagnostic experiments are performed to elucidate the behaviour of the model. Chapter 7 introduces the second approach, TSI-pLSA, showing how it builds naturally from existing work. Experiments are performed on some simple datasets to show the operation of the model. In chapter 8 both the constellation model and TSI-pLSA are tested on more challenging data, containing greater background clutter and a wide variation in viewpoints and aspects. Comparisons are drawn to other competing approaches. Chapter 9 investigates the algorithms' performance on images collected from Internet Search Engines, operating in an unsupervised manner. In contrast to the weakly supervised experiments, the labels of the training images are no longer provided thus we cannot be sure that they contain an instance of the object to be learnt. Finally, in chapter 10 we draw conclusions about the two approaches and discuss future work.

Chapter 2

Literature review

The first attempts at object recognition began in the 1960's. Accordingly, there is an extensive body of literature on the subject which we now review. Progress over the last forty years may be judged by the increasingly realistic data used in experiments; early work concentrating on finding specific instances of objects, with category level recognition being addressed later. In 1970's range data was used since it directly gives 3-D information. In the 1980's images were used directly but the objects were typically on a light-table or uniform background, simplifying the segmentation of the object. Methods were then developed that could handle natural images, the emphasis being recognizing a single object instance, from a variety of viewpoints. The first category-level work in the 1980's targeted a limited set of classes such faces and digits, typically in constrained environments. Work from the late 1990's onwards has tackled a wider variety of classes in increasingly natural image environments, often constraining the viewpoint to simplify the problem.

Reflecting the historical development of the field, we first review the literature on single object recognition and then more recent work on categorization. The chapter is completed by a review of feature detectors and representation schemes which are used in many approaches to recognition, including those presented in this thesis.

When considering each paper, important issues to consider include: what, if any, intra-class variability can be handled; what viewpoint change, if any, can be accommodated; how does the approach cope with cluttered backgrounds; can it localize the object within the image; is it sensitive to illumination changes; what level of supervision is required to obtain the model in

the first place; what level of occlusion can be handled and how general is the approach — does it scale to hundreds of different objects?

2.1 Specific instance recognition

While not directly relevant to this thesis, we can draw some useful conclusions from work on specific object recognition. By not having to consider intra-class variability, the problem is somewhat simplified: the variability in appearance is reduced to five sources: viewpoint change; differences in illumination; occlusion; background clutter and imaging noise (usually small).

Many early approaches used a geometric representation, extracting edge contours from the interior and exterior of the object. This representation is convenient in that it tackles two of sources of variability: it is insensitive to illumination changes and makes the determination of 2-D or 3-D pose relatively straightforward.

More recent work extracts a set of textured regions from the image and uses these to find the object and its pose, largely ignoring edge information. We review the two approaches in turn.

2.1.1 Geometric methods

The first forays in recognition took place in the 1970's and 1980's, focusing on range data [3, 13, 15, 17, 88]. Range images store the depth of the scene, rather than intensity, at each pixel thus make it easier to extract the 3-D shape from the image. Measures such as Gaussian curvature [88] can be used to extract edges and segment the image into planar regions that can be matched to a 3-D CAD model.

The field then advanced to working directly with normal (i.e. intensity-based) images. A variety of geometric approaches were then introduced, which include: (i) alignment techniques where putative matches are made between model and image and searched over to find the best match and (ii) geometric invariants where small sets of points are used to compute a viewpoint-independent descriptor which can act as a key for hashing into a database where models are stored.

In addition to these broad groupings, there are also methods like skeleton-based representations; aspect graphs and geometric primitives (such as ribbons) which we do not concern

ourselves with here. We also do not consider stereo systems such as the disparity gradient approach of Pollard *et al.* [87] where 3-D wire-frame models are reconstructed and then used for recognition.

All these papers are attempting to recognize specific instances of 3-D objects from 2-D views. The models with which to perform recognition are either, in the case of early work, hand built 3-D CAD models or, more recently, extracted directly from the image under controlled conditions. In recognition, the challenge is to make the matching procedure robust, given the difficulty in extracting edge information reliably (e.g. edge segments get fragmented).

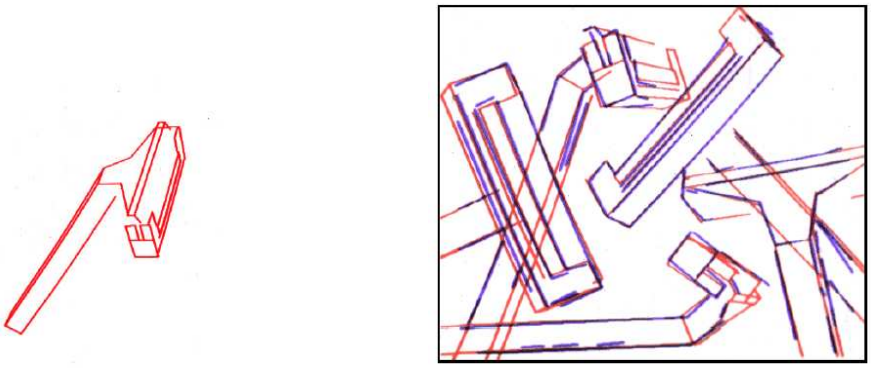
Alignment techniques

Alignment techniques consist of two main stages: first, a correspondence between the model and the image is proposed, using a set of lines or points to hypothesise a transformation; second, a verification stage is used to check the support of the proposed location using edge information from the image. Searching over all possible correspondences is prohibitive due to there being an exponential number of them. Consequently, an interpretation tree [48] is explored to find the best match, guided by constraints that allow aggressive pruning of the tree.

A typical paper to make use of alignment techniques is the SCERPO system of Lowe [73] which first extracts lines from a query image, then groups them using co-linearity and parallelism criteria. The 3-D CAD model is projected down onto the various groups of lines by solving for the unknown viewpoint to give a correspondence. By using subsets of lines within the model, rather than all of them (the global approach), robustness to occlusion is achieved. The 3-D model and recognition results are shown in Figure 2.1.

Huttenlocher and Ullman [54] attempt to recognize 3-D objects for which they have an existing model, obtained from an image as opposed to a CAD model. The model is projected down into the image, by hypothesising correspondence between 3 pairs of model and image points, giving an unambiguous affine transformation. An interpretation tree is used to explore possible matches, comparing the edge contours of the model with edgels chains from the image to validate the match. They demonstrate their method on objects with occlusion in cluttered scenes.

Ullman and Basri [106] represents a 3-D model with a mixture of 2-D models and matches



(a) (b)

Figure 2.1: Figures from Lowe's [73] SCERPO system. (a) The 3-D wire-frame model of a razor shown from a single viewpoint. (b) Successful matches between sets of image segments and particular viewpoints of the model.

this mixture model to the image using lines and points. This raises the interesting issue of how to model a 3-D object. Should, as in Ullman and Basri [106], a combination of 2-D models be used, or is a true 3-D representation (as in Lowe [73], for example) better? Human vision scientists also debate which representation the brain uses: the Geons as proposed by Biederman [11] — a set of 3-D primitives (like cones or cylinder or cuboids) that all 3-D objects may be decomposed into; or the view based representation explored by Reisenhuber and Poggio [90].

Other systems using alignment include the work of Mundy and Heller [81] where 3-D CAD models were used to find objects from aerial imagery by clustering pose estimates from edge data.

Geometric invariants

An important contribution was the application of hashing to recognition, as described in the papers of Lamdan and Wolfson [62] and Rigoutsos and Hummel [91]. These methods use a visual analogy to text-based hashing. Object models consist of sets of interest points. The sets are made invariant to affine transformations by using three points from within the set as a basis. In learning, for each object, all possible triplets of basis points are used, with the remaining points for each triplet basis stored in a hash table. During recognition sets of interest points are extracted from the image and used to index into the hash table, voting for different object models. Recognition thus proceeds by counting the number of votes for each object. The redundant representation of each object within the hash table allows its rapid retrieval

(since any choice of basis points will do) and gives robustness to occlusion. However, the false positive rate increases considerably in the presence of moderate noise or clutter in the data points. Rigoutsos and Hummel introduced a probabilistic voting scheme to address this issue. This geometric interpretation of hashing carries the advantage that it is very fast, since each object is being found by a series of hash lookups hence has a (nearly) constant lookup time with the number of objects stored.

Following on from these approaches, Rothwell *et al.* [93, 94] describe a recognition system for planar objects using projective invariants. A projectively invariant set of index functions is used to represent each object. Each index function is based on the geometrically invariant properties of a small group of points, lines or conics. Two forms of invariants are used: algebraic invariants use sets of lines and conics, which have been fitted to edgel data. The other invariant uses four points on the object to transform edgels chains into a canonical frame, so just leaving the shape of the object. The four points are found by looking for bitangent and tangent points on the curve segments (see Figure 2.2). The learning process involves computing these invariants and storing them in a library. In recognition, combinations of edges, lines and conics in the image are used to compute the index function, which is then compared to the library holding the index functions from all objects. Combinations of these invariants are explored using an interpretation tree. In a verification stage, each combination is projected down into the image and compared to the actual edgel data. Occlusion is dealt with by using many index functions for each object. A wide range of projective transformations can be dealt with due to the nature of the invariants. The key contributions of this work are: the invariant index functions; the use of curves as well as points and also full projective invariance.

Summary of geometric approaches

The strength of these approaches is their ability to perform 3-D recognition in an affine or projectively invariant way.

Methods like geometric hashing can provide both affine/projective invariance and quick recognition, scaling sub-linearly with the number of objects stored — important if thousands of objects are to be stored.

By relying on the outline of the object, the methods are relatively insensitive to the textured

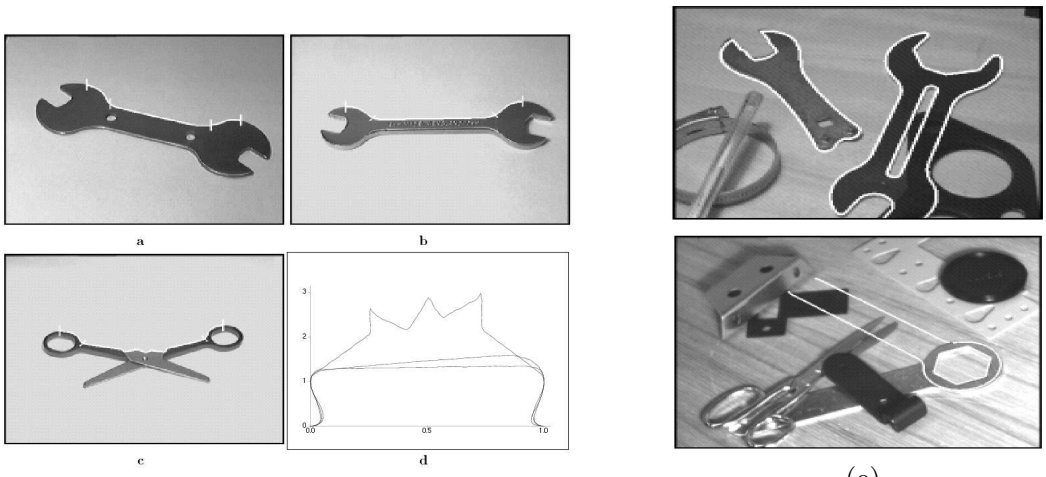


Figure 2.2: Images from Rothwell *et al.* [93, 94]. (a)-(c) show a curve on each object with bitangent and endpoints identified. (d) shows these curves projected into a canonical frame. The scissor curve is very different from the other two, but the two spanners can also be distinguished by the system. (e) shows recognition examples; top: two spanners being correctly localized in a clutter scene; bottom: an incorrect localization due to the textured background.

interior which, for certain classes, will be far more variable than its shape (e.g. bottles). This enables these approaches to model whole classes rather than specific objects. The flip side of this is that although contours may work well for certain classes like bottles, it gives limited representation for other types of objects, like faces, where raw intensity values of regions of the image would be more descriptive.

A major drawback to all the above methods is that they assume edge contours can be reliably found within the image. Extracting chains of edgels can be hard in natural images where there is extensive illumination differences, background clutter or occlusion. Some of the papers only test their algorithms on images consisting of objects on light-tables or a uniform background, so we are unable to assess them in more realistic settings.

2.1.2 Global appearance methods

There are also a variety of other methods which are worthy of note, since they are also applied to category level recognition.

Schiele and Crowley [97] use histograms over the joint statistics of local appearance, as measured by vectors of robust local shape descriptors. The histograms also incorporate the spatial location of the descriptors. Training consists of forming the histograms, which is done automatically. Three different approaches to recognition are evaluated. The best performing

one uses a probabilistic measure of the divergence between the test image histograms and the candidate model. Their algorithm can be regarded as a generalisation (to more than just colour) of the work of Swain and Bollard [103].

Murase and Nayar [82] use a 3-D version of eigenspace methods (described in Section 2.2.2 below) to recognize cars from the COIL database [83], estimating their 3-D pose. Images of each object are made over a wide set of viewpoints and each image projected into the basis of the principal components. The different views of the object form a manifold in the coefficient space. A novel view is then projected into the basis and its coefficients compared to manifolds from known objects, giving the identity of the object and its pose.

Pontil and Verri [89] use a support vector machine (SVM) on the COIL dataset. The SVM attempts to classify between pairs of objects by finding the optimal separating hyperplane between the data points. Each data point is a low-resolution version of each image, so it is essentially a global image-based method. This means that it is sensitive to occlusion, as demonstrated by some of their results. Background clutter would pose a problem for this approach, thus the SVM would need to be applied via a small window that is slid exhaustively over the image, if applied to larger images with background clutter.

The COIL database [83] has the drawback that all the objects are somewhat synthetic in nature, being a variety of children's toys and household objects and also that they are all centrally placed on black backgrounds. The segmented nature of the data means it is far easier to perform well on than natural images, indeed many approaches tested on the COIL dataset are likely to be adversely affected by background clutter were it present.

Overall, global methods, while having advantage of being simple, are sensitive to background clutter and occlusion. Although measures can be used to limit their susceptibility (such as using a set of sub-windows from the image), they waste modelling power on parts of the object than are not useful for recognition. Region-based methods, which we now review, have proved to give superior performance and are robust to clutter and occlusion.

2.1.3 Textured region methods

Above we saw how edge information from the image can be used to perform recognition. Now we look at alternative approaches which instead extract a set of textured regions and use these

for recognition. The general idea is to use a region detector to find a set of interesting parts of the image and then represent them with some kind of descriptor. Recognition proceeds by matching the descriptors from the image to those in a database of descriptors from previously observed objects, an object being found if sufficient matches occur.

If the approach is to be invariant to certain kinds of transformations, then both the way in which the regions are found and their representations must also be invariant to the desired transformations. Robustness to illumination change must be built into the description scheme.

Important questions to ask about such schemes are: what types of regions are used; what representation is used and how are the regions combined to perform matching?

The first to use a region based representation were Schmid and Mohr [98] who proposed a scheme for specific object recognition. Interest points are found in the image using the Harris interest operator (see Section 2.3.4). For each point a vector of local gray-scale measures which is invariant to rotation is calculated. By calculating these measures over a variety of scales invariance to scale is obtained. In training, the vectors from each interest point in the image are stored in a hash table. Recognition works by again finding the interest points and their associated vectors and then looking them up in the hash table. A voting scheme, using the mutual positions of interest points within the image, is used to make the final classification. The representation of relative position was invariant to rotation and scale but not affine transformations. Since many points are used in the voting scheme, the system can tolerate a portion being missing and clutter will produce interest points that do not match to any image. The scheme is tested on both the COIL database [83] and aerial imagery.

In a landmark paper Lowe [71] presented a robust and flexible quasi-affine invariant recognition scheme that can operate in real time. We explain its operation in detail since its design has much that is applicable to category level recognition. In recognition, a difference of Gaussians feature detector (described in Section 2.3.3) is used to extract a large set ($\sim$1000) of regions from the query image. Each of these regions are then represented as a SIFT vector, described in Section 2.3.7, which is similarity invariant.

An approximate nearest neighbour scheme is then used to match the vectors from the image to other vectors from objects the system has been trained with. The matching time grows logarithmically with the number of vectors in the database and has been tested upto 10^6 vectors

while still delivering realtime performance. The vectors matched to a certain object vote in a Hough-based scheme for the pose of that object in the image. If sufficient votes are recieved for an object in a given pose, it is indicated in the image. The robustness of this scheme comes from the fact that a large number of features will be found on the object, so if quite a few are missing through occlusion or poor illumination, a sufficient number will remain for successful matching to occur. Figure 2.3 shows an example of the system in action.

(a) (b) (c)

Figure 2.3: The object recognition system of Lowe. (a) Objects stored in the system. (b) Query image. (c) Image with object superimposed. The small rectangles are the features matched to the object models.

The success of the scheme shows that many objects can be well represented as a collection of regions, something not intuitively obvious. The robustness of the SIFT descriptor and the speed of the approximate matching scheme are key to the systems performance. In extensive matching experiments, the SIFT descriptor has been shown to be superior to many other types [77].

Many subsequent papers have refined the concept [42, 75, 78, 80, 96, 92] to improve performance, for example by moving from similarity to affine invariance or incorporating information about the relative locations of vectors within the image into the matching process. However, none have managed to extend it to object categories. The highly specific matching of the features, vital when matching specific objects, is a hindrance when trying to accommodate intra-class variability as it prevents genuine matches.

In summary, region-based methods seem to be superior to the geometric methods discussed previously, with their ability to operate in highly cluttered environments on an admirably diverse selection of objects. In defence of geometric methods, they are capable of greater degrees of invariance which in some environments may be important. Additionally, if the texture on

the object were to change then geometric approaches would be unaffected while region-based methods would fail. Region-based methods have the key advance over global approaches that they can handle occlusion and clutter (without the need for segmentation). The most important message is that objects can be modelled as a collection of regions found by low-level interest operators — an idea that underpins the methods proposed in this thesis.

2.2 Category level recognition

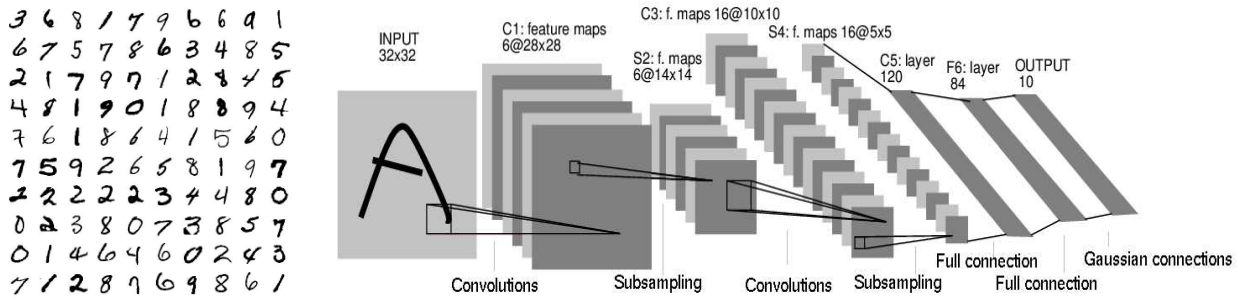
In contrast to the specific object recognition in the previous section, we now look at the problem of recognizing object categories. This is a harder task due to the fact that each instance of the class is no longer identical, hence the matching scheme used must have a way of accounting for the variability across instances in the contours or regions extracted. This may be done either by explicitly modelling it with a probability density function or implicitly with a robust matching scheme.

Historically, work on category level recognition has been concentrated on a small set of objects: handwritten digits, faces, humans and cars. This focus results from the practical importance of these classes. However, many methods applied to these classes can be successful applied to others. Therefore we start our review by looking at the early publications on digits, faces and cars, before covering more recent work on general purpose algorithms for many classes.

2.2.1 Digits

Handwritten character recognition has mainly focused on digits rather than cursive script with a few notable exceptions [63] since the former is far easier and suffices for important applications like postal ZIP code reading and automated cheque processing. Each of the ten digits (0—9) is usually treated as a separate class (4's and 7's sometimes have two separate classes as there are two different writing styles for each). Most digit recognition algorithms assume that the digits have already been segmented from the image and are presented to the algorithm as a small grayscale image with black ink on a white background (e.g. the MNIST dataset [76], samples of which are shown in Figure 2.4(a)). The preprocessed and constrained nature of the data means that the relevance of digit recognition algorithms to more general object recognition is limited, although we will describe some methods that are more generally applicable.

A prominent approach is the convolutional neural network (CNN) of LeCun *et al.* [63]. The CNN is a large neural network where the input is the raw pixels from the (small) image and the output is the class label. Aside from the last couple of layers which are fully connected, the layers in the network alternate between sub-sampling and convolution operations. See Figure 2.4(b) for an illustration of the architecture. The design means that the size of the receptive field at each layer increases towards the latter layers of the network. Hence, if portions of the digit are occluded or missing it will affect only the earlier layers and hopefully not the final output. Training consists of learning the weights in the network by back-propagation, using a set of



(a) (b)

Figure 2.4: (a) Some scale normalized examples from the MNIST dataset [76]. (b) The LeNet convolutional neural network from LeCun *et al.* .

labelled training digits. Training is discriminative in that the error is minimised across all classes simultaneously. A downside of the system is that it requires a large number of training examples (10^5 examples, due to the number of weights in the network being 10^5–10^6, although some are shared). The system is very fast in recognition since it just consists of evaluating a neural network. LeCun *et al.* report some of the lowest error rates on the MNIST dataset, against hundreds of other approaches. In [64] CNNs are applied to categories other than digits, we discuss this in Section 2.2.3.

Amit *et al.* [4, 5] propose a parts and structure model, looking for groupings of edge fragments in stable relative positions and then further grouping these on a global level to form the object model. The hierarchal grouping structure gives a redundancy which lends robustness to occlusion and clutter. Training consists of finding edge groupings that are common in the object category of interest, but rare in other images. The algorithm requires correspondence to be established between parts of the object in the training images. The mean location of edge groups and the distance between them may be used to normalize out scale and translation from

the configuration, resulting in scale and translation invariance. However the experiments show a high false alarm rate — maybe due to the very simple nature of the lowest level features. Human faces, distorted and transformed Latex characters and handwritten digits are used to evaluate the algorithm.

2.2.2 Faces, Cars and Humans

Amongst natural image categories, faces have received much attention. The earliest approaches were relatively simple, due to the meagre computational resources available. One of the most popular was Principal Component Analysis (PCA) otherwise known as the Karhunen-Loeve transformation.

Sirovich and Kirby [59], Turk and Pentland [105] and Murase and Nayar [82] all used PCA in some form to tackle the face recognition problem, using tightly cropped face images. These algorithms extract information from the image by transforming from a high dimensional space (the pixels of the image) to a low dimensional space (eigenpictures) — see Figure 2.5. Thus each image is represented by a small number of coefficients making searching and comparisons easier. Belhumeur and Kreigman [9] used Fisher's Linear Discriminant, a method of dimensionality reduction similar to PCA but discriminative in nature, to recognize faces. Both approaches have some drawbacks on account of the image being modelled globally:

- Any change of pixel values, caused by illumination changes or a transformation in position will change the eigenvector representation of the image.
- Background clutter will cause problems since it will also change the eigenvector representation, as the whole image is treated equally. In the papers, the images are all fairly tightly cropped, so they avoid this problem. Turk and Pentland use a Gaussian to de-emphasize the background of the image.
- Occlusion will also cause similar problems: if we have partial occlusion of the object, this must be represented within the eigenspace, so altering the representation of the image. Belhumeur and Kreigman use the Fisher linear discriminant to maximize the ratio of the between-class variance to the intra-class variance. They show some limited results that might suggest robustness to occlusion due to this refinement.

Consequently, all the PCA methods assume that the face is segmented from the image — a major preprocessing requirement. One way to circumvent this is to take exhaustively extract all sub-windows over location (and scale and rotation if required) from a large cluttered image and evaluate each one in turn. Thus one of the sub-windows is guaranteed to contain the face with little background clutter. Many other approaches which act globally over a sub-window use this method.

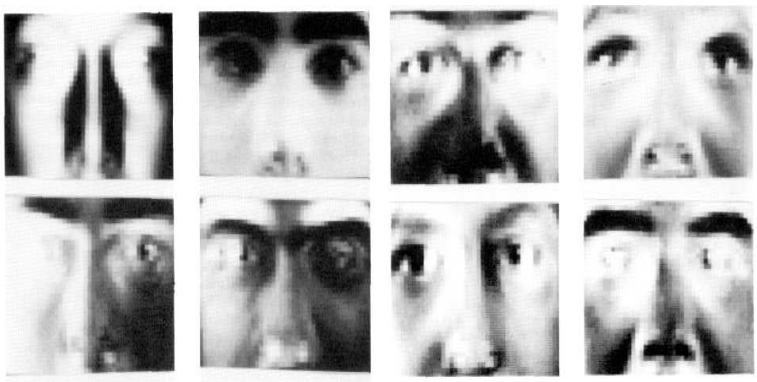


Figure 2.5: The first 8 Eigenfaces from Sirovich and Kirby [59].

Fischler and Elschlager [43] in 1973 were the first to propose what is often termed the "parts and structure" model, where the model consists of a series of small templates (the parts) arranged in some geometric configuration (the structure). When fitting the model the aim is to minimize a cost function which comprises the local fit for each of the parts plus a global deformation term, measuring the deviation from some rest position. The deformation term can be thought of as a series of springs connecting the individual parts, see Figure 2.6 for an illustration of how a face might be modelled. In their paper they attempted face recognition, but achieved limited results due to the low resolution images and the lack of computational resources. Irrespectively, the important contribution was the idea — individual parts linked by some spatial model.

Another neural-network based approach is that of Rowley and Kanade [95] who apply it to frontal face detection. Their architecture consists of a series of receptive fields, each sensitive to a part of the image over a range of scales. These fields are then the input to the network. This allows the network to focus on certain parts of the face, giving robustness to occlusion. Training consists of showing the network a large set of cropped frontal faces images and a large set of non-face images. Detection proceeds by exhaustively running a sub-window over the image at

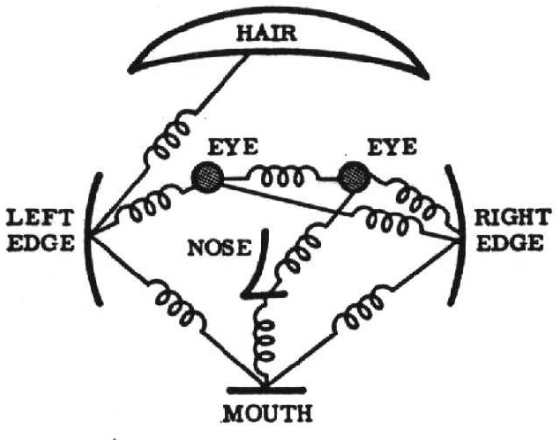
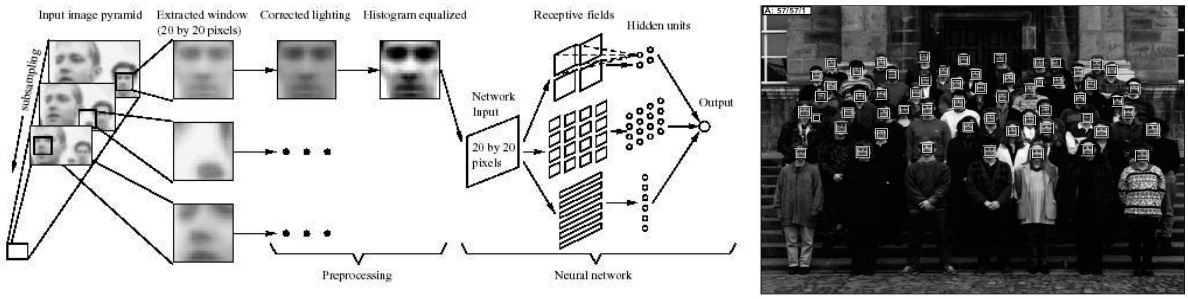


Figure 2.6: The parts of structure model of Fischler and Elschlager [43].

all locations and scales. They present some impressive results on images with a large amount of clutter, see Figure 2.7



(a) (b)

Figure 2.7: From Rowley *et al.* [95]. (a) Overview of the neural network based face detector. (b) Examples of many faces being found, with only one false alarm.

Schneiderman and Kanade [99] tackle the problem of face and car detection. A multi-view capability is obtained by having a separate model for each viewpoint of the object. Their method uses wavelets to perform a multi-scale analysis of cluttered scenes containing cars and faces. A classification decision is made in a probabilistic way using the histograms formed from the wavelet coefficients of the image. It is not obvious how robust this representation is to clutter and occlusion: occlusion is likely to disrupt the low-frequency portions and part of the high-frequency coefficients, but the majority of the high-frequency components should be unaffected. They present convincing results to support their approach, see Figure 2.8 but it is hard to gain an intuitive feel for what the wavelet transform is doing.

Viola and Jones [107] look at the problem of real-time face detection. They use a series of

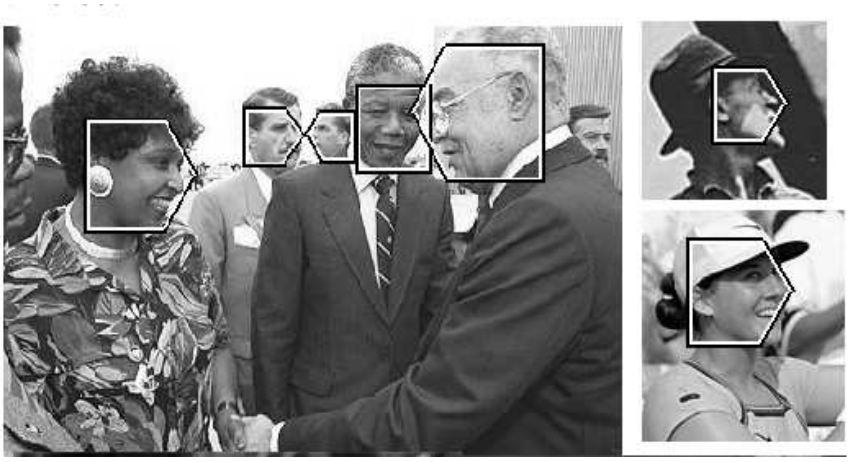


Figure 2.8: Examples of faces being found using the method of Schneiderman and Kanade [99].

very simple features (based on integral images) which are fast to compute and combine them in a cascade of classifiers. The cascade structure means that the entire set of classifiers is only applied to a few promising locations; the majority being discarded after a few levels of the cascade. The simple features, coupled with the cascade structure give the speed necessary for real-time performance. Training images are a large set of tightly cropped images of faces ($\sim 10^5$) and an even larger set of non-face images. In learning the cascade of classifiers is constructed using AdaBoost and detection is performed by exhaustively running a sub-window over the image. While theirs is the first practical real-time face detection system, the performance is slightly worse than Schneiderman and Kanade. A major contribution of this paper is introducing boosting to the field of computer vision and many subsequent papers (covered in Section 2.2.3) have made use of it. Boosting is a way of combining a number of weak classifiers to make a strong classifier, which is a natural fit for problems such as recognition where it is easy to find a set of simple features each of which picks up some signal but is too weak by itself to be much use.

Support vector machines have been used for finding people in images. The generic approach in detection is to take a small sub-window and slide it over the image exhaustively, so avoiding issues of background clutter. However, the representation used within the sub-window varies: early approaches include Papageorgiou *et al.* [86] who used wavelets, while more recent work includes Dahl and Triggs [27] who use a gradient histogram approach.

Forsyth *et al.* [44, 45] look at the problems of horse and nude human classification. These are difficult due to the large amount of visual variation within the classes. Texture and colour-

based detectors are used to find areas of the image with a skin/hide like hue. Then these areas are combined into body-like forms using predefined rules as to what bodies of look like, giving "body plans". Since only portions of bodies are required for identification, most of the body can be obscured. It is designed to work in cluttered images: the skin filter removes most clutter and any skin-like background areas will hopefully not be grouped into the body plan. Unfortunately, the system does seem to have quite a large false alarm rate, but it does cope well with heavy occlusion. Examination of the results reveal that the body plan adds relatively little: the system mainly uses the skin/hide detector. Image noise is likely to have little effect on the algorithm since the system does not work with specific features, but rather areas of the image.

Mikolajczyk *et al.* [79] propose a probabilistic assembly of robust part detectors to find humans. Face, torso and limb detectors are trained together in a discriminative manner. The combination of detectors results in state-of-the-art performance, beating approaches that use face detectors alone.

Cootes *et al.* [24] use an active contour model to distinguish between the faces of different people, each being characterised by the parameters of the fitted model. A drawback of the approach is that it is unable to directly tell if a face is present in the image or not, hence is more suited to a finer level of recognition (i.e. which face is it? rather than is there a face here?) than many of the other face detection schemes covered in this section. Lades *et al.* [61] propose a scheme to distinguish people based on a deformable grid template. Each individual is characterised by the deformation energy at points within the mesh when fitted to the face.

2.2.3 Recent work

Despite the success of methods for car and face recognition, their design is tightly tuned to these particular categories. In contrast, a human can recognize around 10,000 categories [11] with equal ease. Many of these may not have the constrained form and distinctive features that make their representation easy as faces or cars do. For example trees or leopards are both valid classes but are difficult to model: one is an amorphous form while the other has a distinctive form of articulation. Recently there has been much interest in creating algorithms that are applicable to all categories, with no category specific tuning required. The emphasis has been

on modelling the intra-class variability to capture the essence of each class. To simplify the problem many of the approaches require a relatively constrained viewpoint in both learning and recognition, although most can be extended to 3-D recognition by using a series of 2-D models in the manner of Ullman and Basri [106]. This is reflected in the nature of the datasets used for evaluation in many papers, such as the Caltech datasets [37] or the UIUC datasets [2].

All these papers represent the object as a collection of textured patches, purely geometric methods being currently out of fashion, although the details vary widely: the number (from a few to thousands), how the parts are detected and represented, how their position is represented (if at all); whether the variability in appearance and position is represented explicitly or implicitly within the matching algorithm. There are many approaches to learning: some methods are generative, others discriminative, with hybrids in between. As discussed above, for many categories, it is not obvious what features of the object are distinctive and therefore many of the algorithms use some kind of feature selection to make this choice automatically. In recognition, some slide a sub-window exhaustively over the image, while others use the regions themselves to define a coordinate frame. Some methods are able to localize the object(s) within the frame, while others classify the image as a whole.

Appearance only models

A straightforward but powerful approach is the "Bag of Keypoints" model of Csurka *et al.* [26]. In training, a large set of regions are extracted from each image and vector quantised to a set of predetermined clusters. The image is then described by a feature vector listing the number of regions belonging to each cluster. This vector is labelled according to their class (object present/ object absent) and, along with vectors from all other training images, used to train an SVM. Recognition proceeds in the same manner, except that in the last step, the SVM is used to predict the class label. They evaluate their algorithm on the Caltech datasets [37] and others, in a classification role (the algorithm being unable to localize) achieving excellent performance, despite the lack of any location information in the model. This is surprising as intuitively, location information would seem to be an important part of recognition.

Another paper which only uses appearance information is that of Dorko and Schmid [29]. In training, regions are extracted from the training images and clustered using EM. For each cluster

a score is computed, measuring its ability to discriminate between foreground and background classes. The top few clusters based on their discriminative score are then selected to form a final classifier. In recognition, regions are extracted from the query image and assigned to the selected clusters or to remaining background ones. A simple threshold on the number assigned to the selected clusters is used to perform classification. Excellent results are achieved on the Caltech datasets. The paper makes use of a hybrid generative/discriminative scheme in learning: the clustering of commonly occurring features, followed by a discriminative procedure to find the clusters that are distinctive of the class. The latter stage is important since low-level features like corners and edges occur very often but carry little information about the class label.

Opelt *et al.* [84] adopt AdaBoost, using it to select a set of weak classifiers which combine to make up a classifier for the given category. The weak classifiers are regions in the space of appearance descriptors (they use several, including SIFT and moment invariants), the centre and size of the region being determined automatically. This discriminative scheme preforms so well on the Caltech datasets that they introduce a further sets of more challenging images for motorbikes, bicycles and people (known as the Graz datasets [85]). A problem with the algorithm is that many of the weak classifiers used seem not to be on the object itself, indicating that the statistics of the background of the images in the positive class are being modelled rather than the object itself.

In summary, all these approaches perform very well while relying entirely on appearance information. The lack of spatial information makes it easy to combine information from hundreds or thousands of features, making the classifiers robust to occlusion and other noise sources. Additionally, if multiple viewpoints are presented to the algorithm, in learning the classifier will just merge the distributions of appearance from each view, resulting in viewpoint invariance. The negative aspect is that it is difficult to localize the object within the image using these methods since the locations of the features are not used.

Methods incorporating shape information

At the other end of the spectrum, a variety of papers use an explicit shape model that has a relatively small number of parts (< 100), being limited by the computational complexities of incorporating this shape information into the model. Herein lies the challenge: how to get

sufficient location information into the model to be useful without introducing computational complexities that limit the number of regions that can be used.

The original idea of a parts and structure model, introduced by Fischler and Elschlager [43], is discussed in Section 2.2.2. A more recent implementation of this idea, taking advantage of modern processing power is the Constellation Model of Perona *et al.* [18, 20, 109].

Perona *et al.* look at the problem of object classification. A probabilistic model known as the Constellation Model is used that explicitly handles missing features and background clutter, in addition to representing the spatial layout of the parts in the model. A particular attraction is that they show a system which requires minimal supervision to train, even on cluttered images.

The model of Fischler and Elschlager was revisited by Burl *et al.* [21, 20, 66, 67]. Natural images of faces were the input to the system. A face model was manually trained by identifying fiducial points on a set of training faces, giving statistics for a set of detectors as well as joint statistics of their relative location. Detection methods handled spatial deformations in principled manner are were tolerant of background clutter. See Figure 2.9 for detection examples.

(a) (b)

Figure 2.9: From Leung, T., Burl, M. and Perona, P. [21]. (a) The joint shape model allows the locations of other features to be predicted (illustrated by ellipses) once the location of the eyes are fixed. (b) Left column: global best match in each image of whole model. Right column: Second best fit, with a likelihood around an order of magnitude worse than the matches in the left column.

Weber *et al.* [108, 109, 110, 111] then built on the approach of Burl *et al.* The major advance was making the training process unsupervised even with cluttered, occluded images, something that no other approach has previously demonstrated. This was done by automatically obtaining a set of potentially useful pixel patches by running an interest operator on the training set; chopping out patches around each interest point and then using k-means clustering on the

patches. Figure 2.11(a) shows an image with the interest points marked and the codebook of clusters formed from points of all training images, the cluster centres representing a set of commonly occurring patches in the training set. The cluster centres are used as detectors to provide a set of points from which the shape model is learnt. The detections from various combinations of cluster centres are used in turn to build a set of shape models. Each model was trained on a training set using EM and tested on a separate validation set. The final model selected was the one which gave the best performance on the validation set. Hence the final model has a generative location model but uses discriminative features. Figure 2.10 shows the overall scheme. This powerful algorithm was applied to a variety of object category detection tasks (e.g. cars, faces, cartoon characters, leaves). Figure 2.11(b) shows the four clusters chosen and the shape model learnt, superimposed on an example image. In [109] several models are used to perform human head detection from a variety of view points, giving 3-D object detection. In [110] mixture models are introduced and a route to automatic category discovery is proposed.

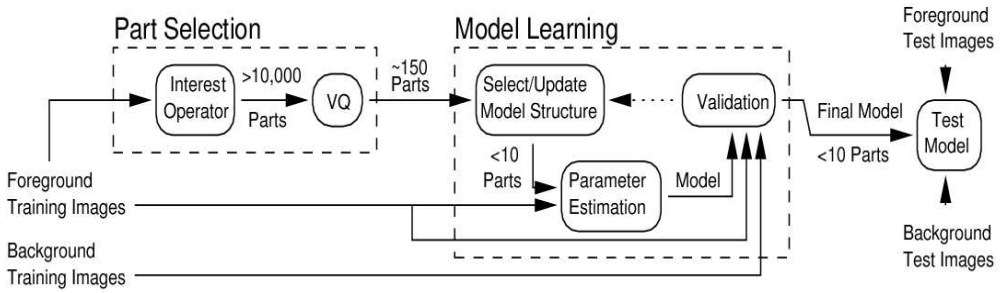
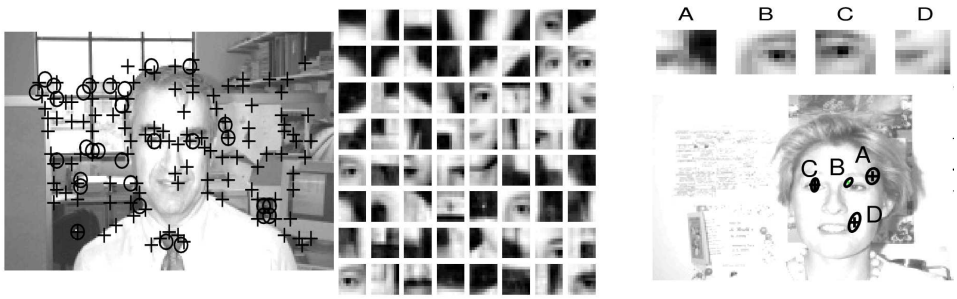


Figure 2.10: An overview of Weber *et al.* [109].



(a) (b)

Figure 2.11: From Weber *et al.* [109]. (a) Left — A training image with interest points superimposed. Right — Codebook of cluster centres obtained by clustering points from all training images. (b) The final shape model superimposed on a test image. The corresponding cluster centres, chosen automatically, are shown above.

This thesis builds directly on Weber *et al.* While their model is rigorous in many respects, it does not deal with the appearance of the model parts in such an elegant way. The clustering procedure can often end up with a large set of uninformative features, from which no useful model can be built. The shape and appearance models are constructed separately; the shape model being learnt from a set of points which contain no information about the quality of the appearance match. By contrast, the model presented in this thesis models appearance using probability density functions, giving a probability for each match. More importantly, the entire representation is now probabilistic, enabling the learning and recognition tasks to be posed as machine learning problems. As such, many techniques from that domain can easily be applied. We make a detailed comparison between the model used by Weber *et al.* and the one introduced in this thesis in Section 4.9.3. A quantitative comparison is also made between the two methods in Chapter 6.

Fei-Fei *et al.* [32] give an example of the application of powerful machine learning methods to the Constellation Model. They introduce a hierarchical Bayesian version of the Constellation Model which is able to incorporate priors into the learning procedure in a principled manner. This enables the algorithm to train from very few images (< 5) rather than the hundreds typically required.

Felzenszwalb and Huttenlocher [36] make an important contribution to the efficiency with which features in the image are matched to parts in the model. They use a parts and structure model where the dependencies in spatial relations between parts are tree-structured (as opposed to the joint model in Weber *et al.*). This model is used to find people in images (see Figure 2.12), the articulated structure of humans being represented as a tree. Although training is manual, in recognition they use efficient methods to find the globally best match for the model over all pixel locations in the image. For a model with P parts and N possible locations per part, a tree-structured model would normally have $O(N^2P)$ possible states that must be evaluated. Their contribution is to reduce this to $O(NP)$, which enables them to take every pixel as a possible location (hence N is $O(10^6)$) obviating the need for feature detectors in recognition.

Crandall *et al.* [25] investigate shape models of increasing complexity in a parts and structure type approach. They find that relatively simple models provide similar performance to complex fully connected spatial models, but have a substantially lower computational cost. They make

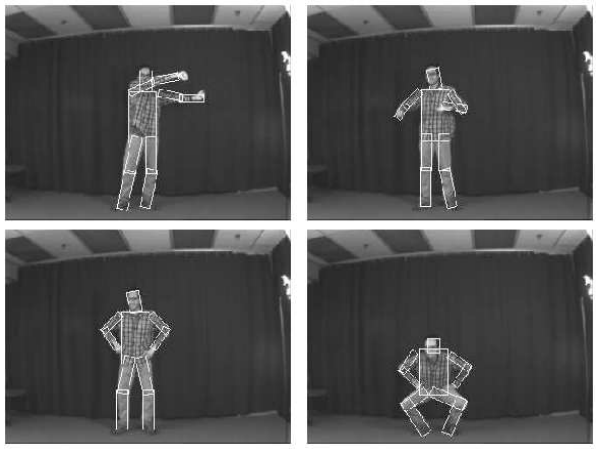


Figure 2.12: Detection examples from Felzenszwalb and Huttenlocher [36]. The articulated human model has tree-structured dependencies, for example the right lower arm is connected to the right upper arm, but not the torso or any other limbs. This reduced dependency structure allows fast matching in recognition.

use of the efficient methods of Felzenswalb and Huttenlocher [36] in recognition, demonstrating its effectiveness on classes other than humans.

Agarwal and Roth [1] tackle car classification. In the manner of Weber *et al.* , interest points are found and clustered to find common pixel patches. These patches are used as filters with which features are found in the image. Vectors are formed from combinations of features in the image, their spatial relations being encoded in a coarse manner. These vectors are then used in a sparse-network-of-winnows (SNOW) classifier [23]. The algorithm uses a window of interest which is moved exhaustively over a query image. Scale invariance is achieved by exhaustively trying different sizes of window. The algorithm differs from Weber *et al.* in that the spatial information is encoded in the vectors used by the classifier rather than forming an explicit shape model. They test on the UIUC dataset [2], achieving mediocre results.

Borenstein and Ullman [16] present a scheme which combines object classification with segmentation. The object is modelled by a small set of image fragments — small rectangular textured patches, distinctive of the class. These fragments are chosen automatically by comparing the frequency of occurrence in class and non-class images, picking those that are commonly found in the former but not the latter. In recognition, full coverage of the object is obtained by combining the fragments, in the manner of a jigsaw. Since each fragment is accompanied by foreground-background mask (the training images are manually segmented), a pixel-level segmentation of the test instance may be obtained.

Inspired by Agarwal and Roth [1], Leibe and Schiele [65] present a leading approach to categorization beating many methods on a wide range of datasets [30]. Although a greater degree of supervision is required in training since each training example must be manually segmented, it has two main advantages: (i) it allows a validation step which measures the support, on a pixel level, of the fit between the model and the image, so reducing false alarms and (ii) it permits a pixel level segmentation of object from test instances in the manner of Borenstein and Ullman [16]. The validation step is reminiscent of the approach used by geometric methods such as Lowe [73]. The scheme first finds a set of regions for each training image, then clusters them in the manner of Weber *et al.* and Agarwal and Roth [1, 109], additionally recording for each cluster (i) the relative location of the object centre and (ii) the average foreground/background mask. In recognition interest points are again found and then a probabilistic Hough scheme used to vote for the position of the object within the image, based on the match of the regions to each of the clusters. The maximum in voting space is found and used to project back into the image the regions which belong to the object. Then the foreground/background masks of each cluster can be used to provide a segmentation of the object. The entire process is illustrated in Figure 2.13.

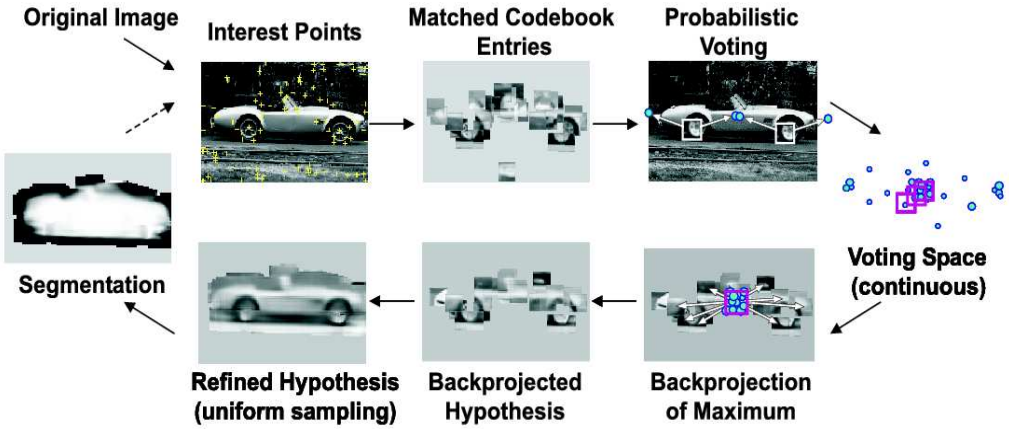
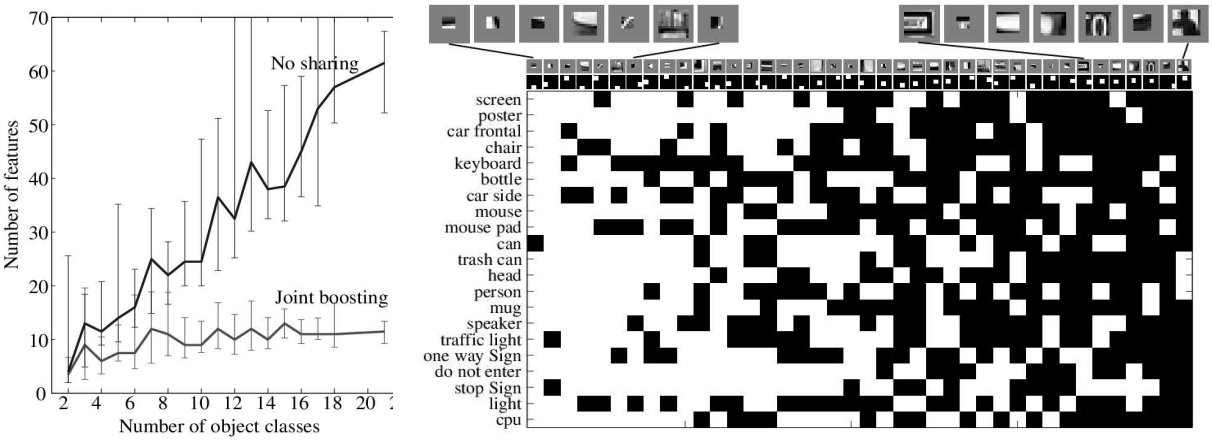


Figure 2.13: An overview of the combined recognition and segmentation scheme of Leibe and Schiele [65].

Torralba, Murphy and Freeman [104] provide a method for scaling object recognition up to many hundreds of categories. A desired property of a recognition system is that the time per test image should scale sub-linearly with the number of category models learnt. Otherwise, any system will become unmanageably slow once hundreds of categories are reached. They use

a variant of boosting to learn a classifier that maximally separates all classes simultaneously, while sharing it across several classes. Each classifier is a small pixel patch, along with a spatial mask, to offset it relative to the centre of the sub-window that is exhaustively slid over the image. The number of classifiers used by their elegant scheme is roughly logarithmic with the number of categories (see Figure 2.14(a)). Around 10 classifiers are used with 20 classes, as supposed to 60 or so if each class were learnt separately. Figure 2.14(b) shows a table listing which classifiers are used by which class. Multi-view experiments are also performed, a shared set of classifiers being learnt for all viewpoints, rather than learning a separate set for each view. By sharing the features they show that, not only is recognition faster, but the performance is improved.



(a) (b)

Figure 2.14: Plots from Torralba, Murphy and Freeman [104]. (a) The lower curve shows how the number of features used scales sub-linearly with the number of categories, while the top curve shows how the scaling is linear if no feature sharing is used. (b) A table indicating which features are used by which category. White indicates it is used, black that it is not. Note that the low-level features (on the left) are used by all categories, while highly specific ones are only used by a few.

Berg and Malik propose a scheme based on deformable shape matching where the correspondence between the model and features in the image is posed as an integer quadratic programming problem. While such problems are typically NP-complete, they use some approximations which enables correspondences to be made in reasonable time with 50 model points and 50 possible matches for each. Their formulation is able to deal with outliers when estimating the correspondence, thus occlusion and background clutter can be handled. Once correspondence has been estimated, a thin-plate spline is used for give an aligning transform between the model and the

test exemplar, giving a dense correspondence between the two. They test on the Caltech 101 dataset obtaining a correct detection rate of roughly three times that of Fei-Fei *et al.* [33].

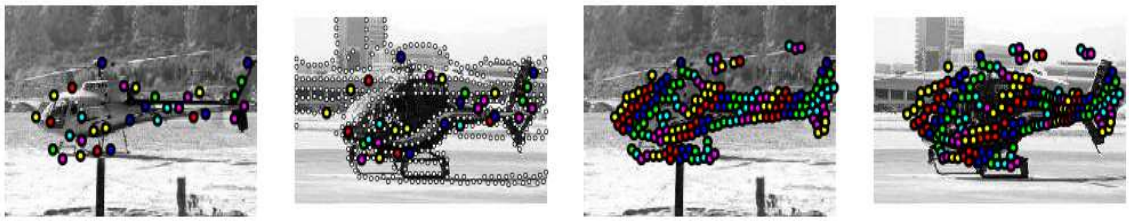


Figure 2.15: From Berg *et al.* [10]. Far Left: a training image with a subset of feature points marked. Left centre: Test image with feature points marked in white and points in correspondence with the training exemplar shown in colour. Right centre: Training image with all feature points marked in colour, according to location in the image. Far Right: Test image with feature points of the training image mapped by a thin-plate spline based on the correspondences established by matches in left centre image.

In [64], LeCun applies convolutional neural networks to classes other than digits and faces. In a series of thorough experiments, a set of toy objects (5 classes, 5 examples per class) with their texture obscured are photographed on a turntable from a wide range of views and lighting conditions to give a large set of images of each class (∼5000). These images are then used to train a convolutional neural network. Since the texture of the object is removed (by painting the object matte green), the CNN uses edge features to distinguish between the objects. The system gives real-time performance in recognition over a wide-range of viewpoints but is adversely affected, to some degree, by cluttered backgrounds since the spurious edges generate false alarms. Since only 5 object instances are used for each category in training and 5 in testing, it is not clear how well the system generalizes to further instances from each class. The small number of instances is a reflection of the effort required to compiling the training set. Thus, although CNNs are undoubtedly powerful systems, it would seem that the sheer amount of training data needed (and effort in obtaining it) is something of a stumbling block.

Barnard *et al.* [8] present a very different paper, notable for its learning methods, where models are learnt from not just from images but also accompanying text labels. Each image is over-segmented using normalized cuts to give a larger of regions. The regions are represented by vectors encoding low-level concepts such as colour and area. The vectors from each image are modelled jointly with the text labels, establishing a correspondence between the two. Hence in a recognition scenario, given one the other can be predicted. They evaluate their model on a large database from an art museum.

Summary of recent work

Although the appearance-only methods perform very well, the interesting intellectual challenge is how to use location information since it is clear that it must part of the solution to the recognition problem. When incorporating a shape model, the problem that must be solved is that of the *correspondence* between the model and the image. Many of the papers above use ingenious schemes to tackle this problem (e.g. [10, 36, 65]). It is notable that very few methods can provide viewpoint invariance other than simple translations or scalings. Solving the 3-D correspondence problem is difficult, consequently of the paper of Torralba, Murphy and Freeman is the only one to show any convincing multi-view experiments as they choose to search exhaustively over the image instead of worrying about correspondences.

Many of the methods require minimal supervision in training - important if thousands of classes are to be learnt. However it is notable one of the best performing methods, that of Leibe and Schiele [65], does require manual segmentation of training images; although only 50 are needed compared to the hundred for other methods.

2.2.4 Summary of literature

It is clear that there are many different ways to tackle recognition. When looking at all the methods it is important to keep mind the vast increase in computing power that has taken place since the first attempts were made. Seen in this context, is natural that much of the older work is bottom-up in nature, since this is less computationally demanding that than a top-down approach. For example, earlier approaches tried to extract rich geometric descriptions from edge information by grouping it until curve segments or combinations of lines were found.

More recently, top-down approaches have gained favour as machines have become fast enough to make them practical. For category-level recognition, a model of some kind would seem an essential part of any scheme and in the top-down paradigm, this model needs to be projected down into the image. The practical implications of this is the aforementioned correspondence problem.

An important recent trend is the application of many standard machine learning techniques such as EM, Neural Networks, SVM's or Boosting to recognition. The subtlety in recent papers is the application of these powerful methods to the recognition problem in a tractable manner.

As we are unable to manually provide a reliable criterion for distinguishing object classes, we have no option but to learn it from the data, hence machine learning methods would seem to be a natural match for the recognition problem.

2.3 Features and Representation Schemes

In this thesis we choose to represent each image as a set of features, thus it is vital that they still retain the distinctive information about the image otherwise the model has nothing to work with. We now review a variety of feature detectors that we will use with the model. Five complementary types of detector were chosen: Kadir & Brady, Curve Segments, Difference of Gaussians, Multiscale Harris and the Sampled Edge detector. We now describe each one in turn. Note that we do not use colour information in our model, thus all detectors operate on greyscale images only.

2.3.1 Kadir & Brady

The detector of Kadir and Brady [56] is well matched to our modelling approached due to its stable output when only a small number of regions per image are required. This method finds regions that are salient over both location and scale. For each point in the image a histogram $P(I)$ is made of the intensities in a circular region of radius (scale) s. The entropy $H(s)$ of this histogram is then calculated and the local maxima of $H(s)$ are candidate scales for the region. The saliency of each of these candidates is measured by $H\frac{dP}{ds}$ (with appropriate normalization for scale [56, 69]).

This gives a 3-D saliency map (over x,y and s). Regions of high saliency are clustered over both location and scale, with a bias toward clusters of large scale, since they tend to be more stable between object instances. The centroids of the clusters then provide the features for learning and recognition, their coordinates within the saliency map defining the centre and radius of each feature.

A good example illustrating the saliency principle is that of a bright circle on a dark background. If the scale is too small then only the white circle is seen, and there is no extrema in entropy. There is an entropy extrema when the scale is slightly larger than the radius of the bright circle, and thereafter the entropy decreases as the scale increases.

In practice this method gives stable identification of features over a variety of sizes and copes well with intra-class variability. Empirical observations show that the detector prefers round, blob-like structures as opposed to the corner structures which many detectors prefer. The saliency measure is designed to be invariant to scaling, although experimental tests show that this is not entirely the case due to aliasing and other effects. Note, only monochrome information is used to detect and represent features. An implementation of this feature detector is available on the Internet [57].

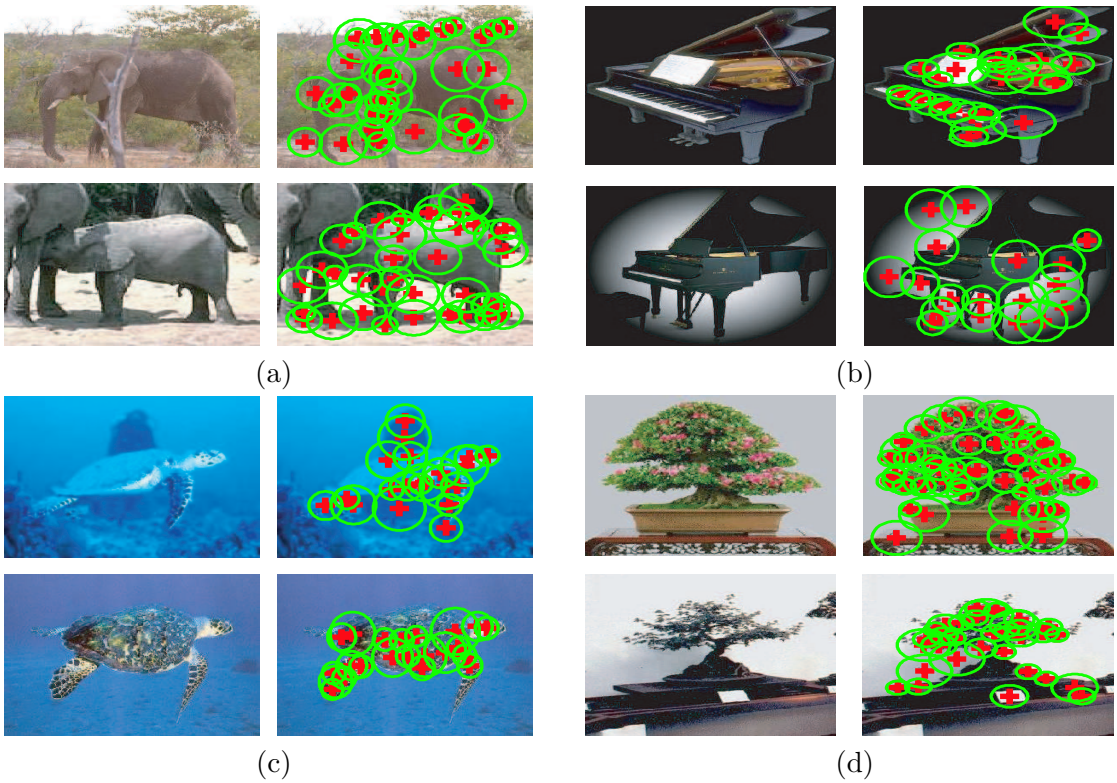


Figure 2.16: Images from four classes with and without the output of Kadir & Brady feature detector superimposed: (a) Elephant, (b) Grand piano, (c) Hawksbill, (d) Bonsai tree.

2.3.2 Curves

The previous feature type is region-based, representing solely the appearance of the object. For certain types of object, such as bottles, the distinctive information about the class is embodied in its profile rather than its appearance. To capture such information, we introduce an alternative feature type designed to capture the outline of the object with curve segments.

Rather than only consider very local spatial arrangements of edge points (as in [4]), extended

edge chains are used, detected by the Canny edge operator [22]. The chains are then segmented into segments between bitangent points, i.e. points at which a line has two points of tangency with the curve. Figure 2.17(b) shows an example.

This decomposition is used for two reasons: first, bitangency is covariant with projective transformations. This means that for near planar curves the segmentation is invariant to viewpoint, an important requirement if the same, or similar, objects are imaged at different scales and orientations. Second, by segmenting curves using a bi-local property interesting segments can be found consistently despite imperfect edgel data.

Bitangent points are found on each chain using the method described in [94]. Note that end points of the curve are also considered bitangent points. Since each pair of bitangent points defines a curve which is a sub-section of the chain, there may be multiple decompositions of the chain into curved sections as shown in Figure 2.17(b). In practice, many curve segments are straight lines (within a threshold for noise) and these are discarded as they are intrinsically less informative than curves. In addition, the entire chain is also used, so retaining convex curve portions.

2.3.3 Difference of Gaussians

Another type of scale-invariant interest operator was introduced by Lowe [70, 72]. It is designed to be efficient to compute so that the regions may be extracted in real-time. The regions found are extrema in the convolution of a difference-of-Gaussian function with the image. This can be computed by repeatedly blurring the image with Gaussian kernels and then taking the difference between images of adjacent blurrings. For efficiency, the scale space is divided into a series of octaves, each one being a factor of 2 smaller than the other. Figure 2.18 illustrates the procedure.

2.3.4 Multiscale Harris

This detector, proposed by Mikolajczyk and Schmid [77], is designed to find corners within images, adjusting the scale of the region to find an optimal size. The first stage of the detector identifies interest points using the Harris corner detector [49]. The second stage finds the characteristic scale of the interest point, thus specifying a region. This is done by varying the

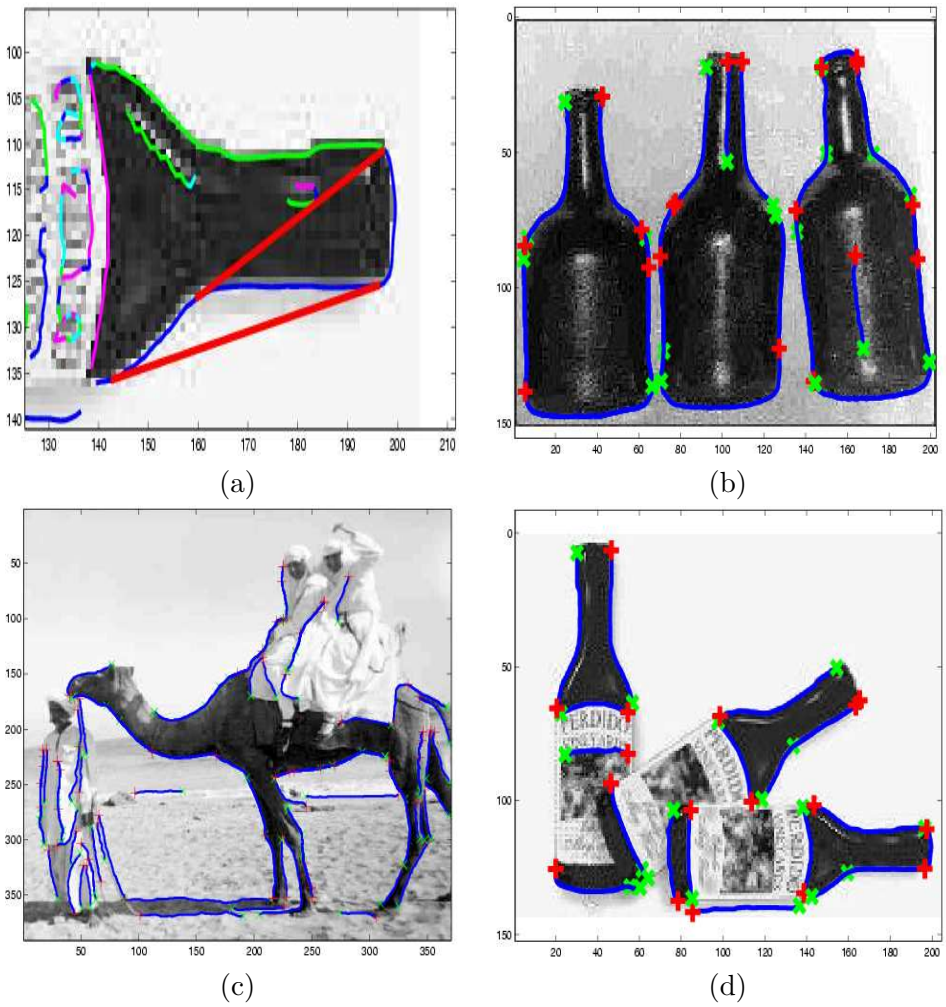
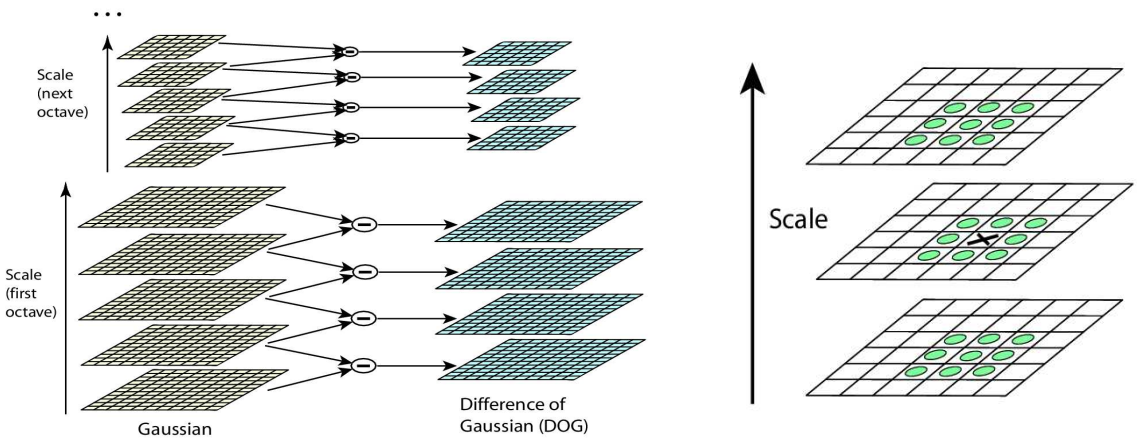


Figure 2.17: (a) A long curve segment being decomposed at its bitangent points. (b), (c) & (d) show the curve segments identified in three images. The green and red markers indicate the start and end of the curve, respectively.

radius of a scale-normalized Laplacian filter and measuring its response when applied to the image at the interest point. The response will have a peak at a certain radius, which is taken to be the characteristic scale.

2.3.5 Sampled Edge operator

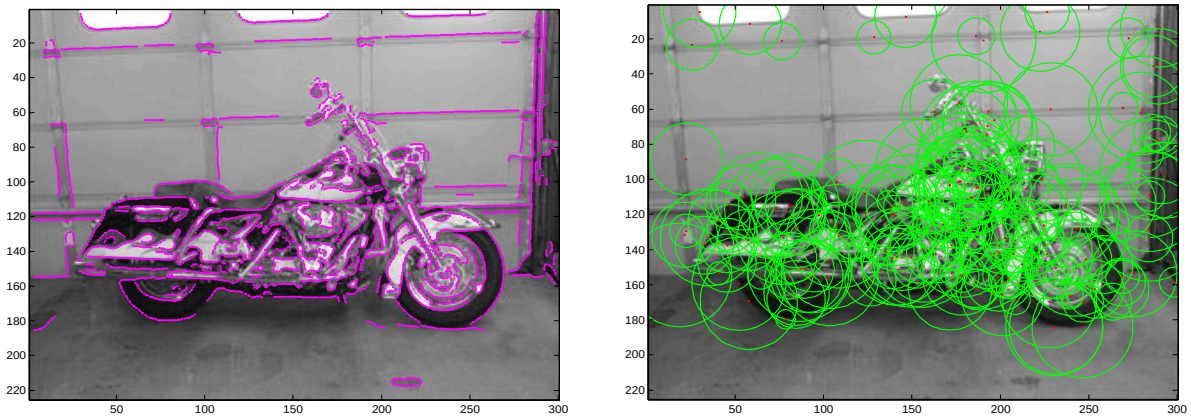
While the curve detector in Section 2.3.2 utilizes edge information from the image, its output is relatively sparse since obtaining long edgel chains is difficult. We therefore propose a simple region-based operator which directly uses edge information. Inspired by the approach of Berg *et al.* [10], we first find edgels in the image and then locate a region at points drawn at random from the edgel set. The scale of the region is chosen by drawing from a uniform distribution



(a) (b)

Figure 2.18: Reproduced from Lowe [72]. (a) For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce a set of images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2 and the process repeated. (b) Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbours in 3x3 regions at the current and adjacent scales (marked with circles).

over a sensible scale range (a radius range of 5-30 pixels). The total number of regions sampled is capped to give a number similar to the other three types of detector. Figure 2.19(a) shows an image with edgels superimposed. The regions samples from these edgels are shown in Figure 2.19(b).



(a) (b)

Figure 2.19: The edge-based interest operator. (a) Sample image with edges found using the Canny edge detector. (b) Regions found by sampling from the edge points. See text for details of its operation.

2.3.6 Comparison of feature detectors

The set of feature detectors are chosen to be complementary in nature, the idea being that their combined output will give a rich, complete coverage of any object while still drastically reducing the quantity of data each image presents to the classifier. The Kadir & Brady prefers round circular blob structures. The Multiscale-Harris, by contrast, fires predominantly on corners, with the difference of Gaussians being somewhere in between both in its behaviour. These three region-based detectors are designed to predominantly capture the textured interior of the object. The remaining two methods capture information about the outline of the object, using edge information alone. The curve detector extracts long coherent edge sections while the sampled edge approach fires randomly over strong edges in the image. Figure 2.20 shows two images containing airplanes with the output of Kadir & Brady, Multiscale Harris and Curves superimposed. The figure illustrates a beneficial property of the Kadir & Brady detector which is that it fires relatively infrequently in the background of images, which often tend to be out-of-focus and uniform, compared to other detectors like Multiscale Harris.

2.3.7 SIFT descriptor

A SIFT (Scale Invariant Feature Transform) [70] takes each region, finds its gradients and then normalizes for orientation by finding the dominant orientation rotating the region so as to make it axis aligned. Then 8-bin orientation histograms are formed of the gradients in each cell of a 4 by 4 spatial grid overlaid on the region. See Figure 2.21 for an illustration of this process. Each region is now described by a $4 \times 4 \times 8 = 128$ dimensional vector. The idea is that the loose grid gives a little bit of slop to accommodate minor translation and scale offsets due to inexact feature detection while the gradient based representation makes it less sensitive to illumination changes.

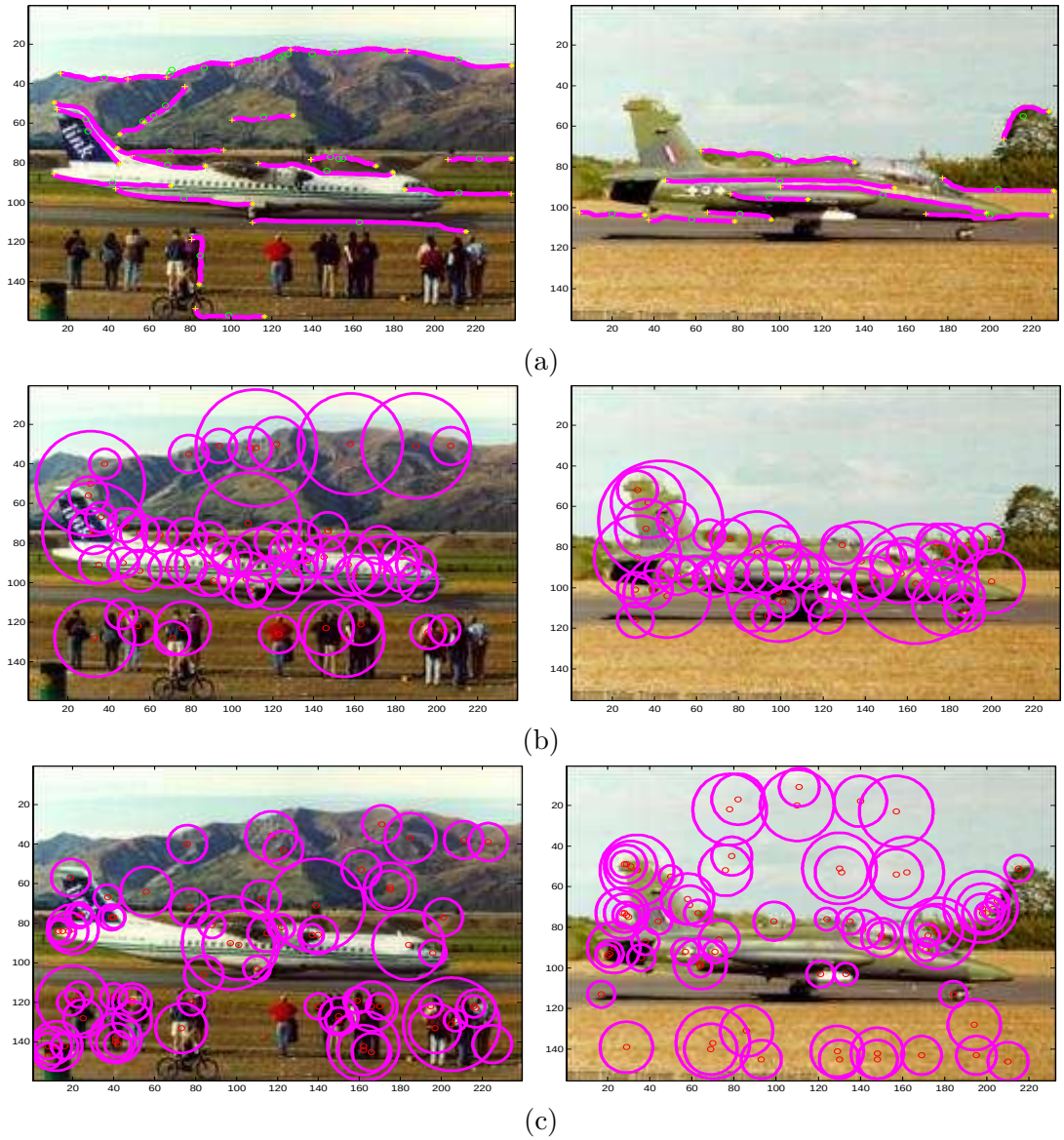


Figure 2.20: Output of three different feature detectors on two airplane images. (a) Curves. (b) Kadir & Brady. (c) Multi-scale Harris.

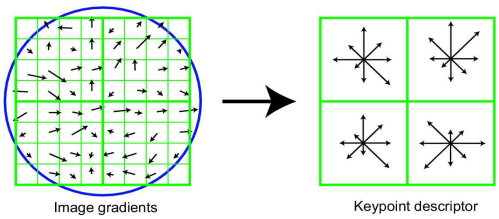


Figure 2.21: The SIFT descriptor of Lowe [72]. On the left are the gradients of an image patch. The blue circle indicates the Gaussian centre-weighting. These gradients are then accumulated over 4×4 subregions, as shown on the right, the length of the arrow corresponding to the sum of the gradient magnitudes in that direction. A 2×2 descriptor array is shown here, but we use a 4×4 array in our experiments.

Chapter 3

Datasets

To provide a thorough evaluation of the two proposed approaches, the Constellation model and TSI-pLSA, we use a wide range of different datasets to train and test the model. This chapter gives details about each dataset and discuss how they differ from one another. We review each dataset, commenting on its relative difficulty is the following areas:

- Intra-class variability - How much the appearance (and outline) of the different object instances vary from frame to frame.
- Occlusion - The degree to which parts of the objects are occluded in different instances. If the visible portion of the object is small, clearly the object becomes hard to detect.
- Viewpoint variation - Are the different instances of same pose and aspect?
- Background clutter - What portion of the image does the object typically occupy? If this is large then recognition is likely to be easier than if it is small.
- Quantity of training data - How much data is available to train from. Training complex models such as those proposed in this thesis is more difficult.
- Multiple instances - Finding many (possibly overlapping) instances is harder than finding a single occurrence.
- Polluted training data - The portion of the training images that actually contain an instance of the stated category.

Each of the dataset focuses on a different subset of these challenges, so that the strengths and weaknesses of our approaches may be elucidated. We evaluate each dataset on the above criteria and summarise in table Table 3.1.

3.1 Caltech datasets

The Caltech datasets [37] consist of 10 object categories: frontal faces; cars from the rear; airplanes from the side; motorbikes from the side; leopards; bottles; camel; guitar; houses and wrist watches. For each category there are between 200 and 900 images containing instances of a similar pose. The leopard dataset, obtained from the Corel database, is only 100 images originally, so another 100 were added by reflecting the original images, making 200 in total. These datasets are intended to be a fairly straightforward starting point for evaluation. Although the intra-class variability is large, the pose variability is small and there is only moderate background clutter. All the instances occupy most of the frame, making the datasets more useful in evaluating classification performance rather than localization. However, 5 of the 10 (airplanes,motorbikes,faces,cars rear and leopards) are also evaluated in a localization setting.

Three different background datasets are used as negative examples when testing in a classification role:

1. Scenes around Caltech consisting of indoor offices; outdoor buildings and plant foliage.
2. A set of empty road scenes on US roads for use with the Cars Rear category.
3. A set of images collected from Google's Image Search (`http://www.google.com/imghp`) using the keyword "things". All unique images above 100 pixels in width were downloaded. The set is designed to form an alternative background set to the Caltech scenes.

Figure 3.1 shows examples from 5 of the Caltech classes and two of the background datasets.

3.2 UIUC dataset

The UIUC dataset [2] consists of a single class - cars from the side. The viewpoint is tightly constrained, the cars being exactly side on, but facing either to the left or right. A tightly cropped, scale-normalized, set of 500 training examples are given, along with negative training

examples. The test set consists of 120 larger images containing multiple car instances (1–3) at the same scale as the training examples. The dataset therefore has no scale (or orientation) variability. Additionally, intra-class variability is small since all the instances are US-style cars, with no vans or trucks present. The dataset is designed to test localization ability only, no set of background test images being provided (as required for classification). Figure 3.1 shows examples from the training set.

3.3 Fawlty Towers

A more challenging dataset designed to test localization performance is that of 1970's television footage in the form of the situation comedy "Fawlty Towers". Three episodes were used: 1. "A touch of class", 2. "The Wedding party" and 3. "The Builders". For each episode, every 30th frame was taken giving a total of 1463, 1934 and 1620 frames for each episode respectively. The image is 720 by 576 pixels, having been deinterlaced from the original television frames. The images are of limited quality, coming from 30 year old footage with some sections contain significant motion blur.

Ground truth was established for two classes of object: barometers and cars. Barometers appear in all three episodes in the same location on the set but the actual instances change between episodes, each one looking different. Cars appear only in "A touch of class" but there are a variety of instances appearing in different locations within the episode. Both objects occur relatively infrequently, making them challenging to find (statistics are given in Table 3.1). Examples of the objects are shown in Figure 3.2.

3.3.1 Training data for Fawlty Towers

Note that Fawlty Towers episodes are only used in testing. The object models for cars and barometers are trained from two small sets of images manually gathered from Google's Image Search using the keywords "barometer" and "car front". 15 images were gathered for each class, examples of which are shown in Figure 3.4.

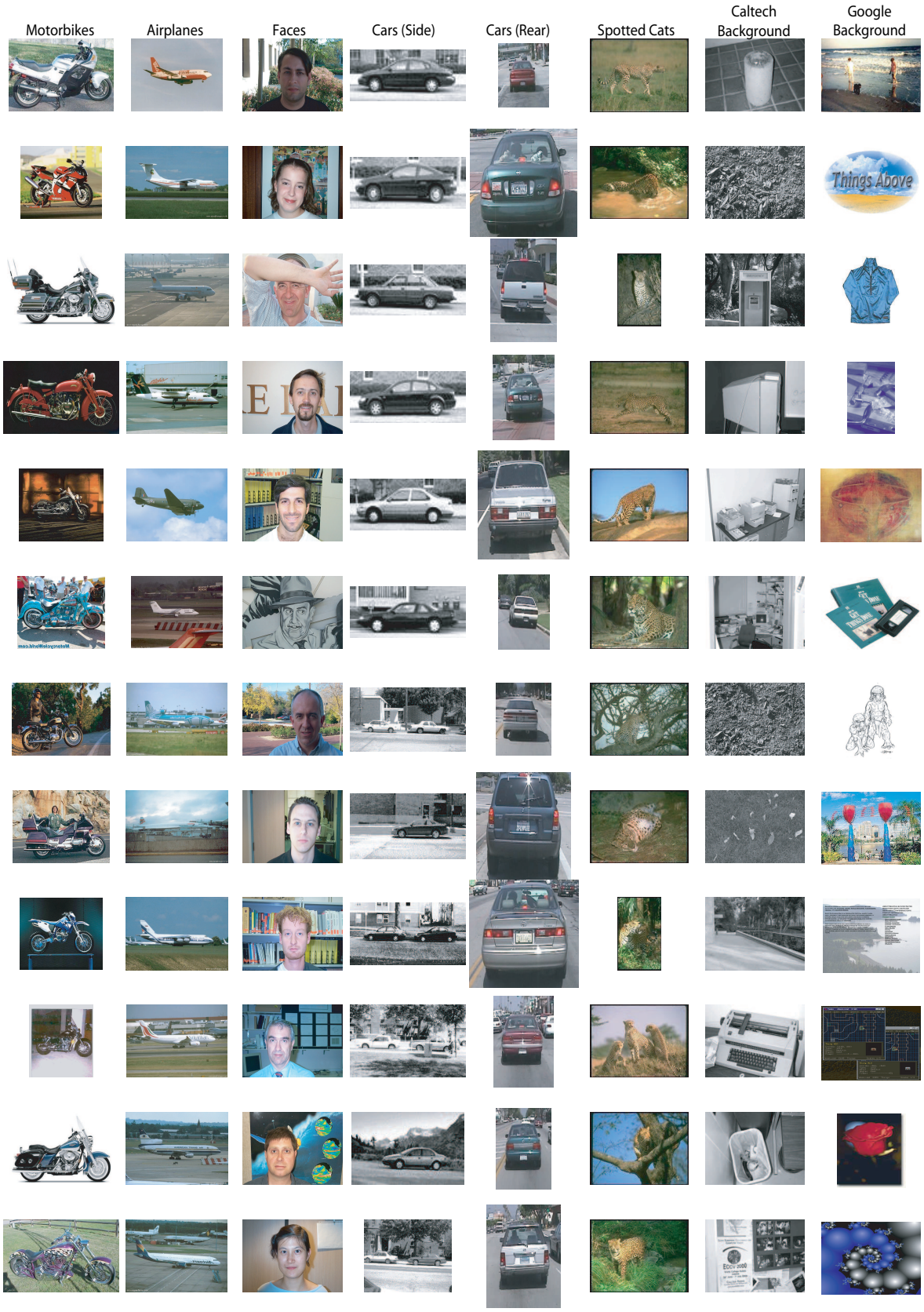


Figure 3.1: Some sample images from 5 of the Caltech datasets (Motorbikes, Airplanes, Faces, Cars (Rear) and Leopards) and the UIUC dataset (Cars (Side)). The two columns on the far right show images from the Caltech background and Google "things" datasets.

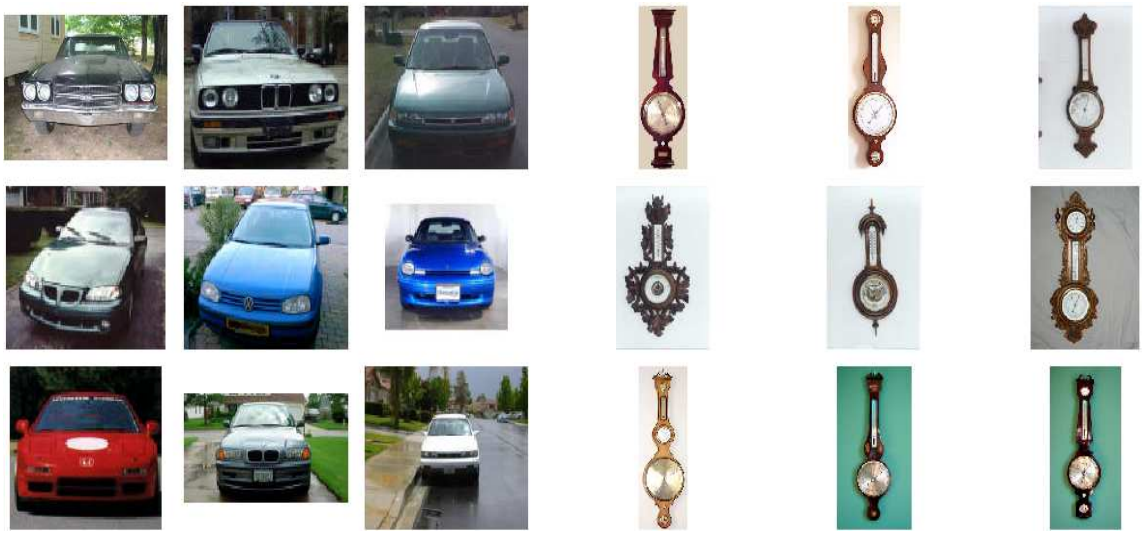
(a) (b) (c)

Figure 3.2: Barometers from Fawlty Towers episodes: (a) A touch of class; (b) The wedding party and (c) The builders. Note that the barometer changes between episodes.

Figure 3.3: Example images from the Fawlty Towers episode "A touch of class". The first row shows general scenes with no car or barometer present; the second row images contain barometers and the third row shows images with cars in.

3.4 PASCAL challenge

The PASCAL challenge [30] is an object recognition competition for which a number of high quality datasets have been prepared. A wide variety of different algorithms have been tested on



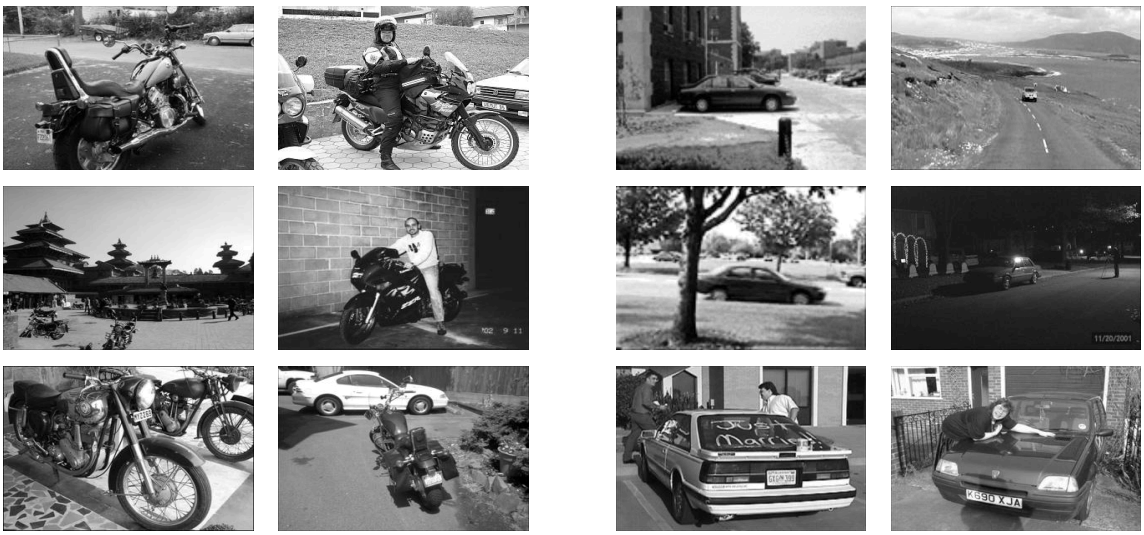
(a) (b)

Figure 3.4: (a) Sample training images for Cars (Front). (b) Sample training images for Antique Barometers.

these datasets in both classification and localization. Four categories were used: Motorbikes, Bicycles, Cars and People. For each dataset, two different test sets were provided, along with a fixed training and validation set. The first test set is moderately easy, having roughly the same level of pose variation as the Caltech datasets. The second test set is far more challenging, containing many instances that are very small within the image, as small as 1% of the image area (see Figure 3.5). We use two of the four categories: Cars and Motorbikes, using both test sets in a classification and localization evaluation.

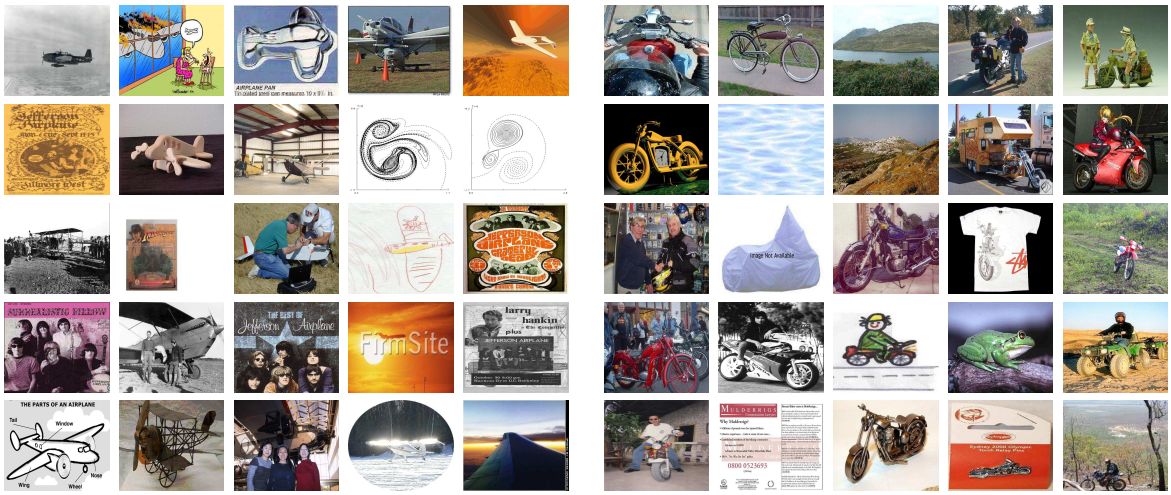
3.5 Image search engine data

Internet search engines provide a free source of images: many popular search engines have Image searches which return images instead of URLs. However the visual content of the images is not used in the search, merely the text in the filename or surrounding HTML. Consequently, as many as 85% of the returned images may be visually unrelated to the intended category, perhaps arising from polysemes (e.g. "iris" can be iris-flower, iris-eye, Iris-Murdoch). Even the 15% subset which do correspond to the category are substantially more demanding than images in typical training sets [37] – the number of objects in each image is unknown and variable, and the pose (visual aspect) and scale are uncontrolled. For example, in Figure 3.6 we see that only 4 or 5 of the 25 images have clear examples of airplanes.



(a) (b)

Figure 3.5: Images from the more challenging PASCAL test (#2) set for (a) motorbike and (b) cars. Note the wide variation in viewpoint and scale.



(a) (b)

Figure 3.6: A representative samples of the images returned from Google's image search using the keyword (a) "airplane" and (b) "motorbike". Note the large proportion of visually unrelated images and the wide pose variation.

For 7 categories, a set of images was automatically downloaded from Google's Image Search using the keywords: Airplane, Cars Rear, Face, Guitar, Motorbike, Leopard and Wrist Watch. Although in this thesis Google's image search was used exclusively, any other image search engine may be used provided that the images can be gathered in an automated manner, using the category name. Duplicates images and very small images ($<$ 100 pixels in width) were

discarded and Google's SafeSearch filter was left on, to reduce the proportion of unrelated images returned. For assessment purposes, the images returned by Google were divided into 3 distinct groups:

1. **Good images:** these are good examples of the keyword category, lacking major occlusion, although there may be a variety of viewpoints, scalings and orientations.

2. **Intermediate images:** these are in some way related to the keyword category, but are of lower quality than the good images. They may have extensive occlusion; substantial image noise; be a caricature or cartoon of the category; or the object is rather insignificant in the image, or some other fault.

3. **Junk images:** these are totally unrelated to the keyword category.

The labeling was performed by an individual who was not connected with the experiments in anyway, possessing no knowledge of our algorithms. Figure 3.7 shows the relative portions of labels for each of the 7 categories. We use the datasets obtained for each keyword to test the

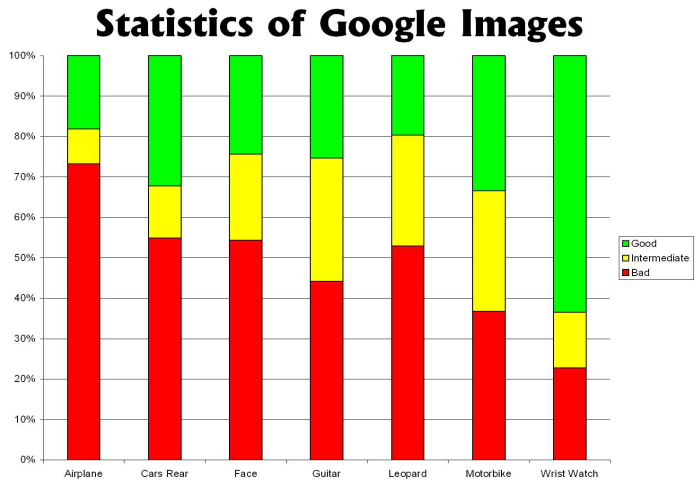


Figure 3.7: Bar chart showing the labels of images downloaded from Google's Image Search for 7 categories. Green - good examples; Yellow - intermediate and Red - junk. See text for a definition of these terms. Note the large portion of junk images for all categories, except wrist watch.

ability of our models to learn from polluted datasets.

3.6 Summary of datasets

The 5 different collections of datasets offer different challenges. The Caltech and UIUC are easier datasets, for classification and localization respectively. The Fawlty Towers datasets are a more challenging localization task, while the PASCAL offers a hard test for both classification and localization. Finally, the Google images test the ability of our schemes operate with polluted training data.

Table 3.1 lists the statistics of all the datasets used in this thesis. The column headings are as follows: Category - name of class or background set; Viewpoint - pose of object, if constant over dataset ((A) indicates alternating left and right facing instances); Collection - where the dataset came from; Evaluation - how the dataset is used (in classification (C) or localization (L) or both); Training pollution - Portion of *training* set which does not contain instances of the category; # Train - number of training images; # Validation - number of validation images (if any); # Test - number of testing images; Mean test instances/image - average number of object instances in *test* set (thus >1 indicates that some images have more than one instance while <1 indicates that some frames lack instances); Mean object area - average fraction of image area occupied by each image instance (to give some idea of how difficult the objects are to localize); Std. object area - standard deviation in the fraction of image area occupied by the object (indicates level of scale variability). Object area statistics are only given for datasets used in localization experiments. Note that some datasets are used exclusively for testing (such as the Fawlty Towers episodes), while others are just used for training (Google images). Bg. indicates a background dataset.

Category	Viewpoint	Collection	Evaluation	Training pollution %	# Train	# Validation	# Test	Mean test instances/image	Mean object area %	Std. object area
Airplane	Side	Caltech	C / L	0	217	0	218	1	33.3	11.4
Car	Rear	Caltech	C / L	0	400	100	400	1	42.1	15.0
Face	Front	Caltech	C / L	0	400	100	400	1	31.7	6.0
Motorbike	Side	Caltech	C / L	0	400	26	400	1	85.8	20.3
Leopard	Varying	Caltech	C / L	0	100	0	100	1	34.6	15.1
Bottles	Varying	Caltech	C	0	124	0	125	1	-	-
Camel	Varying	Caltech	C	0	178	0	178	1	-	-
Guitar	Varying	Caltech	C	0	400	100	400	1	-	-
House	Front	Caltech	C	0	400	100	400	1	-	-
Watch	Front	Caltech	C	0	180	0	181	1	-	-
Caltech Bg.	Varying	Caltech	C	0	400	0	400	-	-	-
Road Bg.	Varying	Caltech	C	0	400	0	400	-	-	-
Google Bg.	Varying	Caltech	C	0	400	0	400	-	-	-
Cars	Side (A)	UIUC	L	0	500	50	170	1.2	20.3	8.0
Barometer	Front	Fawlty	C	0	15	-	-	-	-	-
Car	Front	Fawlty	C	0	15	-	-	-	-	-
Barometer	Front	Fawlty 1	L	0	-	-	1463	0.129	2.4	1.6
Car	Front	Fawlty 1	L	0	-	-	1463	0.015	41.7	9.3
Barometer	Front	Fawlty 2	L	0	-	-	1934	0.091	5.8	4.4
Barometer	Front	Fawlty 3	L	0	-	-	1620	0.194	4.9	4.0
Motorbike 1	Side	PASCAL	C / L	0	107	107	216	1.02	60.2	20.7
Car 1	Side	PASCAL	C / L	0	136	136	275	1.24	17.5	12.3
Motorbike 2	Varying	PASCAL	C / L	0	107	107	202	1.12	47.6	24.0
Car 2	Varying	PASCAL	C / L	0	136	136	275	1.38	27.8	24.6
Airplane	Varying	Google	C	81.9	874	-	-	1	-	-
Car	Rear	Google	C	67.8	596	-	-	1	-	-
Face	Varying	Google	C	75.7	564	-	-	1	-	-
Guitar	Varying	Google	C	74.7	511	-	-	1	-	-
Leopard	Varying	Google	C	80.4	516	-	-	1	-	-
Motorbike	Varying	Google	C	66.6	688	-	-	1	-	-
Wrist watch	Front	Google	C	36.6	342	-	-	1	-	-

Table 3.1: A comparison of all datasets used in this thesis. See the text in Section 3.6 for a key to column headings.

Chapter 4

The Constellation model

4.1 Introduction

In this chapter, we introduce the constellation model, the first of our two approaches. This is an extension of the model proposed by Burl *et al.* [19, 20] and Weber *et al.* [108].

In this approach, we do not use a low-level representation (such as the pixels) of the image, instead we regard the image as a collection of *features*. These features are found by a set of detectors, which have been chosen to respond to different types of structures within images (e.g. interesting features of pixels; the outlines of objects and so on). Multiple types are needed since no one type of feature can represent all types of object: for human faces, patches of pixels are informative, but for wine bottles, the outline is more useful. The features used in our model are reviewed in Section 2.3.

The motivation for using this higher-level representation is two-fold: first, from a complexity point-of-view, it is easier to deal with $O(10^2)$ regions rather than $O(10^6)$ pixels. Second, the properties of the feature detectors mean that they fire preferentially on interesting areas of the image which are more likely to be part of the object of interest, de-emphasising uninformative areas which are typically in the background. Thus the foreground/background ratio is likely to be more favourable in a feature-based representation than in a pixel-based one.

Given a set of images containing an object, features will be found not only on the object but also in the background of each frame. Since our model is generative, our model must explain both sources of features. The *foreground* component of the model needs to represent the features on the object. These will be subject to intra-class variability and characteristics of the feature

detectors themselves. The *background* component needs to model the background of the image.

Before we give the details of the foreground and background models, we must first specify what information we obtain from the features within the image, as it is this that the model is based on. In the course of this chapter, we will also introduce the notation that will be used later. A summary is provided in Table A.2 in the appendix.

4.2 Model inputs

Each feature, regardless of type, provides the following information for use within the model:

- $\mathbf{x}$ - the location $(x, y)^T$ of the feature within the image, measured in pixels.
- s - the local scale of the feature within the image, taken as the radius of the feature (in pixels).
- $\mathbf{d}$ - the description of the feature as coordinates within some descriptor space. Vector of length a.

Although it is clear what $\mathbf{x}$ and s are, $\mathbf{d}$ requires a little more explanation, although it will be dealt within detail in section 4.4.1. $\mathbf{d}$ is a vector which describes a particular feature. If the feature is a curve segment, for example, then $\mathbf{d}$ would describe the shape of the curve; if the feature is a patch of pixels then $\mathbf{d}$ would describe the appearance of the patch.

Each image i contains N_t^i features of type t, with T types in total. In image i, the variables above from all features, of all types, are held in the large structures $\mathbf{X}^i, \mathbf{D}^i, \mathbf{S}^i$. $\mathbf{X}, \mathbf{D}, \mathbf{S}$ are then the structures holding all detections, of all types, from all I images.

4.3 Overview of model

An object model consists of a number of parts. Each part has an appearance, relative scale and can be occluded or not. Each part has a certain probability of being erroneously assigned to background clutter. Shape is represented by the mutual position of the parts. The entire model is generative and probabilistic, so appearance, scale, shape and occlusion are all modelled by probability density functions, which here are Gaussians. The model is scale and translation invariant in both learning and recognition. The process of learning an object category is one

of first detecting features and their scales, and then estimating the parameters of the above densities from these features, such that the model gives a maximum-likelihood description of the training data. Recognition is performed on a query image by again first detecting regions and their scales, and then evaluating the regions using the model parameters estimated in the learning. Note that parts refer to the model, while features refer to detections in the image.

The model is best explained by first considering recognition. Assume, we have learnt a generative object class model, with P parts and parameters θ_{fg}. We also assume that all non-object images can be modelled by a background with a single, fixed, set of parameters θ_{bg}. We are then presented with a new image i and we must decide if it contains an instance of our object class or not. In this query image we have identified N interesting features with locations $\mathbf{X}^i$, scales $\mathbf{S}^i$, and appearances $\mathbf{D}^i$. We now make a decision as to the presence/absence of the object by comparing the ratio of class posterior densities, R, to a threshold T:

$$R = \frac{p(\text{Object}|\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i)}{p(\text{No object}|\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i)} = \frac{p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\text{Object})\,p(\text{Object})}{p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\text{No object})\,p(\text{No object})} \tag{4.1}$$

$$\approx \frac{\int p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\,\theta)\,p(\theta|\text{Object})\,d\theta\,p(\text{Object})}{\int p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\theta')\,p(\theta'|\text{No object})\,d\theta'\,p(\text{No object})} \approx \frac{p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\,\theta_{fg})\,p(\text{Object})}{p(\mathbf{X}^i,\mathbf{S}^i,\mathbf{D}^i|\theta_{bg})\,p(\text{No object})} \tag{4.2}$$

The first approximation results from representing the class with its (imperfect) model, parameterized by θ. The ratio of the priors may be estimated from the training set or set by hand (usually to 1). The last expression is also an approximation since we will only use a single value for θ and θ' (the maximum-likelihood values, θ_{fg} and θ_{bg} respectively) rather than integrating over $p(\theta)$ as we strictly should.

Since our model only has P (typically 3-7) parts but there are N (typically 10 to 30) features in the image, we use an indexing variable $\mathbf{h}$ (as introduced in [20]) which we call a *hypothesis*. $\mathbf{h}$ is a vector of length P, where each entry is between 0 and N, which allocates a particular feature to a model part. The unallocated features are assumed to be part of the background, with 0 indicating the part is unavailable (e.g. because of occlusion). No features are permitted to belong to more than one part. The set H is all valid allocations of features to the parts; consequently $|H|$ is $O(N^P)$. Computing R in (4.2) requires the calculation of the ratio of the

two likelihood functions. In order to do this, the likelihoods are factored as follows:

$$p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i | \, \theta_{fg}) = \sum_{\mathbf{h} \in H} p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h} | \, \theta_{fg}) \tag{4.3}$$

$$= \sum_{\mathbf{h} \in H} \underbrace{p(\mathbf{D}^i | \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}, \theta_{fg})}_{Appearance} \underbrace{p(\mathbf{X}^i | \mathbf{S}^i, \mathbf{h}, \theta_{fg})}_{Shape} \underbrace{p(\mathbf{S}^i | \mathbf{h}, \theta_{fg})}_{Rel.\ Scale} \underbrace{p(\mathbf{h} | \theta_{fg})}_{Other} \tag{4.4}$$

We now look at each of the likelihood terms and derive their actual form. The likelihood terms model not only the properties of the features assigned to the models parts (the foreground) but also the statistics of features in the background of the image (those not picked out by the hypothesis). Therefore it will be helpful to define the following notation: $\mathbf{b} = sign(\mathbf{h})$ (which is a binary vector giving the state of occlusion for each part, i.e. $b_p = 1$ if part p is present and $b_p = 0$ if absent), $f_t = sum(\mathbf{b} == t)$ (the number of foreground features of type t under the current hypothesis) and $\mathbf{n} = \mathbf{N} - \mathbf{f}$ (a vector of the number of background features of each type).

If no object is present, then all features in the image belong to the background. Thus we only have one possible hypothesis: $\mathbf{h}_0 = \mathbf{0}$, the null hypothesis. The likelihood in this case is:

$$p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i | \, \theta_{bg}) = p(\mathbf{D}^i | \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}_0, \theta_{bg}) p(\mathbf{X}^i | \mathbf{S}^i, \mathbf{h}_0, \theta_{bg}) p(\mathbf{S}^i | \mathbf{h}_0, \theta_{bg}) p(\mathbf{h}_0 | \theta_{bg}) \tag{4.5}$$

As we will see below, $p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i | \, \theta_{bg})$ is a constant for a given image. This simplifies the computation of the likelihood ratio in (4.2), since $p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i | \, \theta_{bg})$ can be moved inside the summation over all hypotheses in (4.3), to cancel with the foreground terms.

4.4 Appearance

Here we describe the form of $p(\mathbf{D}^i | \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}, \theta)$ which is the appearance term of the object likelihood. We can simplify the expression to $p(\mathbf{D}^i | \mathbf{h}, \theta)$ if, given the detected features, we assume their appearance and location to be independent. Each feature's appearance is represented as a point in an appearance space, defined below. Each part p has a Gaussian density within this space, with mean and covariance parameters $\theta^{app}_{fg,p} = \{\mathbf{c}_p, V_p\}$ which is independent of other parts' densities. The background model has fixed parameters $\theta^{app}_{bg} = \{\mathbf{c}_{bg}, V_{bg}\}$, each part type t having a separate density with parameters $\mathbf{c}_{t,bg}, V_{t,bg}$. The appearance density is computed over

all features: each feature selected by the hypothesis is evaluated under the appropriate part density while all features not selected by the hypothesis are evaluated under the background density:

$$p(\mathbf{D}^i|\mathbf{h},\theta_{fg}) = \prod_{p=1}^{P} \mathrm{G}(\mathbf{D}^i(h_p)|\mathbf{c}_p, V_p)^{b_p} \prod_{j=1,j\backslash\mathbf{h}}^{N} \mathrm{G}(\mathbf{D}^i(j)|\mathbf{c}_{bg}, V_{bg}) \tag{4.6}$$

where G is the Gaussian distribution, and b_p is the pth entry of the vector $\mathbf{b}$, i.e. $b_p = \mathbf{b}(p)$. $\mathbf{D}^i(h_p)$ is the descriptor of the feature picked by h_p. If no object is present, then all features are evaluated under the background density:

$$p(\mathbf{D}^i|\mathbf{h}_0,\theta_{bg}) = \prod_{j=1}^{N} \mathrm{G}(\mathbf{D}^i(j)|\mathbf{c}_{bg}, V_{bg}) \tag{4.7}$$

As $p(\mathbf{D}^i|\mathbf{h}_0,\theta_{bg})$ is a constant and so is not dependent on $\mathbf{h}$, so we can cancel terms between (4.6) and (4.7) when computing the likelihood ratio in (4.2):

$$\frac{p(\mathbf{D}^i|\mathbf{h},\theta_{fg})}{p(\mathbf{D}^i|\mathbf{h},\theta_{bg})} = \prod_{p=1}^{P} \left(\frac{\mathrm{G}(\mathbf{D}^i(h_p)|\mathbf{c}_p, V_p)}{\mathrm{G}(\mathbf{D}^i(h_p)|\mathbf{c}_{bg}, V_{bg})} \right)^{b_p} \tag{4.8}$$

So the appearance of each feature in the hypothesis is evaluated under foreground and background densities and the ratio taken. If the part is occluded, the ratio is 1 ($b_p = 0$).

4.4.1 Appearance representation

The feature detector identifies regions of interest in each image. The coordinates of the centre give us $\mathbf{X}^i$ and the size of the region gives $\mathbf{S}^i$. Once the regions are identified, they are cropped from the image and rescaled to the size of a small $k \times k$ patch (typically $11 \leq k \leq 21$ pixels). Thus, each patch exists in a k^2 dimensional space. Since the appearance densities of the model must also exist in this space, we must somehow reduce the dimensionality of each patch whilst retaining its distinctiveness, since a 100+-dimensional Gaussian is unmanageable from a numerical point of view and also the number of parameters involved ($2k^2$ per model part) are too many to be estimated.

This is done by using principal component analysis (PCA). We use two variants:

1. Intensity based PCA. The $k \times k$ patches are normalized to have zero mean and unit variance. This is to remove the effects of lighting variation. They are then projected into

a fixed PCA basis in the intensity space of $k \times k$ patches, having d basis vectors. As used in Fergus *et al.* [40].

2. Gradient based PCA. Inspired by the performance of PCA-SIFT in region matching [58], we take the x and y gradients of the $k \times k$ patch. The derivatives are computed by symmetric finite difference (cropping to avoid edge effects). The magnitude of the gradients within the patch is then normalized to be 1, removing lighting variations. Note that we do not perform any orientation normalization as in [58]. The outcome is a vector of length $2k^2$, with the first k elements representing the x derivative, and the second k the y derivatives. The normalized gradient-patch is then projected into a fixed PCA basis of a dimensions. Two additional measurements are made for each gradient-patch: its unnormalized energy and the residual between the reconstructed gradient-patch using the PCA basis and the original gradient-patch. Each region is thus represented by a vector of length $a + 2$. The last two dimensions act as a crude interest measure of the region, while the remaining dimensions actually represent its appearance. [58]

Combining the vectors from all regions we obtain $\mathbf{D}^i$ for an image.

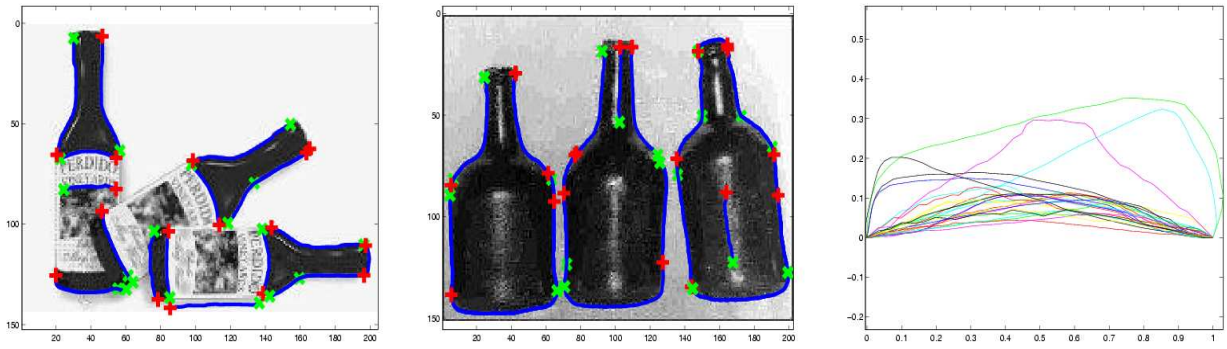
The fixed basis is computed using patches extracted using all Kadir and Brady regions [56] (described in Section 2.3.1) found on all the training images of Motorbikes; Faces; Airplanes; Cars (Rear); Leopards and Caltech background. Note that this basis is used for all object categories. We assume that the covariance terms between components will be zero, thus V_p (the covariance of a part's appearance) is diagonal in nature. Alternative representations such as ICA and Fisher's linear discriminant were also tried, but in experiments they were shown to be inferior.

4.4.2 Curve representation.

Two different representations are used for curves throughout this thesis. The first treats the curve as a region of interest, the centroid of the curve corresponding to the centre of the region and the length of the curve giving a characteristic scale. Thus each curve selects a rectangular patch of pixels in the image which is represented using the PCA-based approaches in Section 4.4.1.

The second representation describes the actual shape of the curve rather than the surround-

ing texture. Each curve is transformed to a canonical position using a similarity transformation such that it starts at the origin and ends at the point $(1, 0)$. If the curve's centroid is below the x-axis then it is flipped both in the x-axis and the line $y = 0.5$, so that the same curve is obtained independent of the edgel ordering. The y value of the curve in this canonical position is sampled at $a-2$ equally spaced x intervals between $(0, 0)$ and $(1, 0)$. Figure 4.1(c) shows curve segments within this canonical space. Since the model is not orientation-invariant, the original orientation of the curve is concatenated to the $a-2$ length vector for each curve, giving a vector of length a (for robustness, orientation is represented as a normalized 2-vector). Combining the 15-vectors from all curves within the image gives $\mathbf{D}^i$.



(a) (b) (c)

Figure 4.1: (a) & (b) Two images containing bottles with curve segments extracted. (c) Curves from many bottle images in a similarity-invariant space — note the clustering.

We have now computed $\mathbf{X}^i$, $\mathbf{S}^i$, and $\mathbf{D}^i$ for use in learning or recognition.

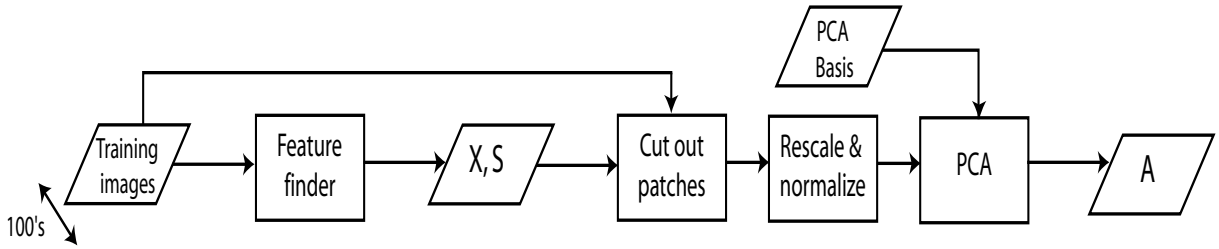


Figure 4.2: The region extraction process during training or recognition. Output variables $\mathbf{X}^i$, $\mathbf{D}^i$, $\mathbf{S}^i$ for every image in the training set.

4.5 Shape

Here we describe the form of $p(\mathbf{X}^i|\mathbf{S}^i, \mathbf{h}, \theta)$ which is the shape term of the object likelihood. We investigate two different forms of shape model. In the first, the shape of the object is represented by a joint Gaussian density of the locations of features within a hypothesis, once they have been transformed into a scale and translation-invariant space. This representation allows the modelling of both inter and intra part variability: interactions between the parts (both attractive and repulsive) as well as uncertainty in location of the part itself. We call this the *full* model.

The second form uses a reduced dependency structure between the parts, so that the location of the model parts are independent, conditioned on a single landmark part. This means only intra-part variance can be modelled. We call this the *star* model. The difference between the two representations is illustrated in Figure 4.3. We first give details for the full shape model, returning to highlight the differences for the star model.

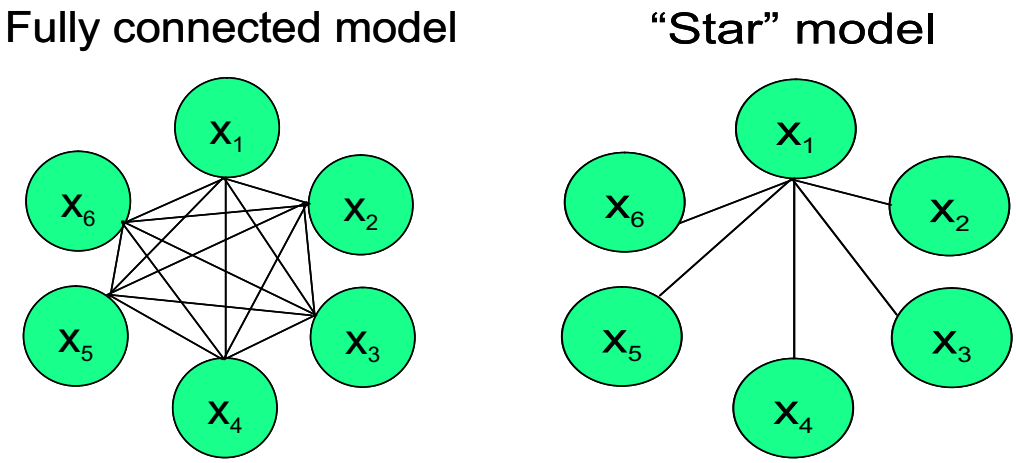


Figure 4.3: Graphical models for the two shape different shape representations. The location of each part is represented as a node in the graph, with the edges indicating the dependencies between them. **(a)** The full model. **(b)** The star model.

4.5.1 Full model

Translation invariance is achieved by using the location of the feature assigned to the first non-occluded part as a landmark. We then model the shape of the remaining features in the hypothesis relative to this landmark feature. Scale invariance is achieved by using the scale of the landmark part to normalize the locations of the other features in the constellation. This

approach avoids an exhaustive search over scale that other methods use, e.g. face detectors such as [107].

If the index of the first non-occluded part is l, then the landmark feature's location is $\mathbf{X}^i(h_l)$ and its scale is $\mathbf{S}^i(h_l)$. $\mathbf{X}^i(\mathbf{h})$ is a $2P$ vector holding the x and y coordinates of each feature in hypothesis h, i.e. $\mathbf{X}^i(\mathbf{h}) = \{x_{h_1}, \ldots, x_{h_P}, y_{h_1}, \ldots, y_{h_P}\}$. To obtain translation invariance, we subtract the location of the landmark from $\mathbf{X}^i(\mathbf{h})$: $\mathbf{X}^i{}_{\mathbb{T}}(\mathbf{h}) = \{x_{h_1} - x_{h_l}, \ldots, x_{h_P} - x_{h_l}, y_{h_1} - y_{h_l}, \ldots, y_{h_P} - y_{h_l}\}$. The operation is performed by using a matrix W_l, which if $l = 1$ is:

$$W_1 = \begin{pmatrix} V_1 & \underline{\underline{\mathbf{0}}} \\ \underline{\underline{\mathbf{0}}} & V_1 \end{pmatrix}, \quad \text{with} \quad V_1 = \begin{pmatrix} -1 & 1 & 0 & \cdots & \cdots & 0 \\ -1 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & & & \vdots \\ -1 & 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}. \tag{4.9}$$

The W_l for other landmarks are permutations of W_1. We model $\mathbf{X}^i{}_{\mathbb{T}}(\mathbf{h})$ with a Gaussian density which has parameters $\theta_{fg}^{shape} = \{\boldsymbol{\mu}, \Sigma\}$. These parameters are also transformed by W_l, mapping the density into a translation-invariant shape space: $\boldsymbol{\mu}_l = W_l\boldsymbol{\mu}$, $\Sigma_l = W_l\Sigma W_l^T$. Note that the dimension of the space is $2(P-1)$, $\boldsymbol{\mu}_l$ being a $2(P-1)$ vector and Σ_l being a $2(P-1)$ by $2(P-1)$ matrix. Unlike appearance whose covariance matrices V_p, V_{bg} are diagonal, Σ_l is a full matrix. The set of P $\boldsymbol{\mu}_l$'s and Σ_l's are equivalent to one another as they are different linear mappings of the same underlying density. This means that we can handle the occlusion of any part while remaining translation-invariant.

A scale invariant representation is obtained by dividing through by $\mathbf{S}^i(h_l)$: $\mathbf{X}^i{}_{\mathbb{TS}}(\mathbf{h}) = \frac{X_{\mathbb{T}}(\mathbf{h})}{S(h_l)}$. When moving between two different landmarks we need to know their relative scale, which is recorded by a relative scale density, described in Section 4.6. Note that $_{\mathbb{T}}$ indicates a representation that is translation invariant, while $_{\mathbb{TS}}$ denotes a representation that is both scale and translation invariant.

All features not included in the hypothesis are considered as arising from the background. The model for the background assumes features to be spread uniformly over the image (which has area α^i), with locations independent of the foreground locations. We also assume that the landmark feature can occur anywhere in the image, so its location is modelled by a uniform

density of $1/\alpha^i$.

$$p(\mathbf{X}^i|\mathbf{S}^i,\mathbf{h},\theta_{fg}) = \begin{cases} \prod_t \left(\frac{1}{\alpha^i}\right)^{N_t} & \text{if } \sum_t f_t = 0 \\ \left(\frac{1}{\alpha^i}\mathrm{G}(\mathbf{X}^i{}_{\mathbb{TS}}(\mathbf{h})|\boldsymbol{\mu}_l,\Sigma_l)\right)\prod_t \left(\frac{1}{\alpha^i}\right)^{n_t} & \text{otherwise} \end{cases} \tag{4.10}$$

If a part is occluded then we marginalize it out, which for a Gaussian entails deleting the appropriate dimensions from the mean and covariance matrix and adjusting the normalization constant. If all parts are missing ($\sum_t f_t = 0$) then there is no foreground shape model and all points are modelled by the background. See [108] for more details.

If no object is present, then all detections are in the background and are consequently modelled by a uniform distribution:

$$p(\mathbf{X}^i|\mathbf{S}^i,\mathbf{h}_0,\theta_{bg}) = \prod_t \left(\frac{1}{\alpha^i}\right)^{N_t} \tag{4.11}$$

Again, this is a constant, so we can cancel between (4.10) and (4.11) for the likelihood ratio in (4.2) to give (assuming at least one part is visible):

$$\frac{p(\mathbf{X}^i|\mathbf{S}^i,\mathbf{h},\theta_{fg})}{p(\mathbf{X}^i|\mathbf{S}^i,\mathbf{h}_0,\theta_{bg})} = \left(\frac{1}{\alpha^i}\mathrm{G}(\mathbf{X}^i{}_{\mathbb{TS}}(\mathbf{h})|\boldsymbol{\mu}_l,\Sigma_l)\right)\prod_t (\alpha^i)^{f_t} \tag{4.12}$$

Note that if one or zero parts are visible, then (4.12) is 1.

4.5.2 Star model

While the majority of the details above apply to the star model, there are some important exceptions:

- The choice of landmark is fixed to one part in the model. One implication of this is that we cannot have this part occluded, thus its presence is enforced at all times. The non-landmark parts can be occluded as before.
- The covariance matrix, Σ_l is no longer a full matrix, now being block diagonal (the blocks being of size 2).
- The occlusion term, $p(\mathbf{b}|\mathbf{B})$, as described in Section 4.7 is no longer a joint density over

the parts occlusions. It is replaced a vector of length $P-1$, modelling the binomial probabilities of each non-landmark part independently being present.

The simplified nature of the model has many benefits that will be discussed in Section 5.1.

4.6 Relative scale

Here we describe the form of $p(\mathbf{S}^i|\mathbf{h},\theta)$ which is the relative scale term of the object likelihood. This term has the same structure as the shape term. The scale of parts relative to the scale of the landmark feature is modelled by a Gaussian density in log space which has parameters $\theta_{fg}^{scale} = \{\mathbf{t}, U\}$. Again, since the landmark feature could belong to any of the P parts, these parameters are really a set of equivalent $\mathbf{t}_l, U_l$'s. The parts are assumed to be independent of one another, thus U_l is a diagonal $(P-1)$ by $(P-1)$ matrix, with $\mathbf{t}_l$ being a $(P-1)$ vector. The background model assumes a uniform distribution over scale (within a range r).

$$p(\mathbf{S}^i|\mathbf{h},\theta_{fg}) = \begin{cases} \prod_t \left(\frac{1}{r}\right)^{N_t} & \text{if } \sum_t f_t = 0 \\ \left(\frac{1}{r}\mathrm{G}(\log \mathbf{S}^i_{\mathbb{T}}(\mathbf{h})|\mathbf{t}_l, U_l)\right) \prod_t \left(\frac{1}{r}\right)^{n_t} & \text{otherwise} \end{cases} \tag{4.13}$$

If the object is not present, all detections are modelled by the uniform distribution:

$$p(\mathbf{S}^i|\mathbf{h}_0,\theta_{bg}) = \prod_t \left(\frac{1}{r}\right)^{N_t} \tag{4.14}$$

Thus the ratio of likelihood becomes (assuming at least one part is visible):

$$\frac{p(\mathbf{S}^i|\mathbf{h},\theta_{fg})}{p(\mathbf{S}^i|\mathbf{h}_0,\theta_{bg})} = \mathrm{G}(\log \mathbf{S}^i_{\mathbb{T}}(\mathbf{h})|\mathbf{t}_l, U_l) \prod_t r^{f_t-1} \tag{4.15}$$

4.7 Occlusion and Statistics of the feature finder

$$p(\mathbf{h}|\theta_{fg}) = \prod_t p_{Poiss}(n_t|M_t)\, \frac{1}{{}^nC_r(N_t, f_t)}\, p(\mathbf{b}|\mathbf{B}) \tag{4.16}$$

The hypothesis $\mathbf{h}$ contains three types of information thus its distribution is a product of three terms. The first term models the number of features in the background, using a Poisson distribution, which has a mean M. The second is a book-keeping term for the hypothesis

variable: we are picking f_t features of each type from a total of N_t and since we have no bias toward particular features, all combinations are equally likely thus it is a constant for all $\mathbf{h}$. The last term models the occlusion of parts within the model. In the case of the full shape model, a joint distribution on the occlusions of model parts is used with a multinomial density (of size 2^P) modelling all possible occlusion patterns $\mathbf{b}$, having a parameter $\mathbf{B}$. This joint distribution allows the modelling of correlations in occlusion: nearby parts are more often occluded together than far apart things. In the case of the star shape model, the independent occlusion probabilities of the non-landmark parts are modelled by a vector of length $P-1$, each element being a binomial probability.

In the null case, we only have one possible hypothesis, $\mathbf{h}_0$, so the only term from (4.16) that remains is the Poisson which now has to account for all features belonging to the background:

$$p(\mathbf{h}_0|\theta_{bg}) = \prod_t p_{Poiss}(N_t|M_t) \tag{4.17}$$

Thus the ratio becomes:

$$\frac{p(\mathbf{h}|\theta_{fg})}{p(\mathbf{h}|\theta_{bg})} = \frac{\prod_t p_{Poiss}(n_t|M_t)}{\prod_t p_{Poiss}(N_t|M_t)} \frac{1}{{}^nC_r(N_t, f_t)} p(\mathbf{b}|\mathbf{B}) \tag{4.18}$$

4.8 Multiple aspects via a mixture of constellation models

An important limitation of the model, as presented, is that we can only model one aspect of the object. While this is a limitation, our approach can be extended to multiple aspects by using a mixture of constellation models, in the manner of Weber *et al.* [109].

Since the constellation model is generative, it is simple to have a mixture model, with each component being a constellation model. For example, if we have Ω components in total, our mixture model would look like:

$$\begin{aligned} p(\mathbf{D}^i, \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}|\theta_{fg}) &= \sum_{\omega=1}^{\Omega} p(\mathbf{D}^i, \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}, \omega|\theta_{fg}) && (4.19)\\ &= \sum_{\omega=1}^{\Omega} p(\mathbf{D}^i, \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}|\omega, \theta_{fg}) P(\omega|\theta_{fg}) && (4.20)\\ &= \sum_{\omega=1}^{\Omega} \pi_\omega \, p(\mathbf{D}^i, \mathbf{X}^i, \mathbf{S}^i, \mathbf{h}|\omega, \theta_{fg}) && (4.21) \end{aligned}$$

$\pi_\omega \stackrel{\text{def}}{=} P(\omega|\mathcal{O}_1)$ is the mixing component, $\sum_\omega P(\omega|\mathcal{O}_1) = 1$. It is also a parameter of our model. Our likelihood expression is now also conditioned on ω, so each foreground mixture component has a separate, independent, set of parameters, θ_ω. The only term that is shared amongst all the components is the Poisson background distribution, since all components share the same background. Therefore the parameter in the Poisson distribution, $\mathbf{M}$ is common to all θ_ω.

More sophisticated variants of mixture models can also be easily formed. For example, it would be possible to have a mixture of descriptor densities but a single shape component. This could be done by forcing $\mu_{l,\omega}$ and $\Sigma_{l,\omega}$ to be the same across all mixture components, whilst letting $\mathbf{c}_\omega$ and V_ω differ.

4.9 Model discussion

Having laid out the structure of the model we have made a variety of design decisions that need justifying.

4.9.1 Appearance term

The assumption of independence between parts' appearance is chosen for its simplicity, although conditioning on the appearance of the landmark could be useful for removing lighting variations.

4.9.2 Alternative forms of shape model

We have chosen to achieve translation invariance by conditioning on a single model part and scale invariance by using local feature information, but a variety of other methods exist to achieve the same goal.

The first alternative is one followed by Helmer and Lowe [51] where the features are mapped into the space of the model using a transformation ϕ which can be regarded as a hidden variable which can be integrated out:

$$p(\mathbf{X}^i|\mathbf{h}, \theta_{fg}) = \int p(\mathbf{X}^i, \phi|\mathbf{h}, \theta_{fg})\, d\phi \tag{4.22}$$

This integral is problematic to compute, so it can be approximated:

$$p(\mathbf{X}^i|\mathbf{h}, \theta_{fg}) \approx \underset{\phi}{\text{argmax}}\, p(\mathbf{X}^i, \phi|\mathbf{h}, \theta_{fg}) \tag{4.23}$$

If ϕ consists of a translation and scaling then the optimal transformation for a given hypothesis $\mathbf{h}$ can be computed via a weighted least squares solution, the weights being the variances on location. While this approach may give a more accurate answer than our representation, the disadvantage is that for every $\mathbf{h}$, a least squares problem must be solved as opposed to the simple translation and division operations required with our approach.

A drawback of our approach to scale invariance is that we use a local measure of scale. For each hypothesis, the constellation of points is divided through by the scale of the landmark. This means that any noise in the scale measurement will distort the shape model, artificially increasing its variance. Additionally, the error in measuring the local orientation of a region prevents us from obtaining rotation invariance.

The second alternative is to use further parts and additional landmarks, to achieve further degrees of invariance, abandoning the use of local scale and orientation measurements from the features. Leung *et al.* [68] proposed an affine invariant shape representation by the use of three model parts as a basis, transforming the Gaussian density of the remaining model parts into a Dryden-Mardia density [74] in shape space. The complex nature of this density restricted its use to recognition, hence learning was performed manually. Although recently methods have been devised to estimate the parameters of Dryden-Mardia densities [112] these are slow.

We therefore use a single landmark and local scale information, restricting ourselves to scale and translation invariance.

4.9.3 Improvements over Weber *et al.*

The model presented above is based on an earlier version developed by Markus Weber and Max Welling at Caltech. It is worth comparing the two to see the differences. The previous scheme lacked the probabilistic representation of appearance. Instead, appearance and shape were learnt in two separate stages:

1. Interest points are extracted from training images and clustered, giving a codebook of possible features.

2. A greedy search is used to select a subset of P features from the codebook, learn a shape-only model and assess it on a validation set.

The problem with this approach is that if the first stage does not produce a sensible codebook then the second stage is doomed. The clustering procedure within the first step can get overwhelmed with lots of low-level features (i.e. orientated edges) which lack discriminative power.

In the new version of the model, both are learnt simultaneously, with the probabilistic representation designed to be more flexible than the correlation templates used previously. Additional contributions include the introduction of scale-invariance; the relative scale term and the direct modelling of the output of a heterogeneous set of feature detectors.

4.9.4 Model assumptions

The assumptions inherent in the model are summarized below:

1. The appearance of the parts are independent of one another.
2. The distribution of appearance for each part is modelled by a single axis-aligned Gaussian.
3. The background feature's descriptors can be modelled by a single Gaussian density.
4. The location of the parts is independent to their appearance.
5. The scale of the parts is independent to their appearance.
6. A Gaussian is a suitable model for the relative location of the parts.
7. The background detections are independent to the foreground and each other.

The validity of many of these assumptions will be examined in Chapter 6.

4.10 Model structure summary

The model encompasses many of the properties of an object, all in a probabilistic way, so this model can represent both geometrically constrained objects (where the shape density would have a small covariance) and objects with distinctive appearance but lacking geometric form (the appearance densities would be tight, but the shape density would now be looser).

Using (4.8),(4.12),(4.15) and (4.18) we can write the likelihood ratio from (4.2) as:

$$\frac{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i|\theta_{fg})}{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i|\theta_{bg})} = \sum_{\mathbf{h}\in H} \frac{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\theta_{fg})}{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}_0|\theta_{bg})} = \sum_{\mathbf{h}\in H} \prod_{p=1}^{P} \left(\frac{\mathrm{G}(\mathbf{D}^i(h_p)|\mathbf{c}_p, V_p)}{\mathrm{G}(\mathbf{D}^i(h_p)|\mathbf{c}_{bg}, V_{bg})} \right)^{b_p}$$
$$\frac{\mathrm{G}(\mathbf{X}^i{}_{\mathbb{TS}}(\mathbf{h})|\boldsymbol{\mu}_l, \Sigma_l)\ \mathrm{G}(\log\ \mathbf{S}^i_{\mathbb{T}}(\mathbf{h})|\mathbf{t}_l, U_l)\ \prod_t (\alpha r)^{f_t - 1}\ p_{Poiss}(n_t|M_t) p(\mathbf{b}|\mathbf{B})}{\prod_t p_{Poiss}(N_t|M_t)\ {}^nC_r(N_t, f_t)} \tag{4.24}$$

The intuition is that the majority of the hypotheses will be low scoring as they will be picking up features from background clutter on the image but hopefully a few features will genuinely be part of the object and hypotheses using these will score highly. This observation suggests methods for efficient learning and recognition with the Constellation Model, which we discuss in the next chapter.

Chapter 5

Learning and Recognition with the Constellation model

In the previous chapter we described the structure of the constellation model. We now flesh out the practical details of the features used; the model learning scheme and how it is applied in recognition. To illustrate the system, we use a running example in the form of the motorbike class. The input for both learning and recognition is a set of features extracted from the image. The detectors used are described in Section 2.3 while the representation scheme for the appearance of each region is explained in Section 4.4.1. Figure 5.1 illustrates six typical images from the motorbike dataset with regions from the Kadir & Brady detector overlaid.

5.1 Learning

In a weakly supervised learning scenario, one is presented with a collection of images containing examples of objects belonging to a given class amongst clutter. However the position and scale of the object within each image is unknown; no correspondence between exemplars is given; parts of the object may be missing or occluded. The challenge is to make sense of this profusion of data. Weber *et al.* [109, 108] approached the problem of weakly supervised learning of object categories in clutter as a maximum likelihood estimation. For this purpose they derived an EM algorithm for the constellation model. We follow their approach in deriving an EM algorithm to estimate the parameters of our improved model.

The task of learning is to estimate the parameters $\theta_{fg} = \{\boldsymbol{\mu}, \Sigma, \mathbf{c}, V, \mathbf{M}, \mathbf{B}, \mathbf{t}, U\}$ of the

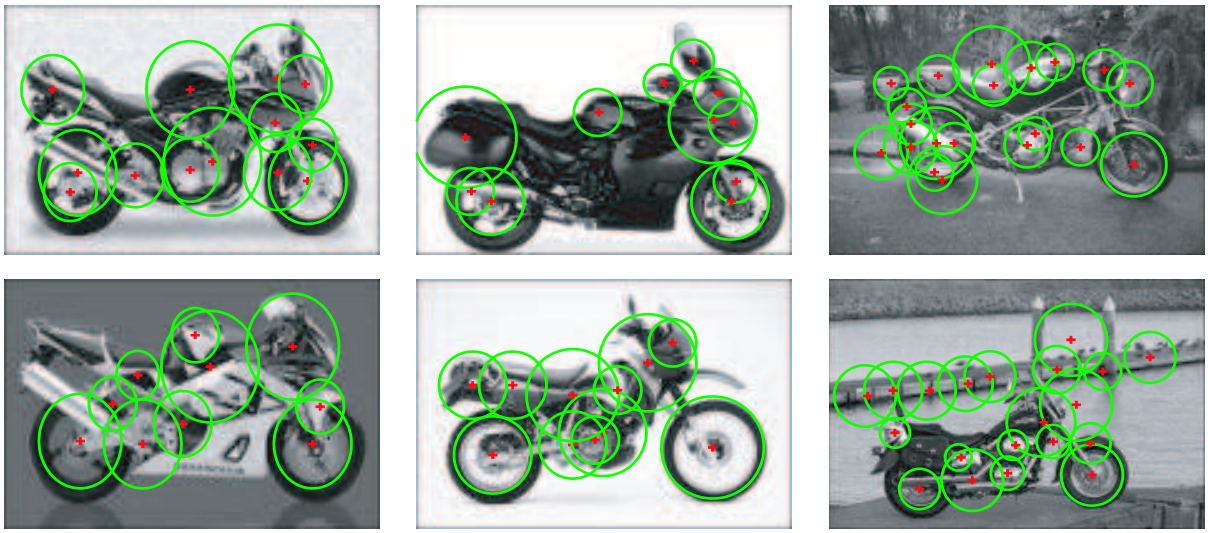


Figure 5.1: Six typical motorbikes images with the output of the Kadir-Brady operator overlaid. The +'s illustrate the centre of the salient region, while the circles show the scale of the region. Notice how the operator fires more frequently on more salient regions, ignoring the uniform background present in some of the images.

model discussed in the previous chapter. The goal is to find the parameters $\hat{\theta}_{ML}$ which best explain the data $\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i$ from all the training images, that is maximize the likelihood: $\hat{\theta}_{ML} = \underset{\theta}{arg\,max} p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i | \, \theta_{fg})$. Note that the parameters of the background, θ_{bg}, are constant during learning.

Learning is carried out using the expectation-maximization (EM) algorithm [28] which iteratively converges, from some random initial value of θ_{fg} to a maximum (which might not be a global one). Figure 5.2 summarizes the process.

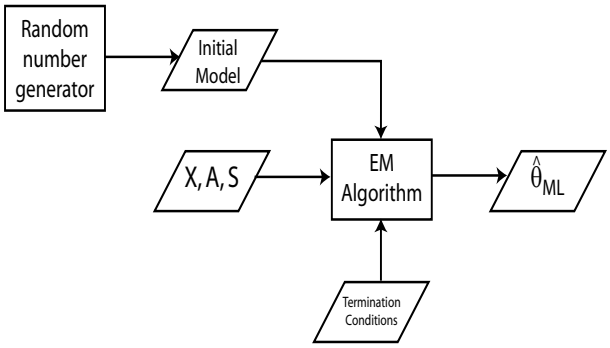


Figure 5.2: Learning a model

We now look at each stage in the learning procedure, giving practical details of its implementation and performance, using the motorbike dataset as an example. We assume that $\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i$ have already been extracted from the images, examples of which are shown in Figure 5.1. In this example, we are using the gradient based PCA representation, with $k = 11$ and $a = 15$.

5.1.1 Initialization

Initially we have no knowledge about the structure of the object to be learnt so we are forced to initialize the model parameters randomly. However, the model which has a large number of parameters, must be initialized sensibly to ensure that the model will converge to a reasonable maximum. For shape, the means are set randomly over the area of the image and the covariances to be large enough so that all hypotheses have a roughly equal weighting, thereby avoiding a bias toward nearby points. The appearance densities are initialized to zero mean, plus a small random perturbation, while the variances are set to be large. The same initialization settings are used in all experiments. In Figure 5.3 we show three typical model initializations of the shape term (the appearance term is hard to visualize due to the large number of dimensions).

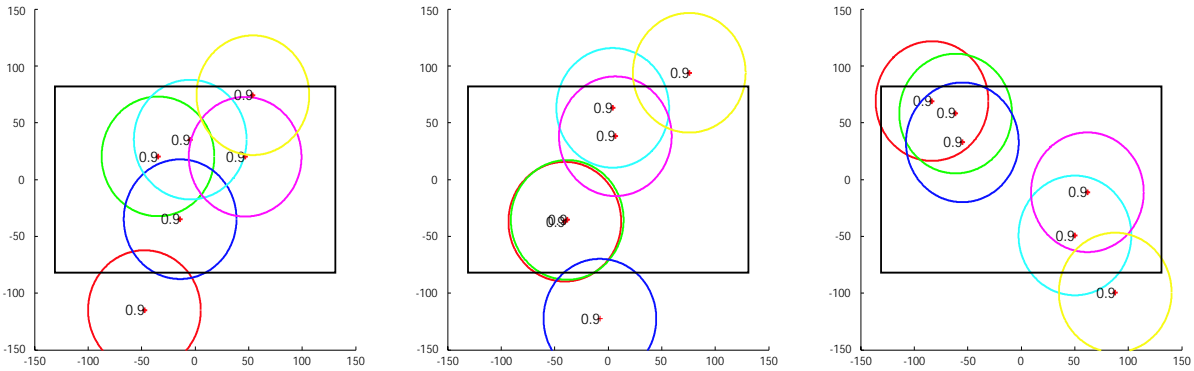


Figure 5.3: Three typical initializations for a 6 part motorbike shape model. The circles represent the variance of each part at 1 standard deviation (the inter-part covariance terms, which cannot easily be shown, are set to zero) with the mean being the centre of the circles. The probability of each part being present is shown just to the left of the mean. The average image size is indicated by the black box. As the images are resized to a constant width and their aspect ratio is unknown, we ensure that tall images are not penalized by allowing the initialization of some of the parts to lie outside the mean image box. Axis units are pixels. The variances here are referred to the centroid of the model.

5.1.2 EM update equations

The algorithm has two stages: (i) the E-step in which, given the current value of θ_{fg} at iteration k, θ_{fg}^{k}, some sufficient statistics are computed and (ii) the M-step where we compute the parameters for the next iteration, θ_{fg}^{k+1} using these sufficient statistics.

We now give the equations for both the E-step and M-step. The E-step requires us to compute the posterior density of the hidden variables, which in our case are the hypotheses.

This is calculated using the joint:

$$p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k) = \frac{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\, \theta_{fg}^k)}{\sum_{h \in H^i} p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\, \theta_{fg}^k)} = \frac{\frac{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\, \theta_{fg}^k)}{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}_0|\, \theta_{bg})}}{\sum_{h \in H^i} \frac{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\, \theta_{fg}^k)}{p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}_0|\, \theta_{bg})}} \tag{5.1}$$

We divide through by $p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}_0|\, \theta_{bg})$ as it is easier to compute the joint ratio rather than the joint directly. We then calculate the following sufficient statistics for each image, i from which we have previously extracted $\mathbf{X}^i, \mathbf{D}^i, \mathbf{S}^i$: $E[\mathbf{X}^i{}_{\mathbb{TS}}]$, $E[\mathbf{X}^i{}_{\mathbb{TS}}\mathbf{X}^i{}_{\mathbb{TS}}{}^T]$, $E[\mathbf{D}_p^i]$, $E[\mathbf{D}_p^i\mathbf{D}_p^{i\,T}]$, $E[\mathbf{S}_{\mathbb{T}}^i]$, $E[\mathbf{S}_{\mathbb{T}}^i\mathbf{S}_{\mathbb{T}}^{i\,T}]$, $E[\mathbf{n}^i]$, $E[\mathbf{B}^i]$ where the expectation is taken with respect to the posterior, $p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k)$, for example:

$$E[\mathbf{X}^i{}_{\mathbb{TS}}] = \sum_{h \in H^i} p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k)\, \mathbf{X}^i{}_{\mathbb{TS}}(h) \tag{5.2}$$

Note that for simplicity we have not considered the case of missing data. The extensions to the above rules for dealing with this may be found in [108]. The general principle is to condition on the features that are present to work out the expected values of those that are missing.

In the M-step we then compute $\theta_{fg}^{k+1} = \{\boldsymbol{\mu}^{k+1}, \Sigma^{k+1}, \mathbf{c}^{k+1}, V^{k+1}, \mathbf{t}^{k+1}, U^{k+1}, \mathbf{M}^{k+1}, \mathbf{D}^{k+1}\}$. The new parameter values are chosen to maximize the function:

$$Q = \sum_{i=1}^{I} \sum_{h \in H^i} p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k)\, log\, p(\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \mathbf{h}|\, \theta_{fg}) \tag{5.3}$$

where I is total number of training images. This is done by taking partial derivatives of Q with respect to each of the parameters in θ_{fg}, setting them equal to zero, for example:

$$\frac{\partial Q}{\partial \boldsymbol{\mu}} = \sum_{i=1}^{I} \sum_{h \in H^i} p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k) \Sigma^{-1} (\mathbf{X}^i{}_{\mathbb{TS}}(h) - \boldsymbol{\mu}) = 0 \tag{5.4}$$

Rearranging (5.4) and noting that $\sum_{h \in H^i} p(\mathbf{h}|\mathbf{X}^i, \mathbf{S}^i, \mathbf{D}^i, \theta_{fg}^k) = 1$, we obtain a closed-form solution for $\boldsymbol{\mu}^{k+1}$: $\boldsymbol{\mu}^{k+1} = \frac{1}{I} \sum_{i=1}^{I} E[\mathbf{X}^i{}_{\mathbb{TS}}]$. The updates for the other parameters are derived

in a similar manner to give:

$$
\begin{aligned}
\boldsymbol{\mu}^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{X}^i{}_{\mathbb{TS}}] & \Sigma^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{X}^i{}_{\mathbb{TS}}\mathbf{X}^i{}_{\mathbb{TS}}{}^T] - \boldsymbol{\mu}^{k+1}\boldsymbol{\mu}^{k+1T} \\
\mathbf{c}_p^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{D}_p^i] \;\; \forall p \in P & V_p^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{D}_p^i \mathbf{D}_p^i{}^T] - \mathbf{c}_p^{k+1}\mathbf{c}_p^{k+1T} \;\; \forall p \in P \\
\mathbf{t}^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{S}_{\mathbb{T}}^i] & U^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{S}_{\mathbb{T}}^i \mathbf{S}_{\mathbb{T}}^i{}^T] - \mathbf{t}^{k+1}\mathbf{t}^{k+1T} \\
\mathbf{M}^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{n}^i] & \mathbf{D}^{k+1} &= \frac{1}{I}\sum_{i=1}^{I} E[\mathbf{B}^i]
\end{aligned}
$$

The two stages are then repeated until θ_{fg}^k converges to a stable point. See section 5.1.5 below for a discussion on the convergence properties of the algorithm.

5.1.3 Computational considerations

In computing the sufficient statistics in the E-step we need to evaluate the likelihood for every hypothesis. Since there are $O(N^P)$ per image, this is the major computational bottleneck in our approach. Possible ways around this include:

1. Use the mode: Approximate the summation over $\mathbf{h}$ by just taking the mode, i.e. the best hypothesis in each image. The problem with this is that the initial assignments are totally random and so initially picking the best hypothesis is unlikely to be close to the optimal one and it is difficult to escape from such local minima in subsequent iterations. The practical consequences are that the model is more prone to numerical explosions (probabilities go to zero somewhere); or the models converge to bad local maxima. See Figure 5.9(b) for an example of the latter.

2. Sample hypotheses: While sampling methods could be used to evaluate the marginalization in the E-step [55], the imposition of constraints (e.g. see Section 5.1.5) on possible hypotheses introduces complications. These constraints mean that it would be more difficult to move through the space of possible hypotheses, since many proposed moves would not be valid, increasing the chances of hitting local minima. For this reason, we prefer more direct computation methods.

3. Reduce the dependencies: The cause of our problems is assuming that the location of all

parts is dependent on one another. A simpler dependency structure could be adopted, like the star model introduced in Section 4.5.2. We consider this model structure in Section 5.3.

4. Efficient search methods: Only a proportion of the hypotheses have a high probability thus we can accurately approximate the summation over all hypotheses by just considering this subset. By utilizing various heuristics, specific to our application, we can efficiently compute the few hypotheses contributing much of the probability mass. The details of this are now investigated in Section 5.1.4.

5.1.4 Efficient search methods for the full model

Computing the very small portion of the hypotheses that have a high probability enables the learning procedure to run in a reasonable time.

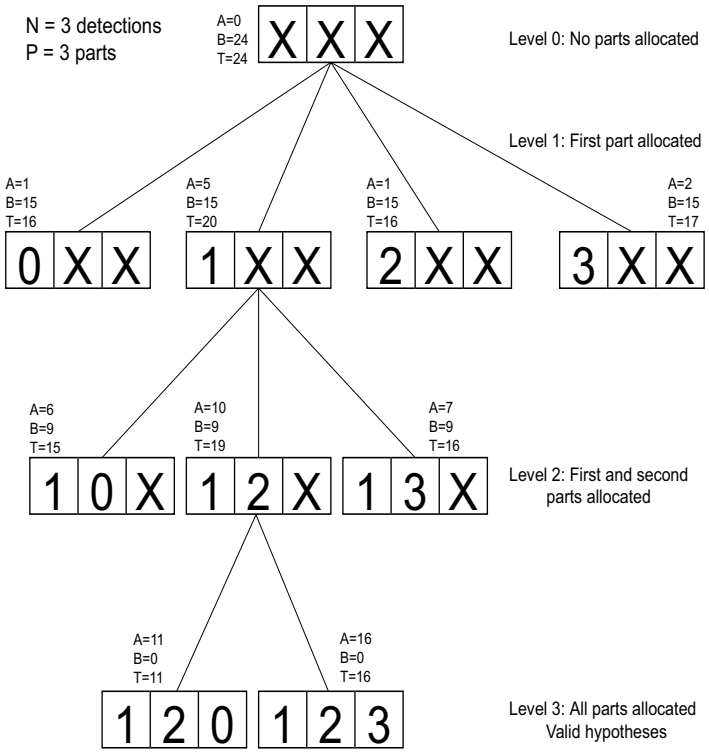
A tree structure is used to search the space of all possible hypotheses. The leaves of the tree are complete hypotheses with each level representing a part: moving down the tree, features are allocated to parts until a complete hypothesis is formed. At a node within the tree, an upper-bound on the probability of remaining, unallocated parts can be computed enabling us to employ the A^* algorithm [48, 50]. This allows the efficient exploration of the tree, resulting in the guaranteed discovery of the best hypothesis. This can be removed from the tree and the search continued until the next best hypothesis is found. In this manner, we can extract the hypotheses ordered by likelihood. Figure 5.4 shows a toy example of this process.

If q parts have been allocated, the upper bound on the probability for the remaining $P - q$ parts is easily computed thanks to the form of the densities in the model. Given the occlusion states of the unallocated parts, the upper bound is a constant, thus it becomes a case of finding the maximum upper bound for each of the 2^{P-q} possible states.

A binary heap stores the list of open branches on the tree, having $\log n$ access time. Conditional densities for each part (i.e. conditioning on previously allocated parts) are pre-computed to minimize the computation necessary at each branch. Details of the A^* search can be found in [41].

For each image, we compute all hypotheses until they become less than some threshold (e^{-15}) smaller than the best hypothesis. Figure 5.5(a) shows how the number of hypotheses

varies for a given likelihood. This threshold was chosen to ensure that the learning progressed within a reasonable time while evaluating as many hypotheses as possible.

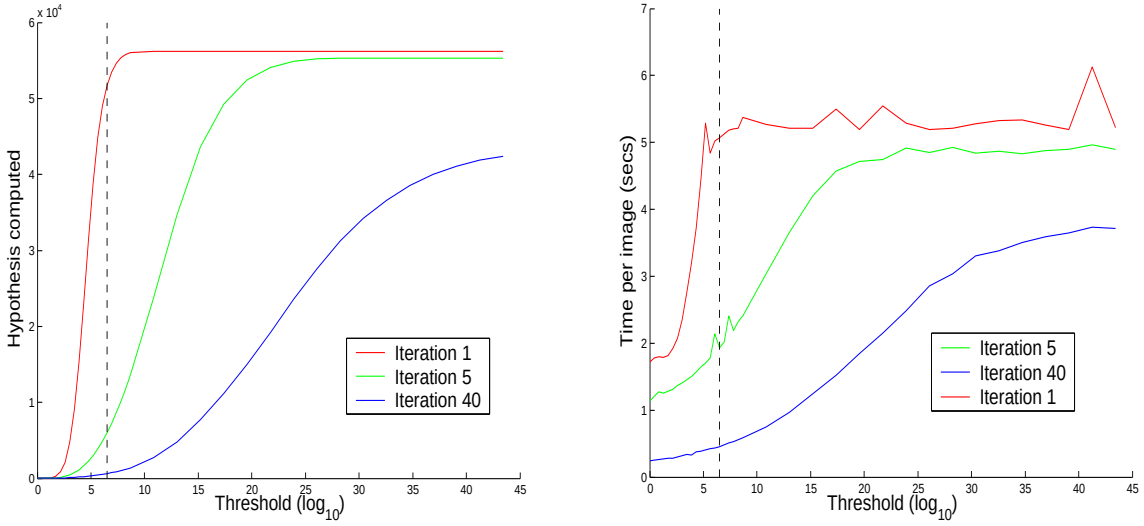


(a)

Figure 5.4: Illustration of A^* search process for a 3 part toy model and an image with 3 regions. As the tree is descended, regions are allocated to parts, with the leaves of the tree constituting complete hypotheses. The score (T) of each node in the tree is a sum of the likelihood of the allocated features (A) and an upper-bound on the likelihood of the remaining, unallocated features (B). The list of open nodes is stored in a binary heap, with the node with the highest overall value (T) being the one opened next. By repeatedly removing complete nodes from the tree, the hypotheses can be extracted in order of likelihood.

Additionally, space search methods are used to prune the branches explored at each new node in the tree. At a given level of the tree, the joint density of the shape term allows the density of location of the current part to be computed by conditioning on the previously allocated parts. Only a subset of the N detections need be evaluated by this density: we assume that we can neglect detections if their probability is worse than having all remaining parts be missing. Since the occlusion probabilities are constant for a given learning iteration, this gives a threshold which truncates the density. If the covariance of the density is small, only the best few detections need to be evaluated, enabling significant numbers of hypotheses to be ignored.

Despite using these efficient methods, learning a $P = 6$–7 part model with $N = 20$–30



(a) (b)

Figure 5.5: (a) A graph showing how the number of hypotheses evaluated increases as the likelihood drops below that of the best hypothesis on a typical motorbike model. The vertical line shows the value of the threshold used in experiments. The three curves correspond to different stages of learning: red – at the beginning when the model variances are large; green – in the middle and blue – when the model variances are small (see Fig. 5.7(c) for evolution of model during learning). Note that as the model variances decrease, the number of hypotheses evaluated decreases. (b) As for (a), but the y-axis is evaluation time per image.

features per image (a practical maximum), using 400 training images, takes around 24 hours to run. This equates to spending 3-4 seconds per image, on average, at each iteration (given a total running time of 24 hours, with 400 training images and 50 EM iterations). It should be noted that learning only needs to be performed once per category, due to the good convergence properties as discussed in Section 5.1.5.

It is worth noting that just finding the best hypothesis (the mode), is not that much quicker than taking the small subset of high scoring hypotheses (see Figure 5.5(b)), since a reasonable portion of the tree structure must be explored before a complete hypothesis is found. This provides another justification for summing over multiple hypotheses rather than just taking the best.

5.1.5 Convergence

Table 5.1 illustrates how the number of parameters in the model grows with the number of parts, (assuming $a = 15$). Despite the large number of parameters in the model, its conver-

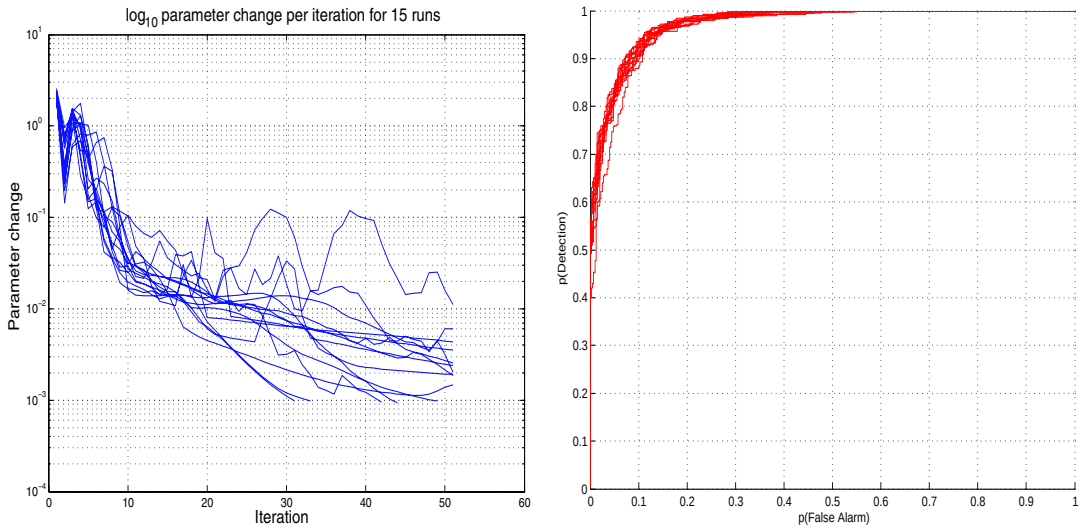
Parts	2	3	4	5	6	7
# parameters	77	123	177	243	329	451

Table 5.1: Relationship between number of parameters and number of parts in model

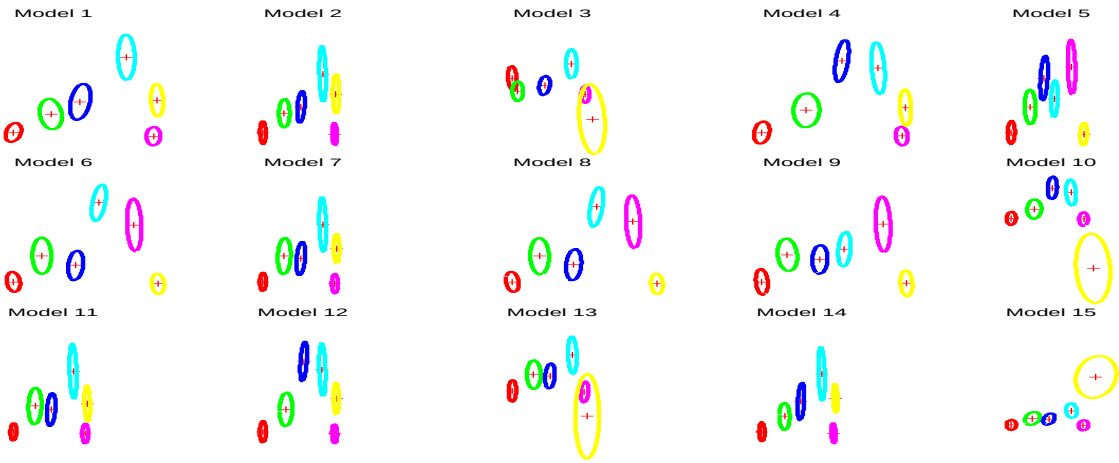
gence properties are respectable. Figure 5.6 shows the convergence properties and classification performance of 15 models started from different initial conditions but with identical data. Note that while the models converge at different rates and to different points in parameter space (see the different shape models for each run in Figure 5.6(c)),the ROC curves of test set performance are very similar, the standard deviation at equal-error rate being 0.6%. Figure 5.7(a) shows the shape model evolving throughout the learning process for a typical learning run. Figure 5.7(b) shows the classification performance of the model improving as the model converges. Both figures demonstrate that the majority of the performance is obtained within the early stages of learning. However, occasionally a superior maximum can be found after a large number of iterations, therefore we continue until we are sure that the model has reached a stable point. Two criteria were used to stop the EM iteration: (i) Number of iterations exceeds some limit (50) and (ii) The absolute value of norm of normalized parameter change per iteration drops below some limit (10^{-3} – for the shape term this equates to around 1/10th of a pixel). In practice the former criterion is used more often.

Figure 5.8 gives an insight into the convergence of the model during a typical learning run. Figure 5.8(a) shows the parameter change per iteration steadily reducing until it hits a 10^{-3} limit. Figure 5.8(b) shows how the log-likelihood ratio over all images increases monotonically as it should. The plot also gives a breakdown of the terms within the model. In Figure 5.8(c) the probability of each part being present is shown. Initially, the probability starts out low, but within 10 iterations or so each one has locked onto a sensible signal thus has a high probability. Between 20 and 50 iterations, some fine settling of the probabilities can be seen as the parts converge down on features of the object.

Convergence to a successful model is dependent on a variety of factors, but two important ones are: (a) a consistent set of regions from which to learn and (b) the introduction of an ordering constraint on the x-coordinates of regions within a hypothesis. While the former is discussed in more detail in Section 6.2.1, we now elaborate on the latter. To aid both convergence and speed, an ordering constraint is placed on the x-coordinates of features allocated to parts: the features selected must have a monotonically-increasing x-coordinate. This reduces the total



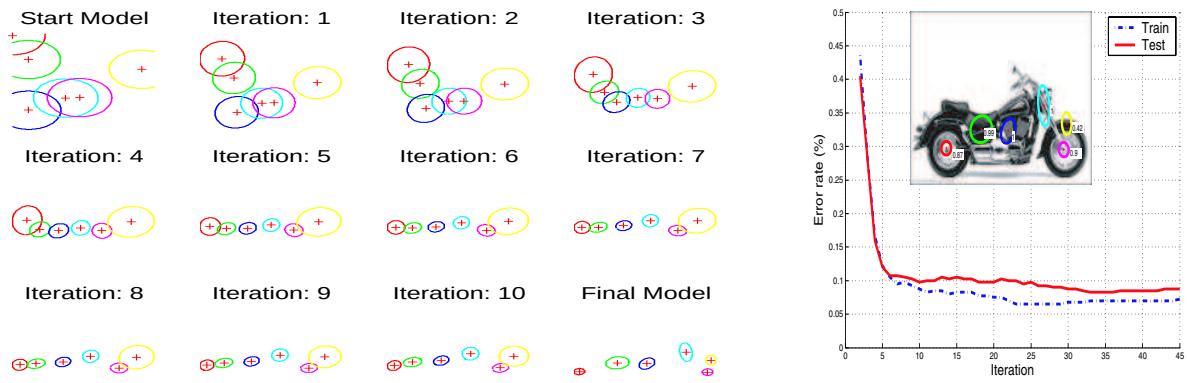
(a) (b)



(c)

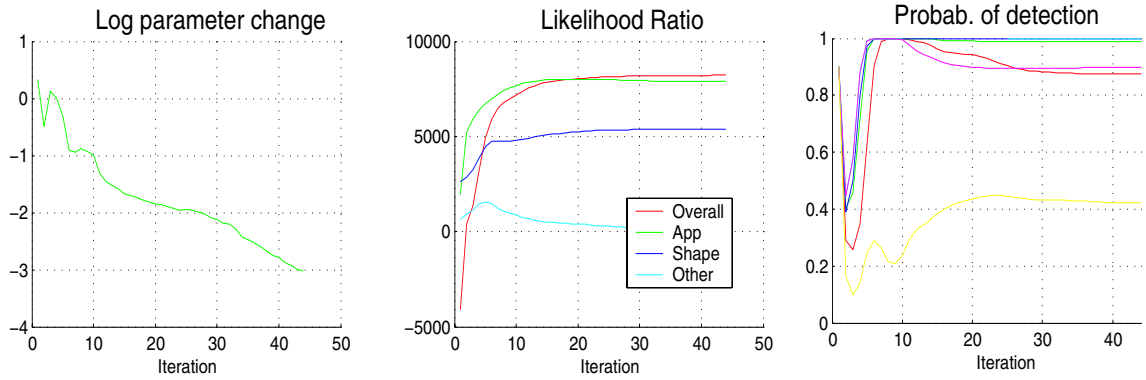
Figure 5.6: 15 learning runs for the motorbike category, each with a different random initialisation. (a) The maximal parameter change per iteration for 15 runs, started at different initial conditions. (b) The 15 ROC curves (evaluating test data) corresponding to each of the runs, the mean error rate is 9.3%, with a standard deviation of 0.6%. (c) The shape models for each of the 15 runs. Despite finding different maxima the wheels are always identified, with only the yellow part varying significantly in its final location.

number of hypotheses by $P!$ but unfortunately imposes an artificial constraint upon the shape of the object. If the object happens to be orientated vertically then this constraint can exclude the best hypothesis. Clearly in this scenario, imposing a constraint on the y-coordinate ordering would resolve the problem but it is not clear how to choose such an appropriate constraint automatically, other than learning models with different ordering constraints and picking the



(a) (b)

Figure 5.7: (a) The evolution of the motorbike shape model throughout learning. (b) Classification performance versus learning iteration. The inset shows the final shape model superimposed on a motorbike image.



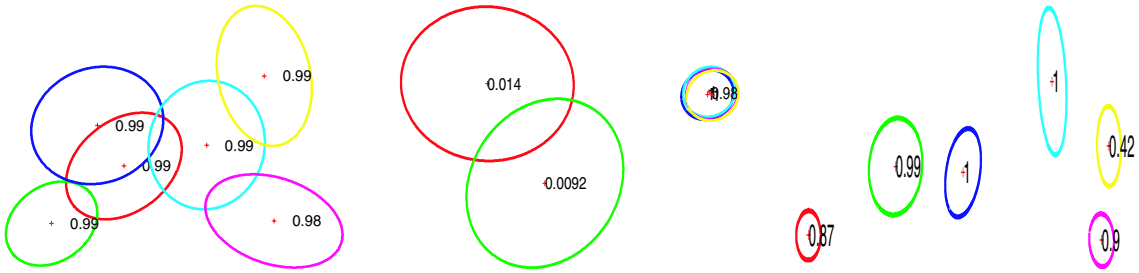
(a) (b) (c)

Figure 5.8: A typical learning run on the motorbikes dataset. (a) The magnitude of largest parameter change versus learning iteration (log-scale) (b) Overall log-likelihood ratio versus iteration. Likelihoods for each of the components within the model are also shown. (c) Evolution of the probability of each part being present. Each colour corresponds to a different part. Note that the probability drops down to a low value for the first couple of iterations before the model finds some structure in the data and the probability picks up again.

one with the highest likelihood. See Figure 5.9(a) for an example of a model learnt without this constraint.

5.1.6 Background model

Since the model is a generative one, the background images are not used in learning except for one instance: the appearance model has a distribution in appearance space modelling background features. Estimating this from foreground data proved inaccurate so the parameters were estimated from a set of background images and not updated within the EM iteration.



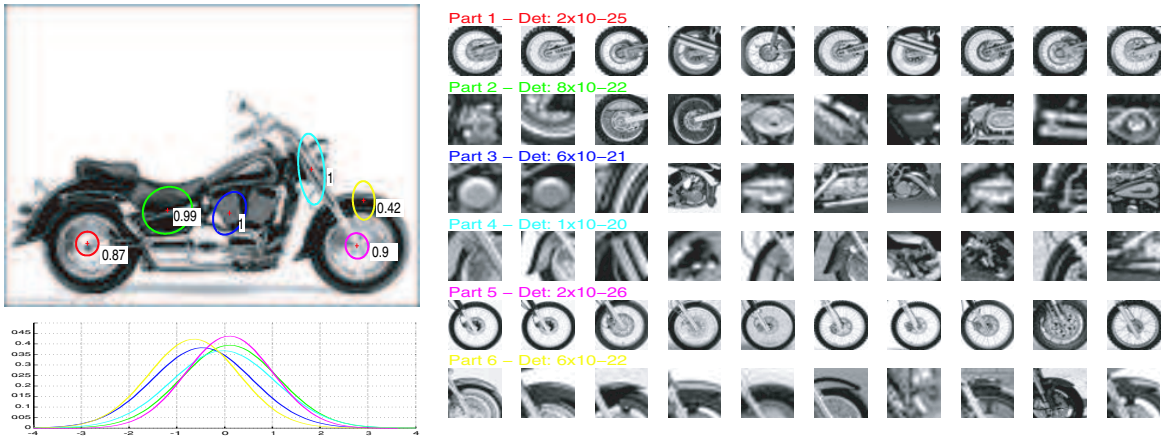
(a) (b) (c)

Figure 5.9: (a) A motorbike shape model learnt using no ordering constraint on regions with hypotheses. Note the large overlapping distributions due to permutations of nearby features. The model is also $P!$ slower to learn than those with ordering constraints imposed. (b) A motorbike shape model learnt just using the best hypothesis in each frame. A poor local maximum has been found with two parts being redundant, having a low probability of occurrence. (c) The shape model obtained if the ordering constraint on the x-coordinates of the hypothesis is imposed.

5.1.7 Final model

In Figure 5.10 we show a complete model, from one of the 15 training runs on the motorbike dataset. It is pleasing to see that a sensible spatial structure has been picked out and that the appearance samples correspond to distinctive parts of the motorbike. In Figure 5.11 the appearance density of the model is analyzed. In Figure 5.11(a) the 15 principal components of appearance are shown. For ease of viewing, the basis is shown in intensity space, having integrated the original gradient space basis. Figure 5.11(b) shows the distribution in appearance space of patches assigned to each model part by the best hypothesis in an image and for the remaining background patches. For example, consider the parts of the model corresponding to the wheels of the motorbike (the 1st (red) and 5th (magenta) parts in Figure 5.10). Looking at the histograms in 5.11(b), we can see that the 8th principal component is a doughnut shape and that the red and magenta histograms are considerably skewed from the background histogram, unlike those for other model parts of this descriptor. The assumption that the appearance data is Gaussian in nature can also be examined in Figure 5.11. For the most part the foreground data is well approximated by a single Gaussian component. The background data seems to follow a Gaussian distribution as well. In Figure 5.11(c) histograms show how discriminative each part is. Although many of the descriptors are weak classifiers, they combine to give a strong classifier. Notice that the parts corresponding to wheels (red and magenta) have a foreground histogram which is quite distinct from the background histogram. Other parts, such as the third

one, are not so discriminative in appearance, having foreground and background distributions that overlap considerably. Instead, this part may contribute in the shape term to the model.



(a) (b)

Figure 5.10: A complete model: (a) Top: Shape density superimposed on an image of a motorbike. The ellipses represent the covariance of each part (the inter-part covariance terms cannot easily be shown) and the probability of each part being present is shown just to the right of the mean. Bottom: The scale distribution of each relative to the first part. (b) Samples belonging to each part (i.e. from the best hypothesis in a training image) which are closest to the mean of the appearance density. The colours correspond to the colours in (a). The determinant of each appearance density is given to provide an idea of the relative tightness of each parts' density. A high exponent corresponds to a tight density with a consistent appearance and vice versa.

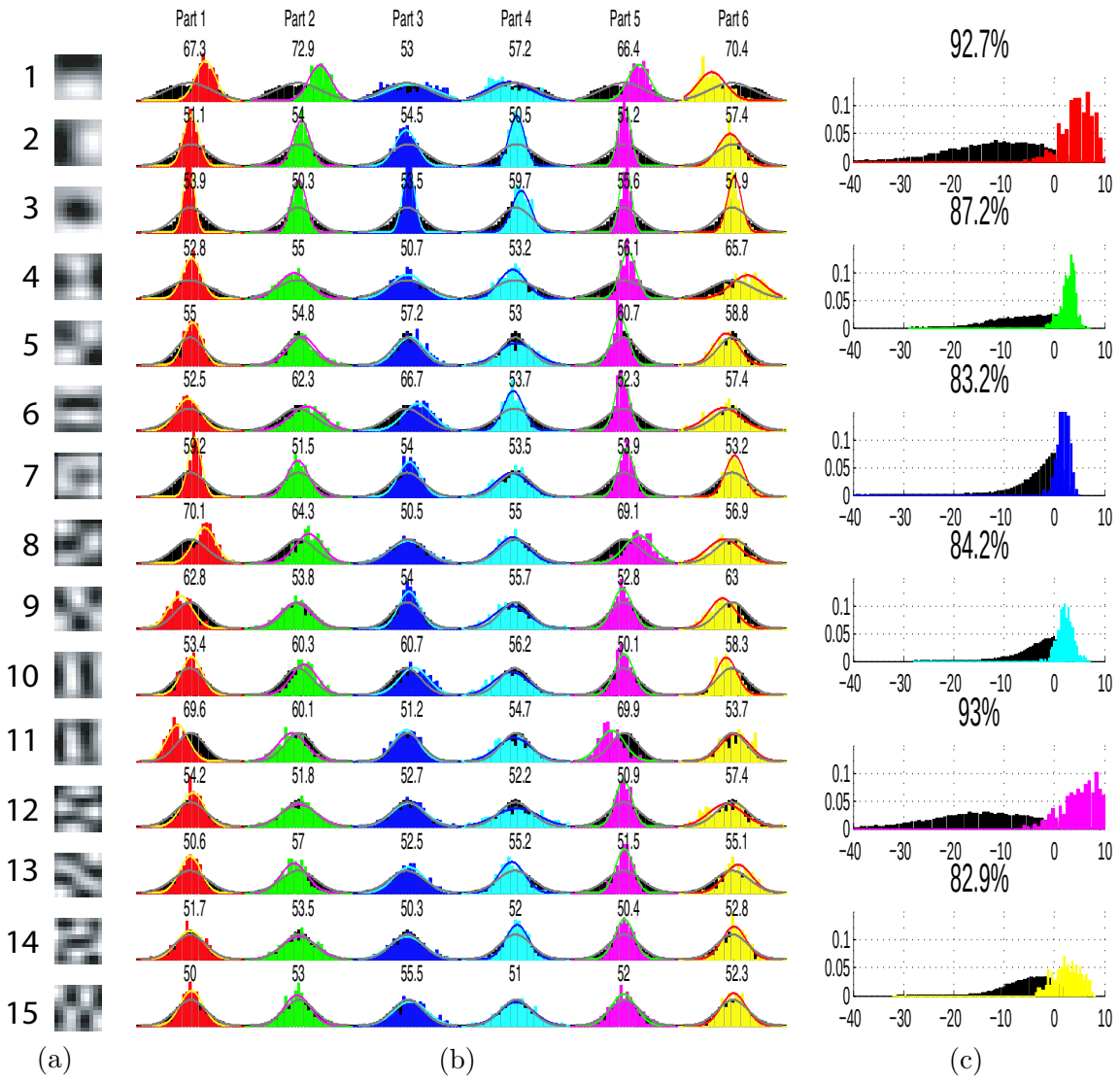


Figure 5.11: A breakdown of the appearance term in the motorbike model of Figure 5.10. (a) The 15 principal components of the PCA basis. (b) Histograms of the background (in black) and foreground coordinate (coloured) on each of the 15 principal direction (rows) for each of the 6 model parts (columns). Superimposed on the histograms is the Gaussian fitted in the learning procedure. The background histogram is produced by considering all regions found by the feature detector from the background dataset. The foreground histogram is produced by considering the regions assigned to the best hypothesis of each image in the training set. The number above each histogram gives the true positive detection rate (in %) at the point of equal false positive and false negative errors on ROC curve between the foreground and background histograms for the particular part/descriptor, so giving a measure of how discriminative it is. (c) The likelihood histograms of both the background data and the foreground data (as used in (b)) under the density for each model part. The title of each plot gives the true detection rate for each part (computed from the ROC curve at the point of equal error), showing which are the most discriminative.

5.2 Recognition

The recognition process is very similar in nature to learning. For query image, t, recognition proceeds by first detecting features, giving $\mathbf{X}^t$ and $\mathbf{S}^t$. These features are processed in same way as in learning, giving $\mathbf{D}^t$. This process is illustrated in Figure 5.12.

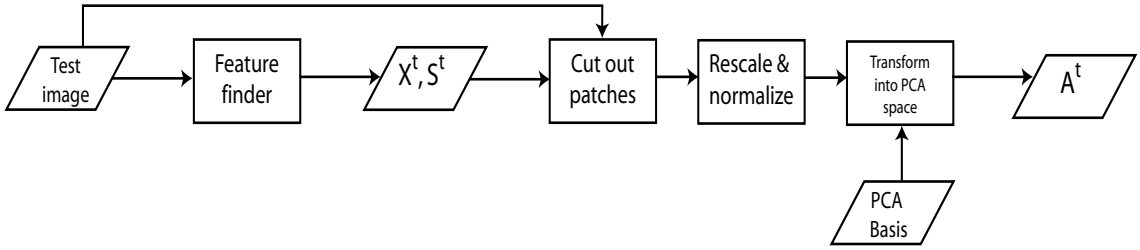


Figure 5.12: Pre-processing a query image

Once $\mathbf{X}^t$, $\mathbf{D}^t$ and $\mathbf{S}^t$ have been obtained we then compute the likelihood ratio using (4.2). To determine if an object is in the image, we should, according to (4.3) sum over all hypotheses. However, this has a drawback: background images typically contain many low-scoring hypothesis, which can occasionally combine to give a false alarm despite no indvidual hypothesis being sufficiently probable. Additionally, the hypotheses causing the false alarm may be widely spatially separated so the false detection has no clear position within the image. To avoid such issues, instead we select the best single hypothesis from the image. The likelihood ratio, assuming we take the ratio of the priors in (4.2) to be 1, is the same as the ratio of posteriors, R. This is then compared to a threshold to make the object present/absent decision. This threshold is determined from the training set to give the desired balance between false positives and false negatives.

If we wish to localize each instance of the object within the image, the best hypothesis is taken and a bounding box around it formed at its location. We then delete all features within the box. The next best hypothesis is then found and the process repeated until the best hypothesis falls below the threshold.

The same efficient search methods described in section 5.1.4 are used in the recognition process to find the single best hypothesis. However, recognition is faster as the covariances are tight (as compared with the initial values in the learning process) so the vast majority of hypotheses may safely be ignored. However the large N and P mean the process still takes around 2–3 seconds per image to perform the search, in addition to the 10 seconds/image needed

to extract the features.

5.3 Considerations for the star model

When using the star model the only major difference concerns the efficient search methods as used by the full model. The reduced dependency structure should mean that both learning and recognition take $O(N^2P)$ time, compared to $O(N^P)$ time of the full model (N being the number of regions per image and P the number of model parts). However, the ordering constraint used in Section 5.1.5 is required to help the model converge in learning to a sensible maximum. Without the constraint, due to the weaker shape model, the learning procedure fails to learn coherent spatial structure, as illustrated in Figure 5.14(a). Unfortunately, imposing the constraint means that learning reverts to an $O(N^P)$ time problem. This forces us to adopt efficient methods, similar to those used in Section 5.1.4 but which take advantage of the simplified shape model. Currently the use of ordering constraints in learning is a necessary evil but we consider alternative schemes in Chapter 10. In recognition no such ordering constraint is needed since the parameters are not being updated, only $O(N^2P)$ time is required.

5.3.1 Efficient methods for the star model

As in Section 5.1.4 we restrict ourselves to computing the small subset of hypotheses which have high probability, enabling us to obtain an accurate approximation to the true value of the sufficient statistics computed in the E-step. The key difference between doing this for the star model, compared with the full model, is that computing the likelihood of a hypotheses is quick (since we can pre-compute all N^2P terms needed to form all possible hypotheses) thus all the search procedure needs to do is to check if a given hypothesis satisfies the ordering constraint or not. The scheme uses a threshold on the maximum allowable difference between the best hypothesis and the worst one considered (δ_h). Given that the landmark part must always be observed, we want it to model a commonly occurring part of the object irrespective of its location. Consequently, we do not apply the ordering constraint to the landmark part. The scheme proceeds as follows:

1. Pre-compute all possible ($N^2(P-1)$) landmark-part combinations, taking part 1 as the landmark part.

2. Compute the best possible hypothesis, ignoring the ordering constraint (h^*). This acts as an upper bound for the best hypothesis incorporating the ordering constraint.

3. For each detection assigned to the landmark part 1, compute the best possible assignment for each non-landmark part ($h_p^*(h_1)$ for $p \neq 1$). This acts as an upper-bound for any unassigned parts in our search.

4. Form a search tree as illustrated in Figure 5.4. At each node in the tree, those child nodes violating the ordering constraint may be pruned. Additionally, if the cumulative probability of the parts assigned at a given node plus the upper bounds ($h_p^*(h_1)$) fall below $h^* - \delta_h$ then the node may be pruned. The search tree is implemented using a single recursive function which does not explicitly store the nodes, so uses very little memory. This is in contrast to the A^* search for the full model which must store nodes within the tree (since the hypothesis probabilities cannot be pre-computed), resulting in a large memory overhead.

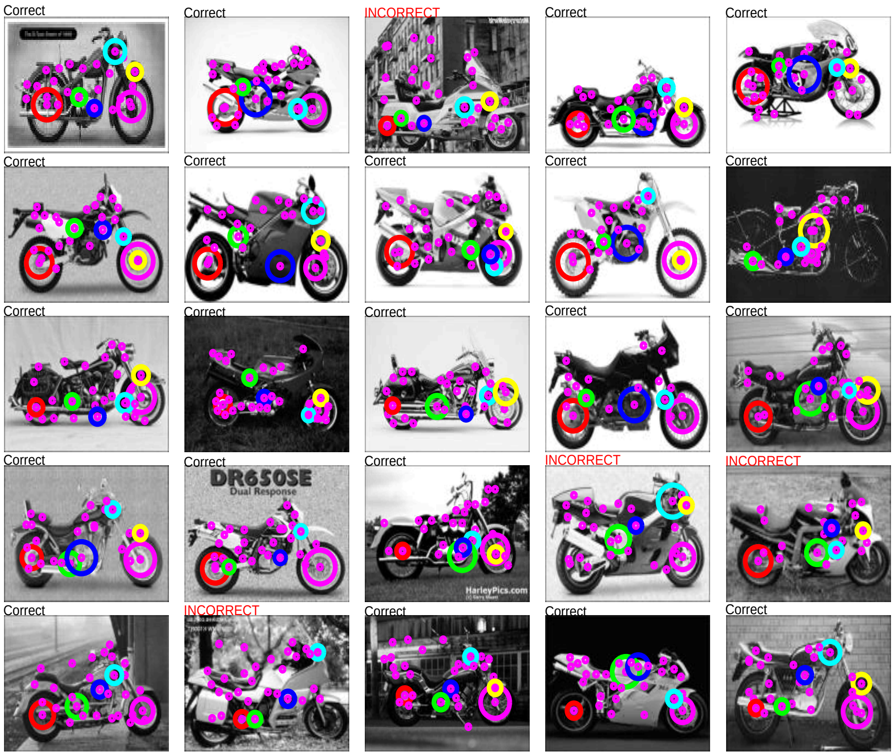
5. The output is a set of hypotheses, all having a probability in the range h^* to $h^* - \delta_h$.

6. It is possible that no hypotheses are returned, since the best hypothesis incorporating the ordering constraint may be less than $h^* - \delta_h$. In this case δ_h is increased and the search repeated.

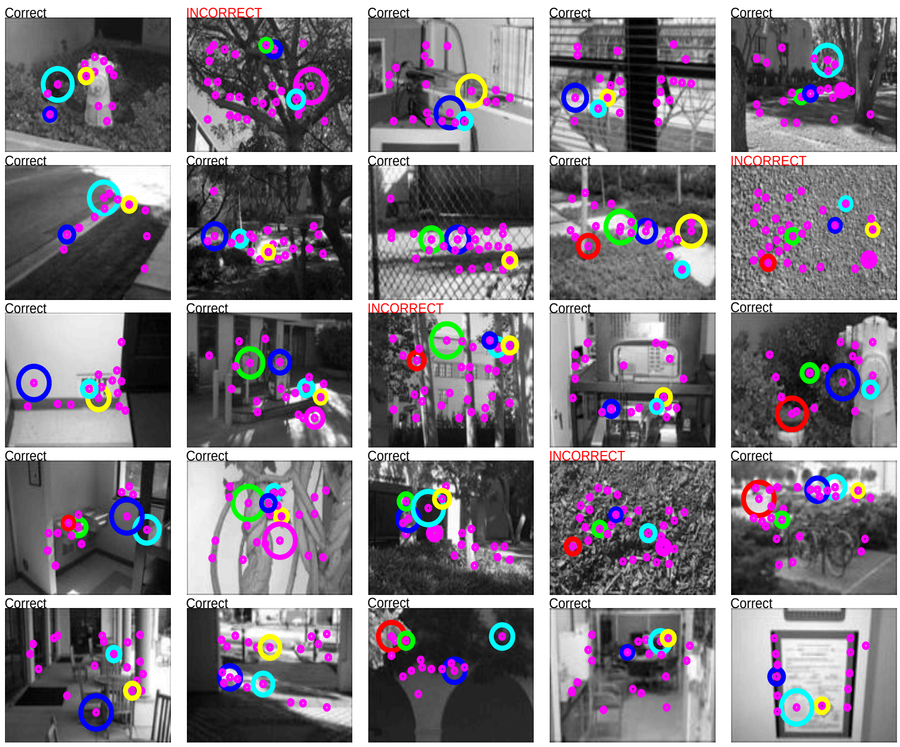
The heavy pruning of the tree and low memory requirements, enable a larger N and P to be accommodated than with the full model, as shown in Figure 5.15. In Figure 5.15(a) P is fixed at 6 and N varied from 20 up to 200 while recording the mean time per image over all EM iterations in learning. The curve has a quadratic shape with the time per image still respectable even for $N = 200$. It should be noted that a full model cannot be learnt with $N > 25$ due its memory requirements. In Figure 5.15(b) N is fixed at 20 and P varied from $2 - 13$ with the mean time per image plotted on a log-scale y-axis. The curve for the full model is a straight line as expected from the $O(N^P)$ complexity, stopping at $P = 7$ owing to the memory overhead. The star model's curve, while also roughly linear, has a much flatter gradient: a 12 part star model taking the same time to learn as a 6 part full model.

For a 6 part model with 20 detections per image the star model typically takes 10-15 minutes to converge, as opposed to the 24 hours of the fully connected model — roughly the same time

as a star model with 12 parts and 100 detections per image would take (timings are for a 2Ghz Pentium 4).



(a)

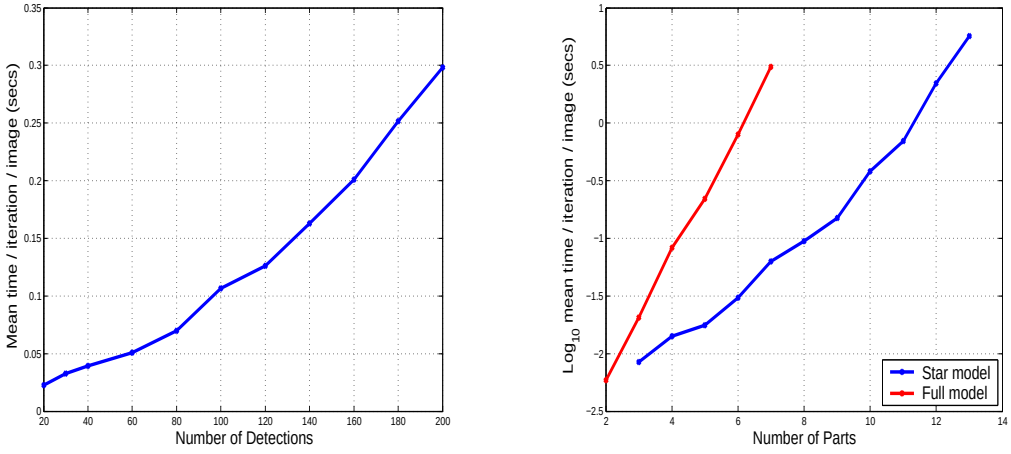


(b)

Figure 5.13: (a) The motorbike model from Fig. 5.10 evaluating a set of query images. The pink dots are features found on each image and the coloured circles indicate the features of the best hypothesis in the image. The size of the circles indicates the scale of feature. The outcome of the classification is marked above each image, incorrect classifications being highlighted in red. (b) The model evaluating images from the negative test set (scenes around Caltech).

(a) (b)

Figure 5.14: (a) The star-structured shape model for motorbikes, learnt without an ordering constraint. (b) The star-structured shape model for motorbikes, learnt using an ordering constraint. No coherent structure is found in (a), in contrast to the model in (b). In (b) note that the landmark, indicated by the red ellipse, is not subject to the ordering constraint, thus it is located in the middle of the object.



(a) (b)

Figure 5.15: Plots showing the learning time for a star model with different numbers of parts (P) and detections per image (N). (a) P fixed to 6 and N varied from 20 to 200. The curve has a quadratic shape, with a reasonable time even for $N = 200$. (b) N fixed to 20 and P varied from 2 to 13, with a logarithmic time axis. The full model is shown in red and the star model in blue. While both show roughly linear behaviour, the star model's curve is much flatter than the full model: a 12 part star model can be learnt in the same time as a 6 part full model.

Chapter 6

Weakly supervised experiments with the constellation model

Having explained the structure and operation of the Constellation Model, we now evaluate its performance on a variety of categories from the Caltech datasets. Both the full spatial model and the star model are evaluated and compared. We also perform experiments to understand how the model works.

Each of the Caltech datasets contain images representing a different category. Since the model only handles a single viewpoint, the datasets consist of images that are mirror-reversed, if necessary, so that all instances face in a similar direction, although there is still variability in location and scale within the images. Additional experiments are performed to elucidate the properties of our model and include baseline methods.

For each experiment, the dataset is split randomly into two separate sets of equal size. The model is then trained on the first and tested on the second. In recognition, the models are tested in both classification and localization roles.

In classification, where the task is to determine the presence or absence of the object within the image, the performance is evaluated by plotting receiver-operating characteristic (ROC) curves. To ease comparisons we use a single point on the curve to summarize its performance, namely the point of equal error (i.e. p(True positive)=1-p(False positive)) when testing against one of two background datasets. For example a figure of 9% means that 91% of the foreground images are correctly classified and 9% of the background images were incorrectly classified

(i.e. thought to be foreground). While the number of foreground test images varied between experiments, depending on the number of images available for each category, the foreground and background sets are always of equal size.

In localization, where the task is to place a bounding box around the object within the image, the performance is evaluated using recall-precision curves (RPC), since the concept of a true negative is less clear in this application. A recall-precision curve shows retrieval performance. Detections output by the algorithm are ordered by their likelihoods and the precision is plotted against recall as a likelihood threshold is lowered. Recall is defined as the number of true positives (the correct localizations) over the total positives in the data set (total number of object instances), and precision is the number of true positives over the sum of false positives (incorrect localizations) and true positives.

To be considered a correct detection, the area of overlap, a_o between the estimated bounding box B and the ground truth bounding box, B_{gt} must exceed 0.5 according to the criterion:

$$a_o = \frac{\text{area}(B \cap B_{gt})}{\text{area}(B \cup B_{gt})} \tag{6.1}$$

A demonstration of this algorithm is given in the Figure 6.1. When evaluating the UIUC dataset, we adopt the same criterion on bounding box overlap and relative centroid location as used in Agarwal and Roth [1].

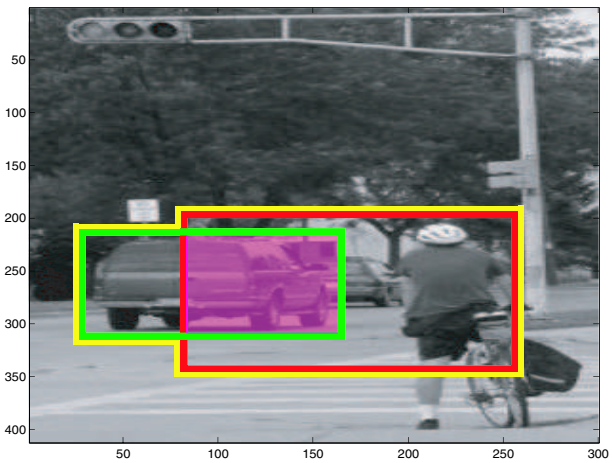


Figure 6.1: An illustration of the localization criterion of (6.1). The estimated bounding box B is shown in red. The ground truth bounding box B_{gt} is shown in green. area$(B \cap B_{gt})$ is the magenta tinted region, while area$(B \cup B_{gt})$ is outlined in yellow.

6.1 Full model experiments

The first set of experiments use the full shape model with Kadir and Brady features, as introduced in Section 2.3.1. Identical software and settings are used for all categories (with the exception of the UIUC dataset, due to the small image sizes). Images were resampled to 300 pixels in width by bilinear interpolation, regions extracted at scales between 10 and 30 pixels in radius. The gradient-based PCA representation is used with $a = 15$, with the two additional energy and residual terms bring the number of descriptors up to 17 per feature. A patch size of $k = 21$ is used. In learning, we use $P = 6$ and $N = 20$. N was increased to 30 in recognition. These values will subsequently be referred to as the "standard model settings".

Figures 6.2-6.4 show models and test images with a mix of correct and incorrect classifications for four of the datasets. Notice how each model captures a simple description, be it in appearance or shape or both, of the object. The face and motorbike datasets have tight shape models, but some of the parts have a highly variable appearance. For these parts any feature in that location will do regardless of what it looks like (hence the probability of detection is 1). Conversely, the leopard dataset has a loose shape model, but a highly distinctive appearance for each patch. In this instance, the model is just looking for patches of spotty fur, regardless of their location. The differing nature of these examples illustrate the flexible nature of the model.

The majority of errors are a result of the object receiving insufficient coverage from the feature detector. This happens for a number of reasons. First, when a threshold is imposed on N (for the sake of speed), many features on the object are removed. Second, the feature detector seems to perform badly when the object is much darker than the background (see examples in Figure 6.2). Finally, in the Kadir & Brady detector, the regions are produced by a clustering procedure which is somewhat temperamental and can result in parts of the object being missed.

Table 6.1 shows the model being tested on five Caltech dataset in a classification role. The results show a fair degree of consistency over the datasets, all being under 12.3%. The error margin on the values are $+/-1\%$ or so. Comparing column (b) to (a), the scale-invariant model always outperforms the fixed-scale model, most notably in case of cars (rear). Column (c) shows that changing the negative test examples gives a moderate increase in the error rate.

Table 6.2 shows a confusion table between the different categories, using the models evalu-

Dataset	Total size of dataset	(a)	(b)	(c)
Motorbikes	800	3.3	3.3	6.0
Faces	435	10.6	8.3	10.1
Airplanes	800	6.7	6.3	6.5
Leopards	200	12.0	11.0	11.0
Cars (Rear)	800	12.3	9.2	9.3

Table 6.1: Classification results on five Caltech datasets. (a) is the error rate(%) at the point of equal-error on an ROC curve for a fixed-scale model, testing against the Caltech background dataset (with the exception of Cars (rear) which uses empty road scenes as the background test set). (b) is the same as (a) except that a scale-invariant model is used. (c) is the same as (b), except that the Google background dataset was used in testing.

ated in Table 6.1. Despite being inherently generative, the models can distinguish between the categories well. The cars rear model seems to be somewhat weak, with many car images being claimed by the airplane model. Examination of Figures 6.4 and 6.5 reveals that the two model indeed look very similar.

		Recognised category			
True class	A	C	F	S	M
(A)irplane	88.8	6.0	0.3	0.7	4.2
(C)ars (Rear)	19.7	67.0	0.8	3.3	9.2
(F)ace	2.8	1.4	86.2	2.3	7.3
(S)potted Cats	3.0	1.0	3.0	76.0	17.0
(M)otorbike	1.3	0.0	0.0	1.0	97.7

Table 6.2: Confusion table between the four categories. Each row gives a breakdown of how a query image of a given category is classified (in %). No clutter dataset was used, rather images belonging to each category acted as negative examples for models trained for the other categories. The optimum would be 100% on the diagonal with zero elsewhere.

Detection performance on the five datasets are shown in Figure 6.6. The predicted bounding box was produced by taking the bounding box around the regions of the best hypothesis and then enlarging it by 40% since the model representation tends to miss some parts of the object (e.g. the tail of the airplanes). It was assumed that a single instance was present in each image. Recall precision curves are plotted for the fixed scale model; the scale-invariant model and a crude baseline, using the criterion in (6.1).

The baseline localization algorithm outputs a bounding box that is the same size as the test image, which a score that is equal to the likelihood ratio given by the scale-invariant Constellation Model on the same image. Hence the baseline gives an indication of the total area the objects occupy within the dataset. For example, the motorbikes tend to fill most of

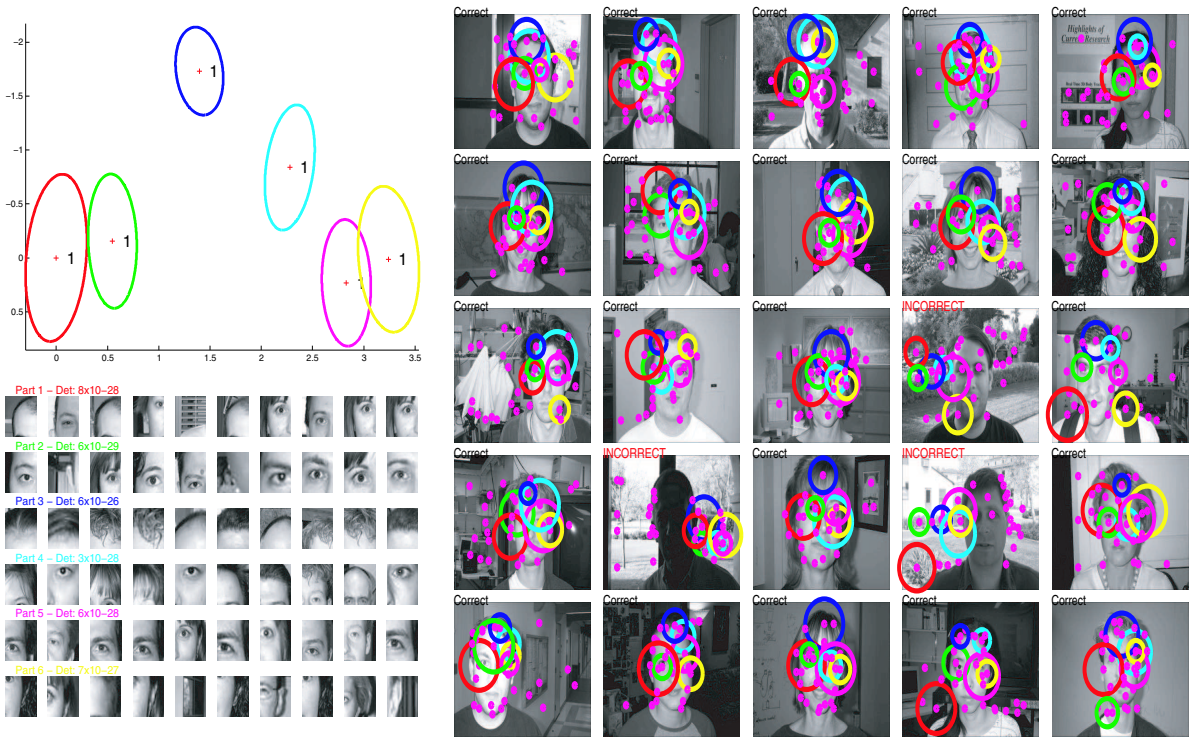


Figure 6.2: Left: A typical face model with 6 parts. Right: A mix of correct and incorrect detections.

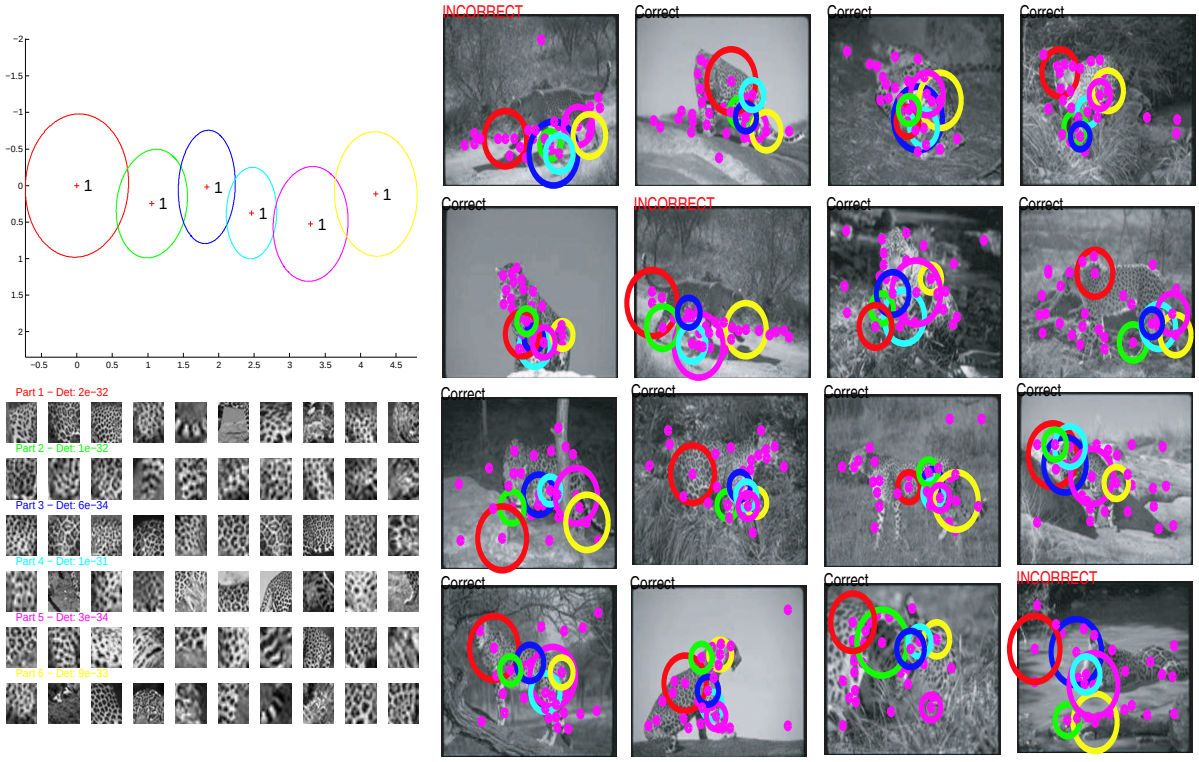


Figure 6.3: A typical leopard model with 6 parts. Note the loose shape model but distinctive "spotted fur" appearance.

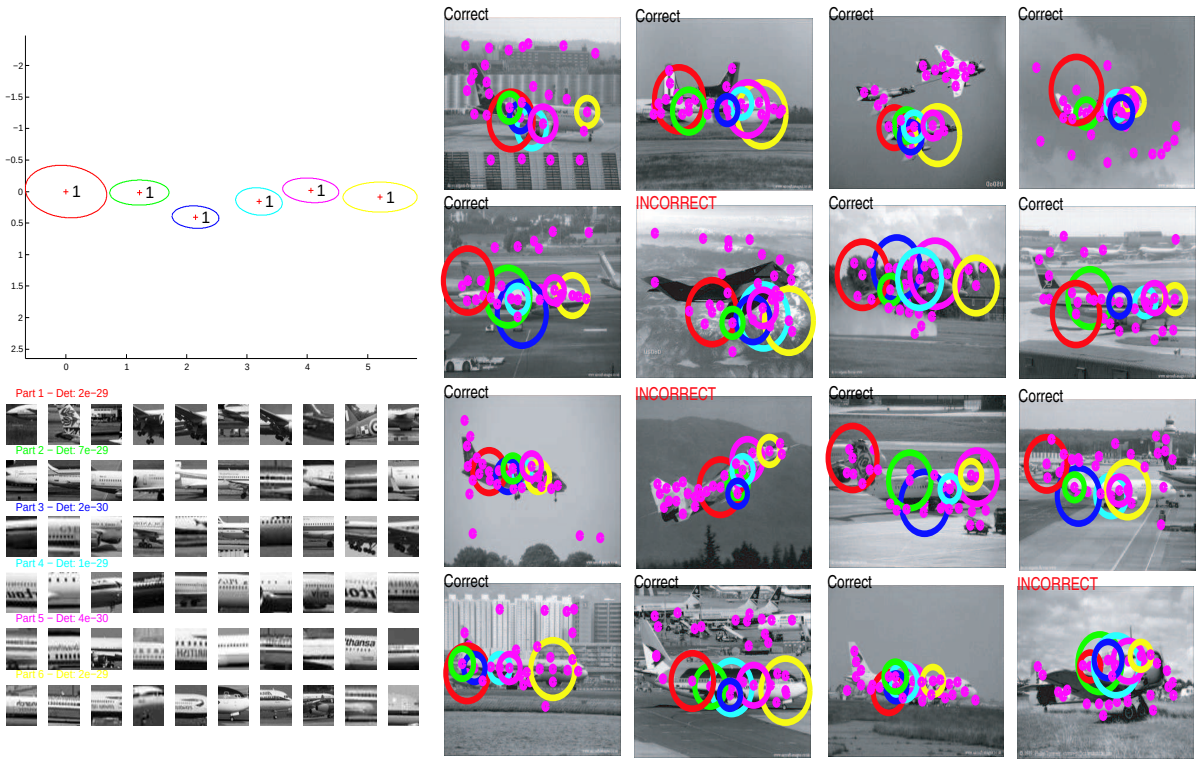


Figure 6.4: A typical airplane model with 6 parts. The long horizontal structure of the fuselage is captured by the shape model.

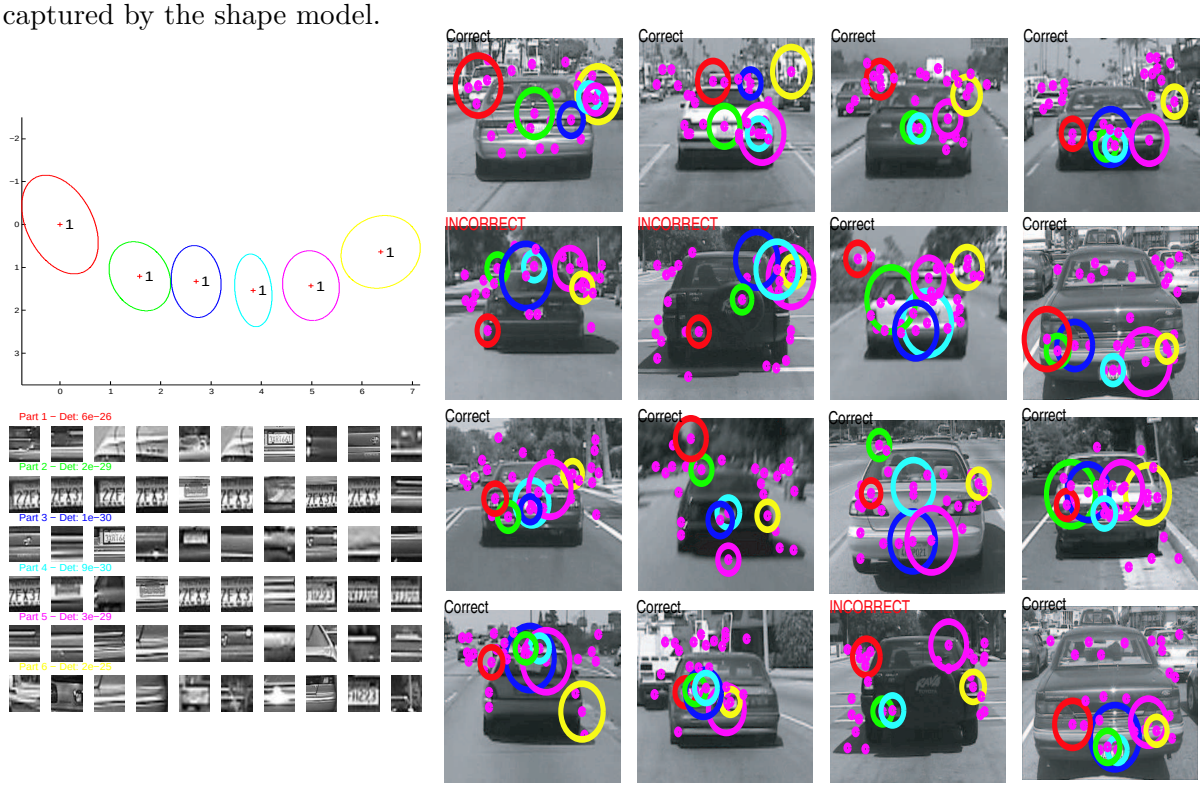
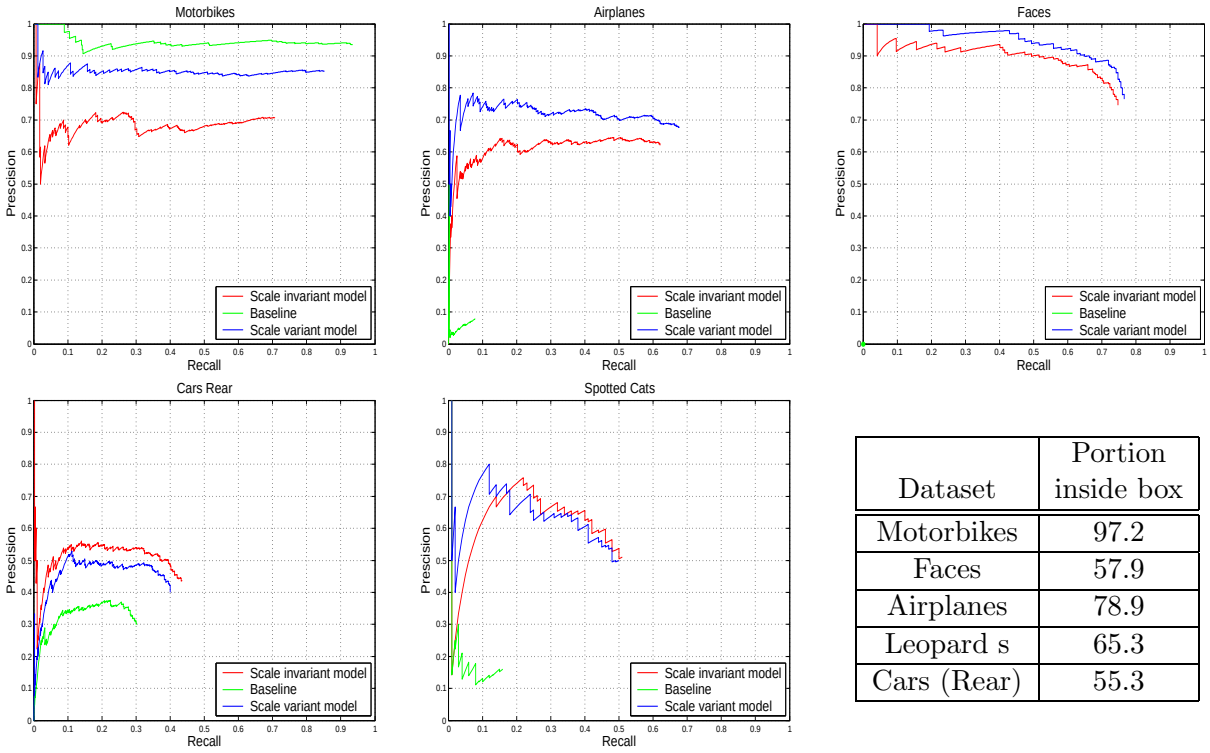


Figure 6.5: A 6 part Cars (Rear) model. The model consists of low-level horizontal line type structures on the back of the car, along with the license plate.



Dataset	Portion inside box
Motorbikes	97.2
Faces	57.9
Airplanes	78.9
Leopard s	65.3
Cars (Rear)	55.3

Figure 6.6: Recall-precision curves evaluating the localization ability of the models for each of the five datasets. The fixed scale model is shown in blue; the scale-invariant model in red and a crude baseline in green. With the exception of cars rear, the fixed scale model preforms better than the scale-invariant probably because the scale of the objects in the training and testing set is fairly constant for most categories. The table lists the portion of regions lying within the ground truth bounding box of the object for each dataset. This is a crude measure for the difficulty of each category.

the image so the baseline beats two model variants but performs badly on the other categories. Another baseline localization measure is given in Figure 6.6(f) where we specify the portion of regions, by number, inside the ground truth bounding box of the object (using the centroid of the region). It is interesting to note that the fixed-scale models beat the scale-invariant models since in many of the datasets the scale variation is not that large and the scale-invariant model is making predictions within a larger space. The exception to this is the cars (rear) dataset, where the scale variation is larger and the scale-invariant model outperforms the scale variant one.

Using the cars (side) dataset, the ability of the model to perform multiple-instance localization is tested. The training images were mirror-reversed, if necessary so that all cars would face in the same direction, but the test examples contained cars of both orientations. Given the approximate symmetry of the object, the right-facing model had little problem picking out

left-facing test examples. The model and test examples are shown in Figure 6.7, while the recall precision curve comparing to Agarwal and Roth [1] is shown in Figure 6.4(a).

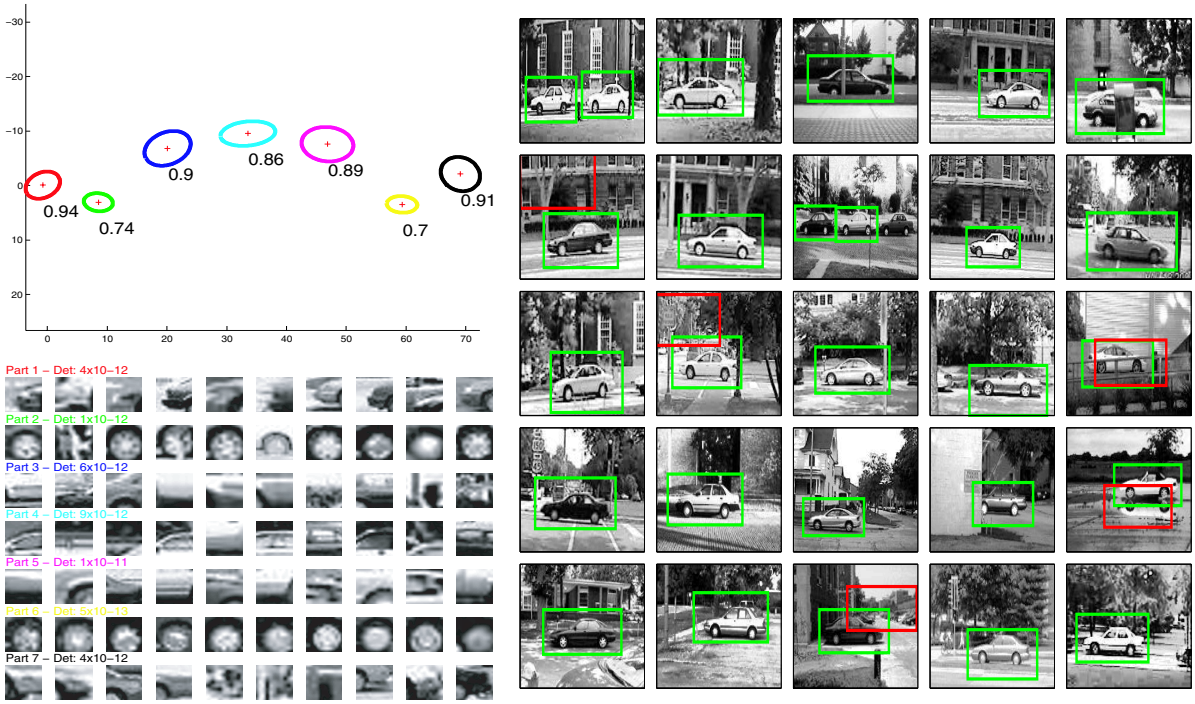


Figure 6.7: The UIUC Cars (Side) dataset. The task here is to localize the object instance(s) within the image. On the left, we show the 7 part model. On the right, examples are shown from the test set, with correct localizations highlighted in green, while false alarms are shown in red.

6.1.1 Baseline experiments

To put the performance of the model in context, we apply a variety of baseline methods to the datasets:

- **Orientation histograms**: The gradients of the whole image are computed and thresholded to remove areas of very low gradient magnitude. An 8 bin histogram is computed of the orientations within the gradient image, with weighting according to the magnitude of the gradient. Each image is thus represented as by an 8-vector. The classifier consists of a single Gaussian density for both the foreground and background class, modelling the mean and variance of the 8 histogram bins. The parameters of these 8 dimensional densities are estimated from the training data. A query image is evaluated by computing the likelihood ratio of the images' 8-vector under the foreground and background Gaussian

models.

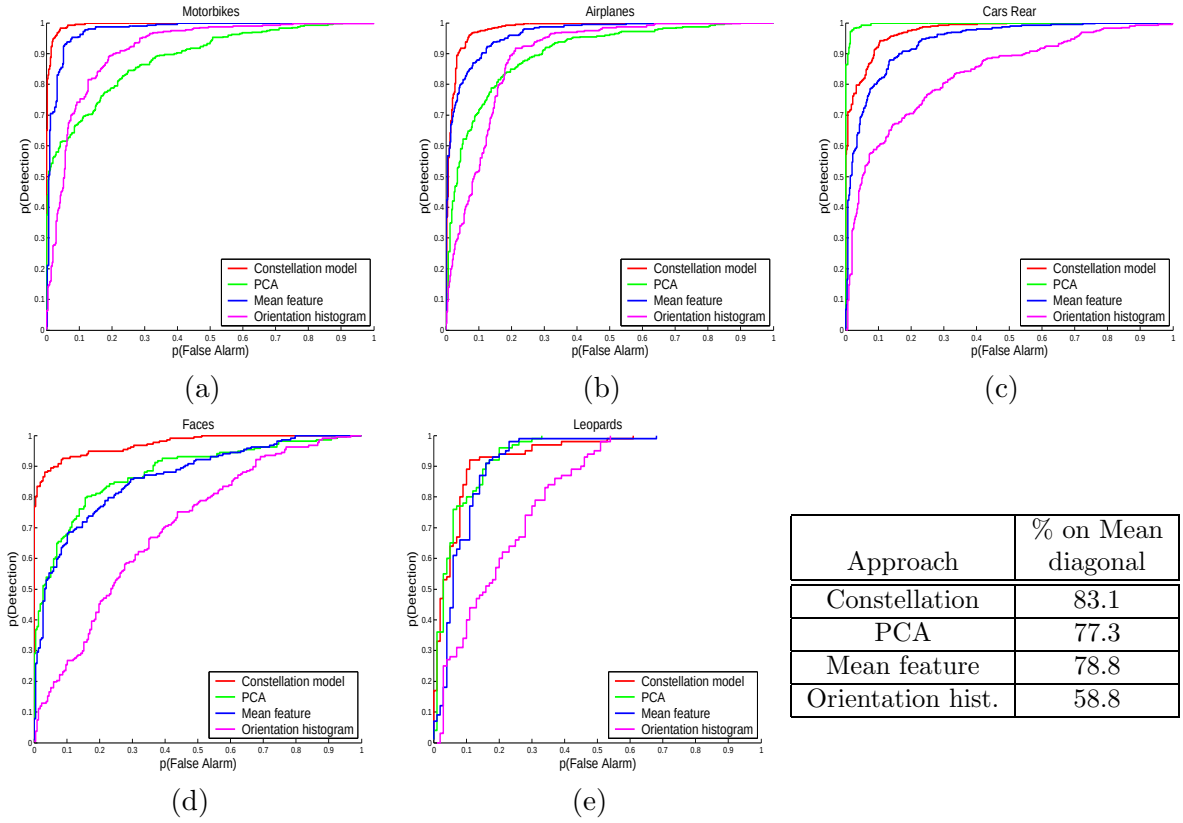
- **Mean feature**: The arithmetic mean of **D** (the appearance of all features in each image) is computed, giving a 17-vector for each image. The classifier consists of a single Gaussian density both for the foreground and background class. The parameters of these 17 dimensional densities are estimated from the training data. A query image is evaluated by computing the likelihood ratio of the images' 17-vector under the foreground and background Gaussian models. This baseline method is designed to reveal the discriminative power of feature detector itself.

- **PCA**: Each image is resized to a 21x21 patch. After appropriate normalization, it is then projected into a 25 dimensional PCA basis (precomputed and the same for all categories). The classifier consists of a single Gaussian density for both the foreground and background class, modelling the coefficients of each basis vector. The parameters of these 25 dimensional densities are estimated from the training data. A query image is evaluated by computing the likelihood ratio of the images' 25-vector under the foreground and background Gaussian models. The baseline method is likely to perform well if the object pose is highly constrained.

In Figures 6.8(a)-(e), we show ROC curves for the three baseline methods and the constellation model on the five datasets in a classification task. Two of the baseline methods perform reasonably well. However, the constellation model is only beaten in one case - cars (rear) - by the PCA baseline. In Figure 6.8(f) we compare the performance in a multi-class setting by giving the mean diagonal of a confusion table over the five categories for each of the approaches, with the constellation model showing superior performance. It should be emphasized that the constellation model allows localization of each detected object and of its parts, while the baseline methods do not.

6.2 Analysis of performance

We look at variations from the standard model settings, such as altering the number of model parts; contaminating the training data, and removing terms from the model.



Approach	% on Mean diagonal
Constellation	83.1
PCA	77.3
Mean feature	78.8
Orientation hist.	58.8

Figure 6.8: ROC curves of the baseline experiments on the five datasets (a)–(e). The red curve is the constellation model while the green, blue and magenta curves are PCA; the mean feature and orientation histogram baselines respectively. The constellation model beats the baselines in all cases except for the cars rear dataset. (f) The table gives the mean diagonal of a confusion table across all five datasets for the 4 different approaches.

6.2.1 Changing scale of features

If the scale range of the saliency operator is changed, the set of regions extracted also changes, resulting in a different model being learnt. Figure 6.9 shows the effects of altering the feature scale on the face dataset. Figures 6.9(a) & (b) show a model using regions extracted using a hand selected scale range of 5-12 pixels in radius. The model picks out the eyes as well as the hairline. The standard scale range of 10-30 pixels in radius results in a model shown in Figures 6.9(c) & (d). The eyes are no longer picked up by the model, which relies entirely on the hairline. The models have similar error rates: 8.3% for the generic scale setting and 10.1% for the hand-tuned one.

This illustrates that while the output of the feature detector varies depending on the settings used, the ability of the learning algorithm to find consistency within whatever feature set it is

given makes the algorithm as a whole less sensitive to the performance of the feature detector.

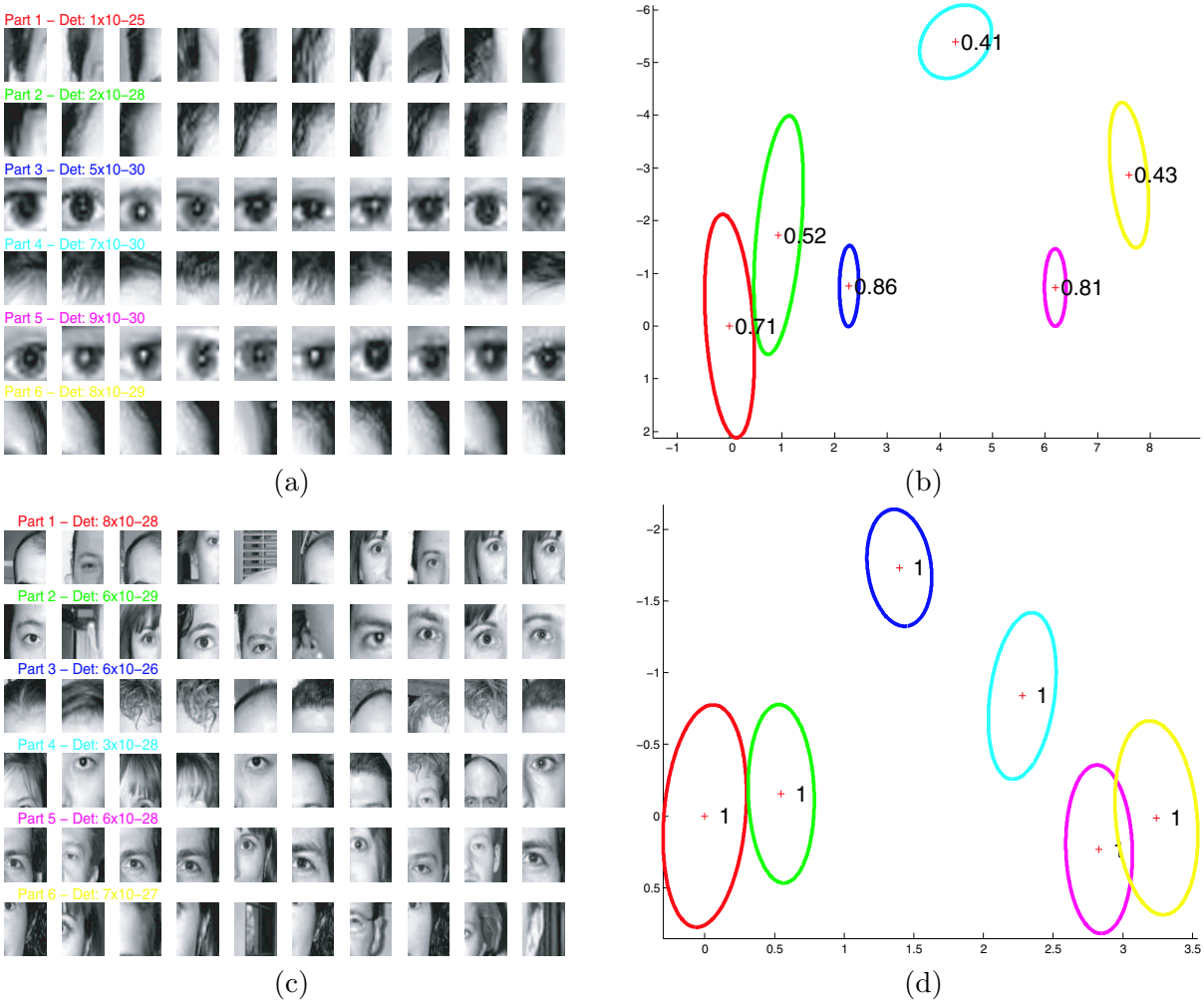


Figure 6.9: (a) & (b): A face model trained on images where the feature scale has been hand-tuned. The model captures both the hairline and eyes of the face. (c) & (d): A face model trained on regions extracted using a generic feature scale (as used in the experiments of Section 6.1). The de-tuned feature detector no longer picks out distinctive features such as the eyes. Comparing the determinants of the parts in each model reveal that the tuned model has tighter appearance densities.

6.2.2 Feature Representation

We now investigate different methods of representing the appearance of the regions within an image. In Section 6.1 we used a 15 dimensional gradient-based PCA approach with two additional dimensions capturing (i) the energy of the gradients in the regions and (ii) the residual between the original region and the point in PCA space. The 2nd column of Table 6.3 shows the classification error rates for the five datasets. If the two extra dimensions are removed from the representation, then the error rates increase slightly, as seen in the 3rd column

of Table 6.3. Fergus *et al.* [40] used a 15 dimensional intensity based PCA representation. Its performance is compared to the two gradient based PCA approaches in the 4th column of Table 6.3. Here a generic PCA basis used for all categories, as opposed to the per-category PCA basis of [40].

Dataset	Gradient PCA 15+2	Gradient PCA 15	Intensity PCA 15
Motorbikes	3.3	7.5	9.5
Faces	8.3	13.3	10.1
Airplanes	6.3	8.0	6.8
Leopards	11.0	11.0	13.5
Cars (Rear)	8.8	11.5	7.8

Table 6.3: Classification results on five datasets for three different representations.

6.2.3 Number of parts in model

The number of parts within the model must be chosen beforehand and has an exponential effect on the learning time, despite the use of efficient search methods. Clearly, more parts gives more coverage of the object, but it makes the model slower to learn and introduces over-fitting problems, due to the increased numbers of parameters. Figure 6.10 shows that there is little improvement between $P = 6$ and $P = 7$. If it were possible to investigate beyond $P = 7$, no doubt the error rate would start to increase due to over-fitting.

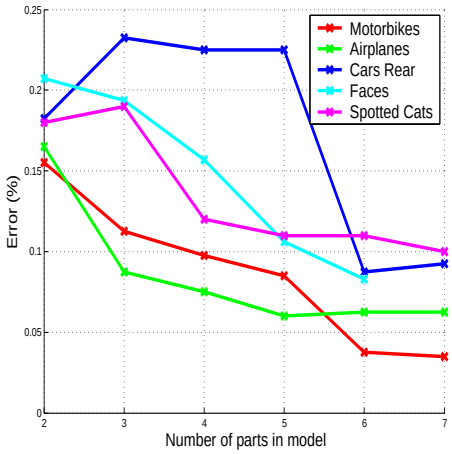


Figure 6.10: The effect of the number of parts, P, in the model versus error rate for the five datasets for $N = 20$ regions/image.

6.2.4 Contribution of the different model terms

Figure 6.11 shows contribution of the shape and appearance terms to the performance of the model. The figure shows ROC curves evaluating classification performance and RPC evaluating localization on the six datasets. A single model was learnt for each category, using a complete model with both shape and appearance, while in recognition, three different forms of the model were used: (i) the complete model, (ii) the complete model, but ignoring the appearance term and (iii) the complete model, but ignoring the shape term. Across all six datasets, it is noticeable that the appearance term is more important than the shape for classification. Only in the case of airplanes did the shape-only model perform well. Indeed, for cars (rear) the removal of the shape term actually improves the performance slightly. This is due to the ordering constraint imposed on the shape which in this case may be removing the best hypothesis from each image. However, when the task is localization, the shape-only model outperforms the appearance only one. In some cases the difference is dramatic, e.g. airplanes and cars (side).

When learning was performed with degraded models, the use of the appearance component alone produced a model with a performance close to that of the full model. However, when using only the shape component, the models frequently did not converge to a meaningful model, resulting in chance level classification performance.

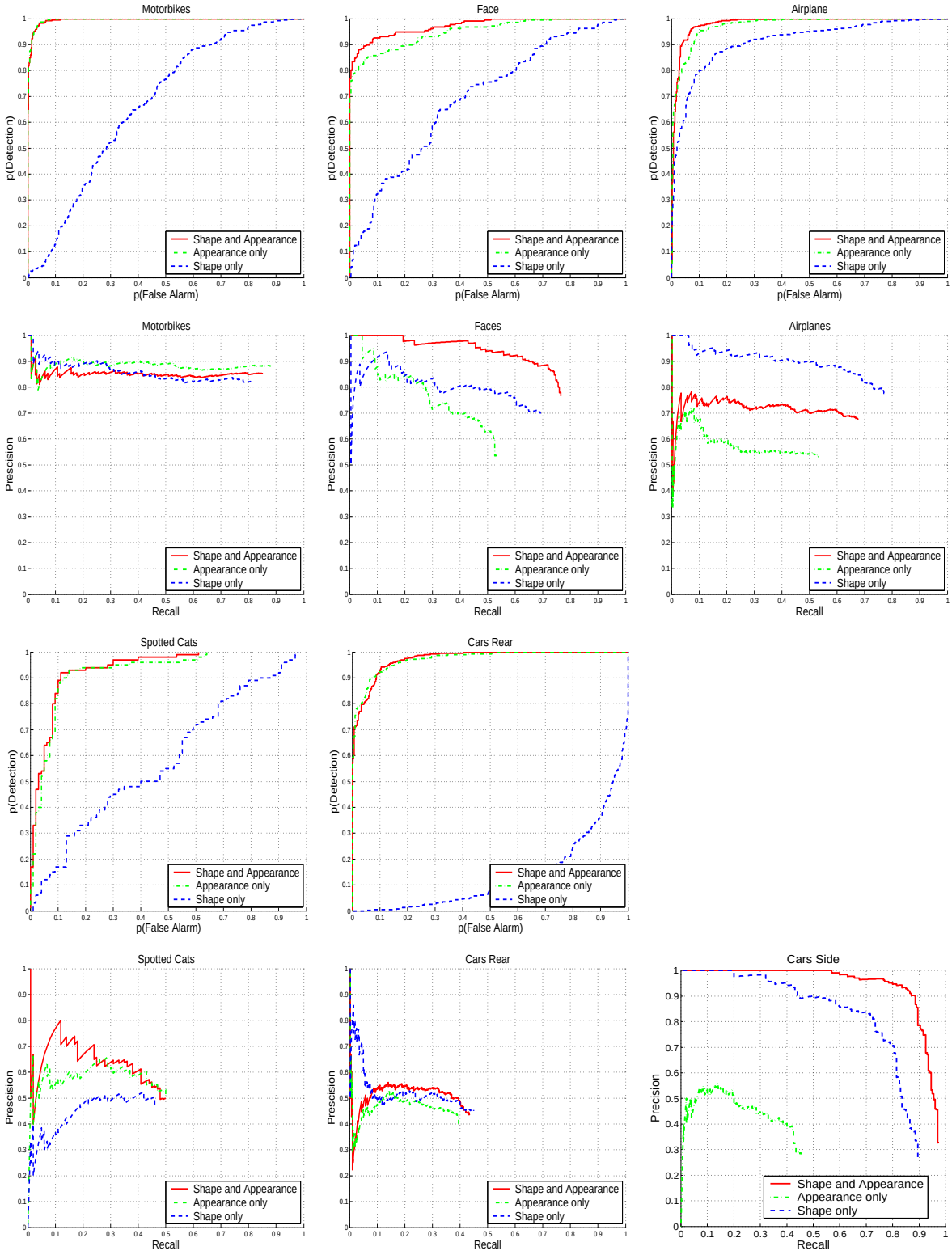


Figure 6.11: ROC and RPC curves for 6 different datasets. The effects of removing the shape or appearance terms are shown. The models rely heavily on appearance in a classification task (1st and 3rd rows). In detection, by contrast, the shape is more important than appearance (2nd and 4th rows).

6.2.5 Over-fitting

The large number of parameters in the model, as shown in table 5.1, means that a large number of images are needed in training to avoid over-fitting the training data. Figure 6.12 shows that around 250 training images are needed for the model to generalize well. The use of priors in learning can dramatically reduce the training requirement on the number of images, down to just a handful or even one. See Fei-Fei *et al.* [32] for a Bayesian extension to the constellation model, incorporating such priors in learning.

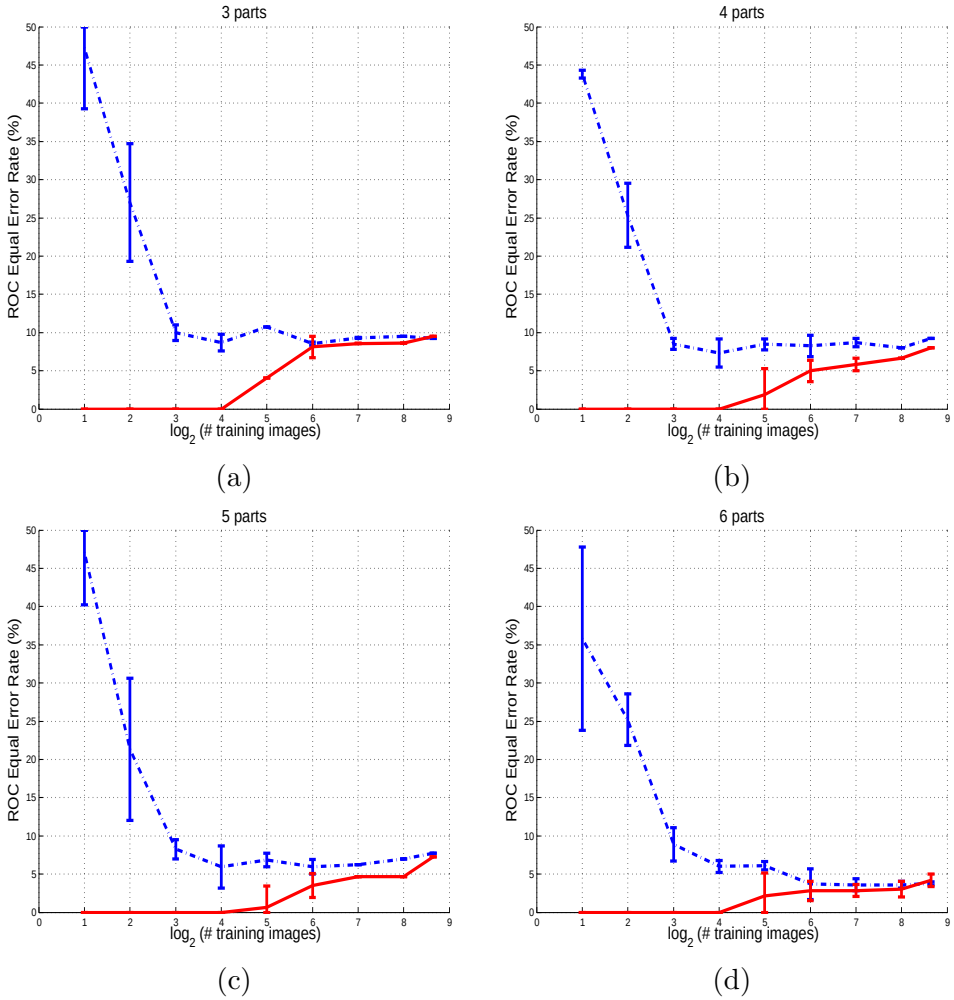


Figure 6.12: (a)-(d) The training (red) and test (blue dot-dashed) errors versus the number of training images for the motorbike dataset with models having (a) 3 parts, (b) 4 parts, (c) 5 parts, (d) 6 parts. Note the log scale on the x-axis. The curves converge to the Bayes' error once the number of training images is 256. The error bars show two standard deviations on performance, computed from 10 runs for 2–64 training images and 5 runs for 128–400 images.

6.2.6 Contamination of the training set

Collecting hundreds of images of a given category of sufficient quality is a laborious time-consuming job. Being able to learn from datasets where some of the images consist only of clutter or are of insufficient quality is a useful property since it simplifies the task of building a training set. In Figure 6.13 we show the results of deliberately introducing background images (e.g. images not containing an object belonging to the class that is being learnt) into a training set. For most of the datasets the drop in performance is small even with 50% background images in the training set. At this level of contamination, the models are typically are loose: they have appearance densities with a large variance and each part has a probability 1 of being observed. Notice that the graph bottoms out at around 20%–30% correct when the training set is entirely background images, while one would have expected 50% correct (i.e. chance performance). This implies that the background images contain some structure that is being learnt by the model, implying that our assumptions about the nature of the background are not strictly true (see Section 4.9.4). In [38], the problem of learning from contaminated data is investigated in more depth.

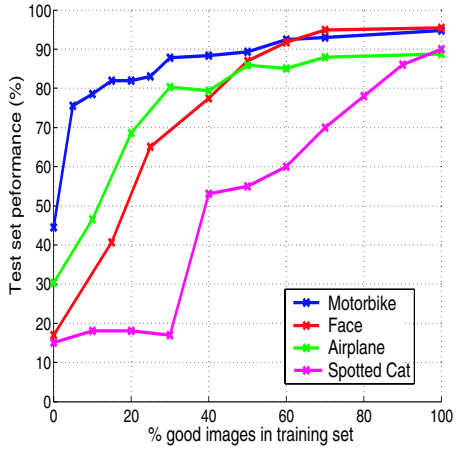
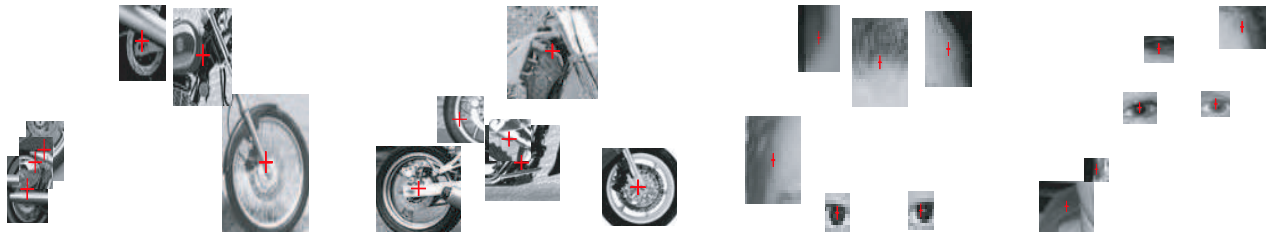


Figure 6.13: The effect of mixing background images into the training data, for 4 of the different datasets. With the exception of the leopard dataset, even with a 50-50 mix of images with/without objects, the resulting model error has only increased by a small margin, compared to training on uncontaminated data.

6.2.7 Sampling from the model

Since our models are generative in nature we can sample from them. However, we are not able to directly sample back into pixel-space since the use of PCA for appearance introduces a non-probabilistic stumbling block. While we are able to draw samples within the PCA-space, there exists a many-to-one mapping from patch-space into the PCA-space. Additionally, the normalization of the patch introduces a similar many-to-one mapping problem. However, we can use one of two approximations: (i) Draw a sample from the appearance density and find the closest data point from all images and use its corresponding pixel patch or (ii) Form the patch directly from it's coordinates in the k dimensional PCA-space, assuming that all $121 - k$ coefficients are zero. Note that this will give a patch that is still normalized. In Figure 6.14 we show samples drawn using method (i) above.



(a) (b) (c) (d)

Figure 6.14: (a) & (b) Two samples drawn from a motorbike model. (c) & (d) Two samples drawn from face models.

6.3 Comparison with other methods

We now compare the Constellation Model to a variety of different approaches in both classification and detection tasks. In Table 6.4 we show results on the same datasets (with identical training and testing images) for the following algorithms: an earlier incarnation of the constellation model by Weber *et al.* [109, 108] (which lacked a probabilistic appearance model and also learnt the parts' appearance separate from their location); the region-based discriminative SVM-based method of Opelt *et al.* [84]; the bag-of-words pLSA approach of Sivic *et al.* [100] and the Hough space voting scheme of Leibe *et al.* [65]. The table also lists the supervision required in training. The performance of the model can be seen to be comparable to the other approaches, beating the other methods in the majority of cases. The method of Leibe *et al.*

[65] achieves better performance; however, it requires the manual segmentation of the objects within the training images and has only been applied to a subset of the categories.

Figure 6.15 shows a recall-precision curve comparing the algorithm to Agarwal and Roth [1], with our algorithm showing a distinct improvement. The superior performance of Leibe *et al.* [65] on this dataset may be explained by the manual segmentation of the training images and the use of a validation scheme in their approach, which helps to eliminate false positives.

Method	CM	Weber *et al.* [109, 108]	Opelt *et al.* [84]	Sivic *et al.* [100]	Leibe *et al.* [65]
Supervision	I	I	I	N	I,S
Motorbikes	3.3	16.0	7.8	15.4	6.0
Faces	8.3	6.0	6.5	5.3	-
Airplanes	6.3	32.0	11.1	3.4	-
Leopards	11.0	-	-	-	-
Cars Rear	8.8	-	8.9	21.4	6.1
Cars Side	11.5	-	17.0	-	3.0

Table 6.4: Comparison of the Constellation Model (CM) to other methods [1, 65, 84, 100, 108, 109]. The table gives ROC equal error rates (except for the car (side) dataset where it is a recall-precision equal error) on a number of datasets. The second row shows the supervision required in training for each method: N - None; I - Image labels and flipping to give consistent viewpoint; S - Object segmentation.

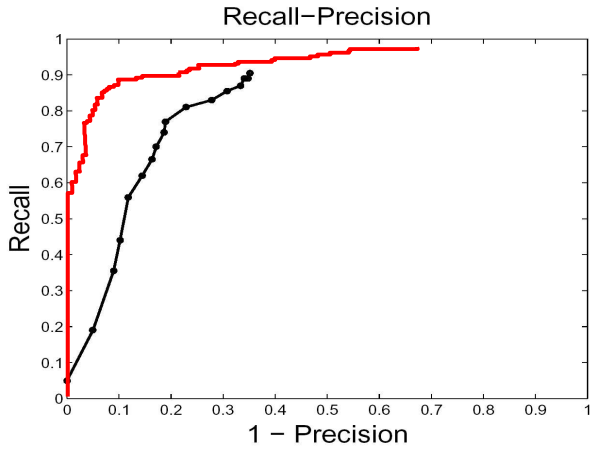


Figure 6.15: The recall-precision curve comparing Agarwal and Roth [1] (in black) and the Constellation Model (in red) on the cars (side) dataset.

6.4 Star model experiments

A clear limitation of the full model is the small number of parts and regions that can be handled in learning and recognition. We now substitute the full shape model for the simpler star model and observe the change in performance. The star model allows us to use larger values of N and P. Additionally, we try out different combinations of features types including curves and corner features on a further 4 of the Caltech datasets: Bottles, Camels, Guitar and Houses.

6.4.1 Comparison to full model

We compare the star model directly with the fully connected model, seeing how the recognition performance drops when the configuration representation is simplified. The results are shown in Table 6.5. It is pleasing to see that the drop in performance is relatively small, only a few percent at most. The performance even increases slightly in cases where the shape model is unimportant. Figures 6.17-6.20 show star models for guitars, bottles and houses.

Dataset	Total size of dataset	Full model test error (%)	Star model test error (%)
Airplanes	800	6.4	6.8
Bottles	247	23.6	27.5
Camels	350	23.0	25.7
Cars (Rear)	900	15.8	12.3
Faces	435	9.7	11.9
Guitars	800	7.6	8.3
Houses	800	19.0	21.1
Leopards	200	12.0	15.0
Motorbikes	900	2.7	4.0

Table 6.5: A comparison between the star model and the fully connected model across 9 datasets, comparing test equal error rate. All models used 6 parts, 20 Kadir & Brady detections/image. In general, the drop in performance is a few percent when using the simpler star model. The high error rate for some classes is due to the inappropriate choice of feature type.

6.4.2 Heterogeneous part experiments

Here we fixed all star models to use 6 parts and have 40 detections/feature-type/image and alter the feature types used by the model. Kadir & Brady, multi-scale Harris and Curves types, and combinations thereof, are tried. The different detectors are introduced in Section 2.3. These experiments used the PCA-based representation of curves as explained in Section 4.4.2.

Table 6.6 shows the different combinations of features which were tried, along with the best one picked by means of a training/validation set. For large datasets, some of the training images were kept aside to act as a validation set while for smaller datasets, we were forced to use the training set itself to select the best combination. Table 3.1 gives details of the validation sets across the different datasets. We see a dramatic difference in performance between different feature types. It is interesting to note that several of the classes perform best with all three feature types present. Figure 6.17 shows a heterogeneous star model for Cars (Rear).

Dataset	KB	MSH	C	KB,MSH	KB,C	MSH,C	KB,MSH,C
Airplanes	**6.3**	22.5	27.5	11.3	13.5	18.3	12.5
Bottles	24.2	23.3	17.5	24.2	20.8	**15.0**	17.5
Camel	25.7	**20.6**	26.9	24.6	24.0	22.9	24.6
Cars (Rear)	11.8	6.0	5.0	2.8	4.0	5.3	**2.3**
Faces	10.6	16.6	17.1	12.0	13.8	12.9	**10.6**
Guitars	**6.3**	12.8	26.0	8.5	9.3	18.8	12.0
Houses	**17.0**	22.5	36.5	20.8	23.8	26.3	20.5
Leopards	14.0	18.0	45.0	**13.0**	23.0	23.0	18.0
Motorbikes	**3.3**	3.8	8.8	3.0	3.3	3.8	3.5

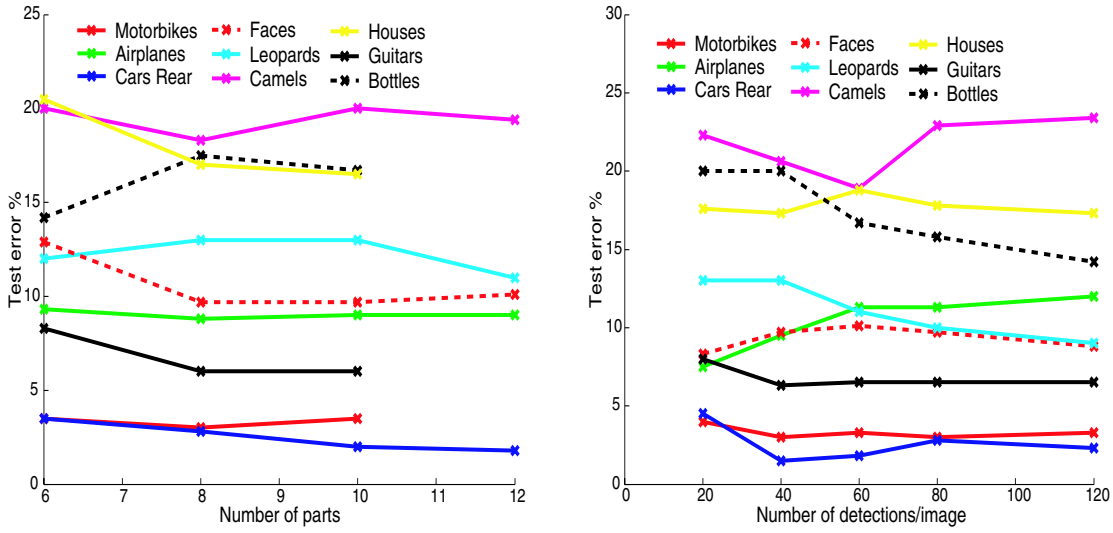
Table 6.6: The effect of using a combination of feature types on test equal error rate. Key: KB = Kadir & Brady; MSH = Multi-scale Harris; C = Curves. All models had 6 parts and 40 detection/feature-type/image. Figure in bold is combination automatically chosen by training/validation set.

6.4.3 Number of parts and detections

Taking advantage of the efficient nature of the star model, we now investigate how the performance alters as the number of parts and the number of detections/feature-type/image is varied. The choice of features-types for each dataset is fixed for these experiments, using the optimal combination, as chosen in Table 6.6.

As the number of parts in the model is increased (for a fixed number of detections/image) some of the categories show a slight change in performance but many remain constant. Examination of the models reveals that many of the additional parts do not find stable features on the object, suggesting that more features on the image are required. Increasing the number of detections/feature-type/image increases the error rate since many of the additional detections lie in the background of the image, so increasing the chances of a false positive. With a suitable combination of feature-types however, the increased number of parts and detections gives a

more complete coverage of the object, improving performance (e.g. Cars (Rear) where the error drops from 4.5% at 8 parts to 1.8% with 12 parts, using 40 detections/image of all 3 feature types).



(a) (b)

Figure 6.16: (a) Test equal error rate versus number of parts, P, in the star model for 40 detections/feature-type/image. (b) Test equal error rate versus the number of detections/feature-type/image, N, for 8 part star models. In both cases the combinations of feature-types used was picked for each dataset from the results in Table 6.6 and fixed.

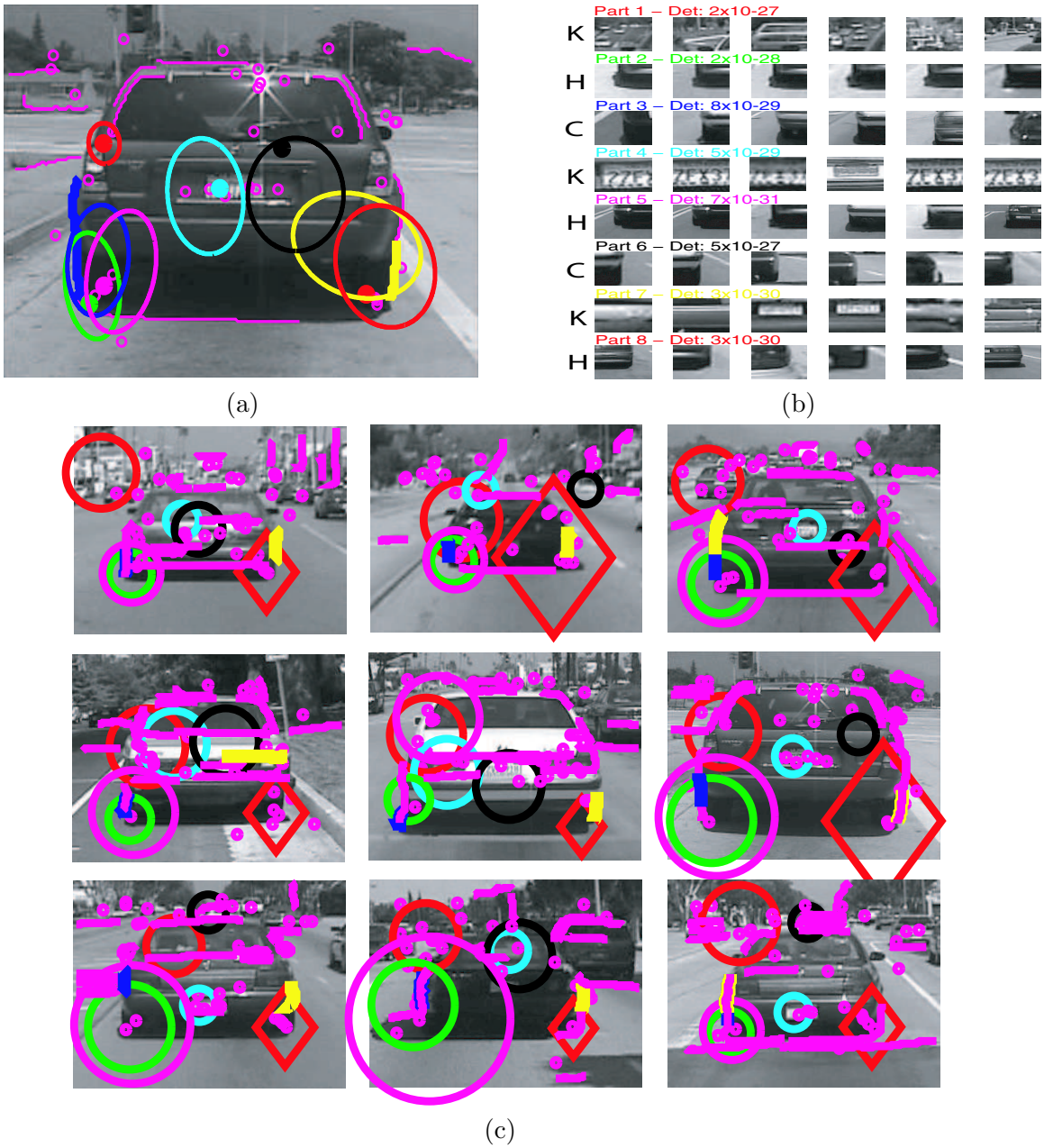
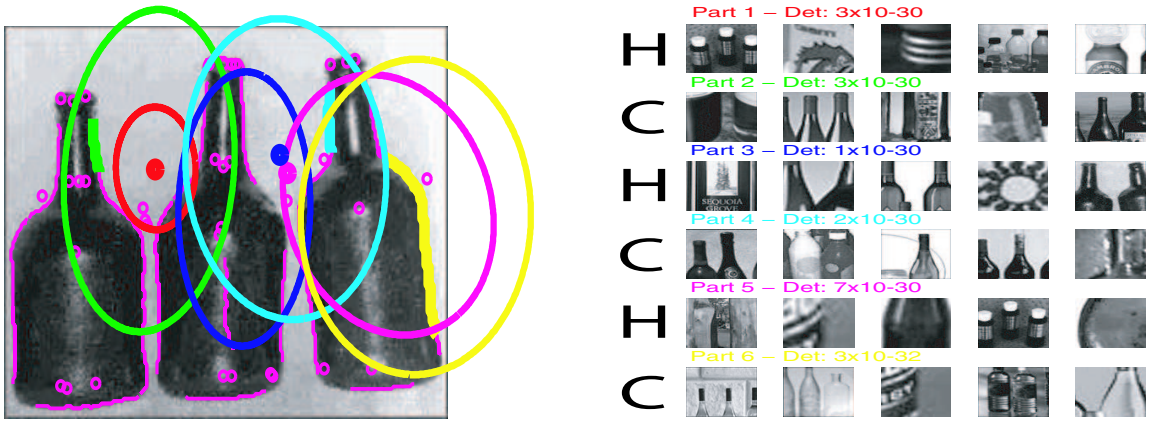


Figure 6.17: An 8 part heterogeneous star model for Cars (Rear), using all three feature types (Kadir & Brady (K); multi-Scale Harris (H); Curves (C)). (a) Detection in a test image with the spatial configuration model overlaid. The coloured dots indicate the centres of regions (K or H) chosen by the hypothesis with the highest likelihood. The thick curve in red is the curve selected by one of the curve parts – the other curve part being unassigned in this example. The magenta dots and thin magenta curves are the centres of regions and curves assigned to the background model. The ellipses of the spatial model show the variance in location of each part. The landmark detection is the top left red one. (b) 7 patches closest to the mean of the appearance density for each part, along with the determinant of the variance matrix, so as to give an idea of the relative tightness of each distribution. The colour of the text corresponds to the colour of the dots in the other panels. The letter by each row indicates the type of each part. (c) More detection examples. Same as top left, but without the spatial model overlaid. The size of the coloured circles and diamonds indicate the scale of regions in the best hypothesis. The test error for this model is 4.5%.

Figure 6.18: An 8 part model for Guitars, using 40 Kadir & Brady features per image. 6.3% test error.



(a)

(b)

(c)

Figure 6.19: A 6 part model for Bottles, using a maximum of 20 Harris regions and 20 Curves per image. 14.2% test error.

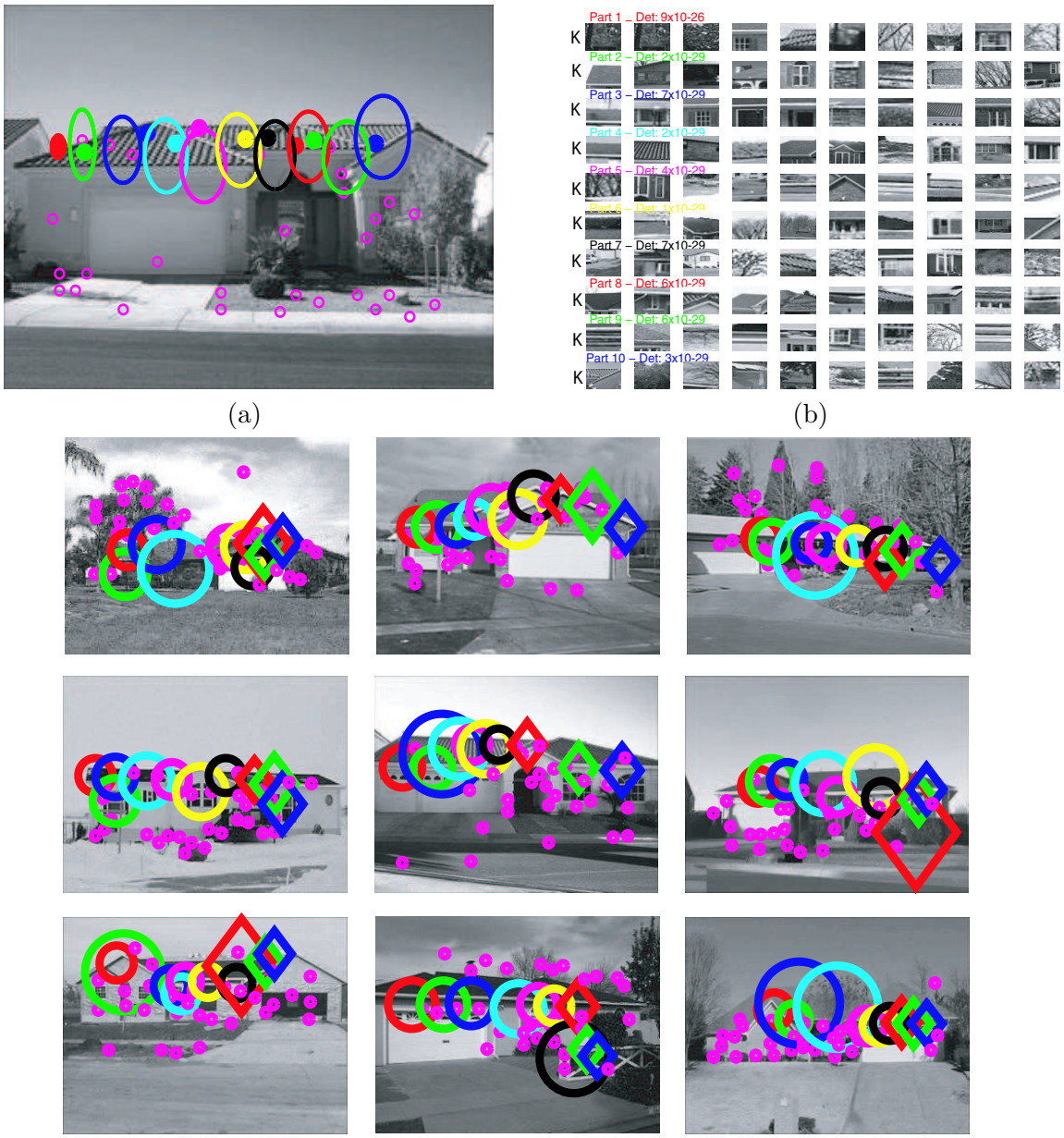


Figure 6.20: A 10 part model for Houses, using 40 Kadir & Brady features per image. 16.5% test error.

Chapter 7

Translation and Scale Invariant Probabilistic Latent Semantic Analysis

In this chapter, we present a second approach to object category recognition, very different in nature to the Constellation Model. A shortcoming of the Constellation Model is that the object representation is sparse — only a small number of parts are used, which are modelling a modest number of regions from the image. Consequently, only a fraction of information from each image is used.

Our second approach models histograms of regions instead of the regions themselves so is able to use many hundreds or thousands of regions per image. It is based on Probabilistic Latent Semantic Analysis (pLSA), a technique from the text analysis community. Recently, Sivic *et al.* [100] and Fei-Fei and Perona [34] have applied this (and a related approach, Latent Dirichlet Allocation) to the visual domain, tackling object recognition and scene analysis respectively. The drawback to these schemes is that they incorporate no spatial information, hence are a simple bag-of-words model [26]. Our approach adds location information into the pLSA model in such a way that translation and scale invariance is achieved and localization of objects within an image is possible.

The structure of the chapter is as follows: we first review pLSA in the textual domain and then explain how it can be applied to vision, following the approach of Sivic *et al.* and Fei-Fei

and Perona. We then introduce a series of additions to the scheme, adding location information to the model. We discuss the implementational details of our scheme and then assess it on the Caltech datasets. We also perform some elucidatory experiments.

7.1 Probabilistic Latent Semantic Analysis (pLSA)

A problem of significant interest in the text analysis community is that of extracting coherent components, such as topics, from a large corpus of documents. Latent Semantic Analysis (LSA) is an approach whereby each document is represented by a histogram of word counts over a vocabulary of fixed size. The histograms from all documents in the corpus form a large co-occurrence matrix which is then decomposed using SVD, with the eigenvectors corresponding to different topics within the corpus and the eigenvalues giving their relative weighting.

Hofmann [52] placed LSA in a probabilistic context, calling it Probabilistic Latent Semantic Analysis (pLSA). Each document is modelled as a mixture of Z topics, each topic being a distribution over the vocabulary of words. More formally, we have a set of D documents, each modelled as a histogram of word counts over a vocabulary of size W. The corpus of documents is represented by a co-occurrence matrix of size $W \times D$, with entry $n(w, d)$ listing the number of words w in document d. Document d has N_d words in total. The model has a single latent topic variable, z, associating the occurrence of word w to document d:

$$P(w, d) = \sum_{z=1}^{Z} P(w|z)P(z|d)P(d) \tag{7.1}$$

Thus we are decomposing a $W \times D$ matrix into a $W \times Z$ matrix and a $Z \times W$ one. $P(w|z)$ captures the co-occurrence of words within a topic, while $P(z|d)$ gives the weighting of each topic within a document. The graphical model is shown in Figure 7.1. The densities of the model, $P(w|z)$ and $P(z|d)$, are learnt using EM. The E-step computes the posterior over the

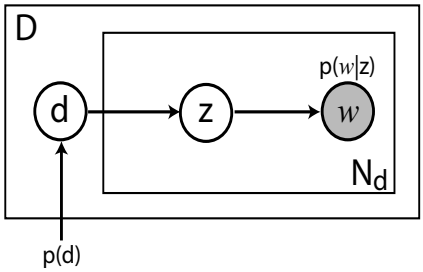


Figure 7.1: Graphical model of pLSA. $p(w|d)$ is a W by Z matrix, each column of which is a discrete distribution on the words of a given topic.

topic, $P(z|w,d)$ and then the M-step updates the densities:

$$\text{E step:} \qquad P(z|w,d) = \frac{P(w|z)P(z|d)}{\sum_{z'} P(w|z')P(z'|d)} \tag{7.2}$$

$$\text{M step:} \qquad P(w|z) \propto \sum_{d=1}^{D} n(w,d)P(z|w,d) \tag{7.3}$$

$$P(z|d) \propto \sum_{w=1}^{W} n(w,d)P(z|w,d) \tag{7.4}$$

This maximizes the likelihood of the model over the data:

$$L = \prod_{d=1}^{D} \prod_{w=1}^{W} P(w,d)^{n(w,d)} \tag{7.5}$$

A novel document d^* is classified by running EM with $P(w|z)$ fixed, so computing $P(z|d^*)$, the mix of topics within the image.

7.1.1 Latent Dirichlet Allocation (LDA)

Strictly speaking, pLSA is not a generative model since it is unable to generate novel documents. A hierarchical Bayesian version of pLSA was introduced by Blei and Jordan [14], called Latent Dirichlet Allocation (LDA), addresses such issues. This approach is more rigorous from a mathematical perspective and gives the ability to introduce priors into the learning scheme; being adopted by Fei-Fei and Perona [34] for scene analysis.

	Domain	
Symbol	Text	Visual
d	Document	Image
w	Word	Visual word
z	Topic	Topic / Object

Table 7.1: Different concepts within pLSA as applied to the text and visual domains.

7.2 Applying pLSA to visual data

While pLSA and LDA are established methods in the text community, their application to visual data is novel. Sivic *et al.* [100] applies pLSA to unsupervised category discovery in a dataset of prepared images, while Fei-Fei and Perona [34] apply LDA in a weakly supervised manner to scene classification. Both make similar analogies between the text and visual domains, summarised in Table 7.1. The terms in the table are self-evident except for *visual words* which we now explain.

7.2.1 Visual words

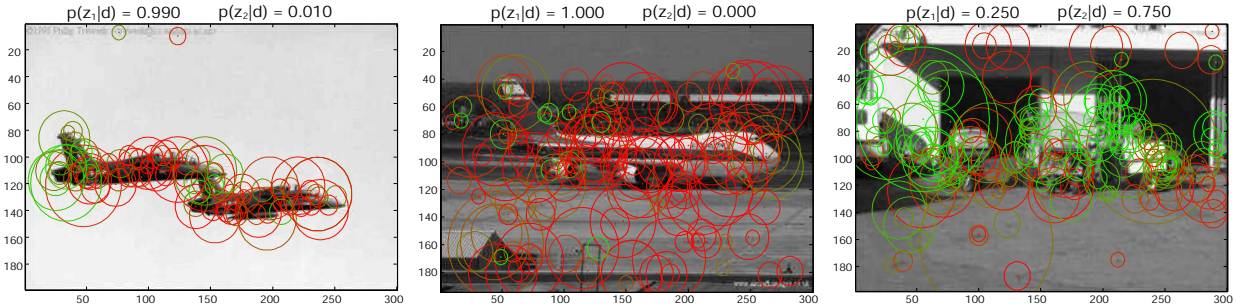
Both Sivic *et al.* and Fei-Fei and Perona adopt the same procedure whereby a set of regions is extracted from the image and their appearance vector quantized to a pre-built vocabulary of *visual words* [26, 101]. No location information is taken from the regions, the image being represented solely by a histogram of visual words.

The visual vocabulary is built by running k-means on a large set of regions from an independent set of training images of widely varying content. The size of the vocabulary is a specified parameter of the system, with Sivic *et al.* and Fei-Fei and Perona using W in the range $200 - 2000$.

Sivic *et al.* described each region using a 128 dimensional SIFT descriptor [70], as described in Section 2.3.7. Fei-Fei and Perona tried two different representations for each region: first, the SIFT descriptor and second, normalizing it to an 11 by 11 patch and taking the raw intensity values as a 121 dimensional vector. Their experiments showed that the SIFT descriptor gave a superior performance.

7.2.2 An example

To consolidate our description we give a toy example. Figure 7.2(a)-(c) shows the results of a two topic model trained on a collection of images of which 50% were airplanes from the Caltech datasets and the other 50% were background scenes from the Caltech datasets. Learning is performed in an unsupervised manner in that image labels (airplane or background) were not provided to the algorithm. The regions are coloured according to the most likely topic of their visual word (using $P(w|z)$): red for the first topic (which happens to pick out the airplane images) and green for the second (which picks out background images). $P(z|d)$ is shown above each image.



(a) (b) (c)

Figure 7.2: (a)–(c) Two airplane images and one background image, with regions superimposed, coloured according to topic of a learnt pLSA model. Red corresponds to topic 1; green to topic 2. The colour of each circle is given by the probability of belonging to each topic, based on the regions' visual word (e.g. a brown circle means the region is equally likely to belong to both topics). Only a subset of regions are shown for clarity. The numbers above each image show the overall weighting of each topic in the image.

7.3 Adding location into the pLSA model

Having described how pLSA may be applied to visual data, we now introduce extensions to the model, designed to incorporate location information so that it can be used to localize object instances with images. We choose to apply these changes to pLSA rather than LDA since the former is simpler in nature. The benefits of LDA are marginal since, for the most part, we do not intend to train the model from a small number of images, making the priors in LDA irrelevant. However, the proposed changes could easily be made to LDA also.

The first addition to the model is a simple one, intended as a baseline method for the more

practical scheme that follows.

7.3.1 Absolute Position pLSA

A straightforward way to incorporate location is to quantize the location within the image into one of X bins and then to have a joint density on the appearance and location of each region. Thus $P(w|z)$ in pLSA becomes $P(w, x|z)$, a discrete density of size $(W \times X) \times Z$:

$$P(w, x, d) = \sum_{z=1}^{Z} P(w, x|z)P(z|d) \tag{7.6}$$

We denote this model ABS-pLSA. The graphical model is shown in Figure 7.3. The same pLSA

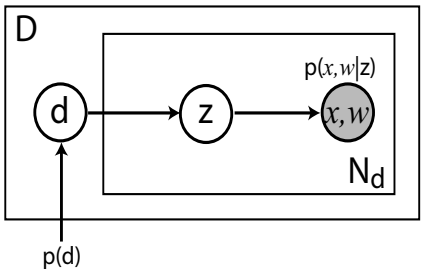
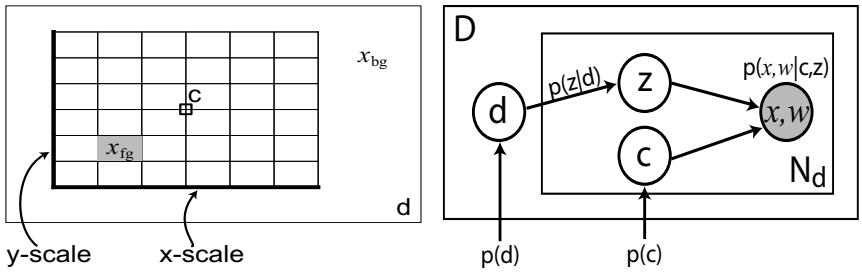


Figure 7.3: Graphical model of ABS-pLSA

update equations outlined above can be easily applied to this model in learning and recognition. The problem with this representation is that it is not translation or scale invariant at all, since x is an absolute coordinate frame.

7.3.2 Scale and Translation Invariant pLSA

The shortcomings of the ABS-pLSA model are addressed by introducing a second latent variable, c, which represents the position of the centroid of the object within the image, as well as its x-scale and y-scale, making it a 4-vector specifying a bounding box. As illustrated in Figure 7.4(a), location x is now modelled relative to the centroid c, over a sub-window of the image. Within the sub-window, there are X_{fg} location bins and one large background bin, x_{bg}, giving a total of $X = X_{fg} + 1$ locations a word can occur in. The word and location variables are then modelled jointly, as in section 7.3.1. This approach means that we confine our modelling of location to only the object itself where dependencies are likely to be present and not the background, where such correlations are unlikely. The graphical model of this approach is



(a) (b)

Figure 7.4: (a) The sub-window plus background location model. (b) Graphical model for translation and scale invariant (TSI-pLSA).

shown in Figure 7.4(b).

We do not model an explicit $P(w, x|c, z)$, since that would require establishing correspondence between images as c remains in an absolute coordinate frame. Rather, we marginalize out over c, meaning that we only model $P(w, x|z)$:

$$P(w, x|z) = \sum_c P(w, x, c|z) = \sum_c P(w, x|c, z)P(c|z) = \sum_c P(w, x|c, z)P(c) \qquad (7.7)$$

$P(c)$ here is a multinomial density over possible locations and scales, independent of topic, making for straightforward adaptations of the standard pLSA learning equations: $P(w, x|z)$ in (7.6) is substituted with the expression in (7.7). In learning we aggregate the results of moving the sub-window over the locations c.

Due to the high dimensionality of the space of c, it is not possible to marginalize exhaustively over scale and location within the image. Instead we use a small set of c, proposed in a bottom up manner for each topic.

Proposing object centroids within an image

We first run a standard pLSA model on the corpus and then fit a series of mixtures of Gaussians (having 1 up to K components) to the location of the regions, weighted by $P(w|z)$ for the given topic. The idea is to find clumps of regions that belong strongly to a particular topic, since these may be the object we are trying to model. The mean of the component gives the centroid location while its axis-aligned variance gives the scale of the sub-window in the x and y directions. We try different number of components, since there may be clumps of regions in the background separate from the object, requiring more than one component to fit. To ensure

that all topics observe the data in the same way, each topic uses the centroids from all topics, not just those proposed by itself. This process gives us a small set (of size $ZC = ZK(K+1)/2$) of values of c to sum over for each topic in each frame. We use a flat density for $P(c)$ since we have no more confidence in any one of the c being the actual object than any other. Figure 7.5(a)-(c) shows the pLSA model using to propose centroids for the TSI-pLSA model, which are shown as dashed lines in 7.5(d)-(f). In the example, $K = 2$ and $Z = 2$.

In recognition, the method for proposing object bounding boxes is different to learning. There is no need to learn a standard pLSA model first. Instead, for each topic the average word density over the learnt TSI-pLSA models' sub-window ($\hat{P}(w|z) = 1/X_{fg} \sum_{x_{\text{fg}}} P(w, x|z)$) can be used to weight each region and then compute putative centroids for that topic in the manner above. Having obtained a set of centroids using $\hat{P}(w|z)$, recognition proceeds by locking $P(w, x|z)$ and iterating to find $P(z|d)$ for the novel images. In estimating $P(z|d)$, all states of c are summed over, thus once convergence is reached, we compute c^*, the value of c within a frame which has the highest likelihood. The solid boxes in Figure 7.5(d)-(f) show c^* in each frame.

Observations about TSI-pLSA

- Multiple object instances in a frame can be captured with $K > 1$, with their information being combined by the marginalisation process. See Figure 7.5(d) for an example.

- The model is entirely discrete, consisting of $WXZ + DZ$ parameters, thus is able to cope with multi-modal non-Gaussian distributions. This provides one method for the model to handle multiple aspects of the object since the different word-locations densities for each aspect will appear as different modes within the $P(w, x|z)$ density.

- Since all three approaches use histograms, unless the object occupies a reasonably large proportion of the image, it will not have a sufficient number of detections to compete with regions on the background, meaning that the image is misclassified as background. While the sub-window approach of TSI-pLSA will help, it cannot overcome this effect entirely, so the object must still occupy a reasonable proportion of the image (1/5 to 1/3 of image area).

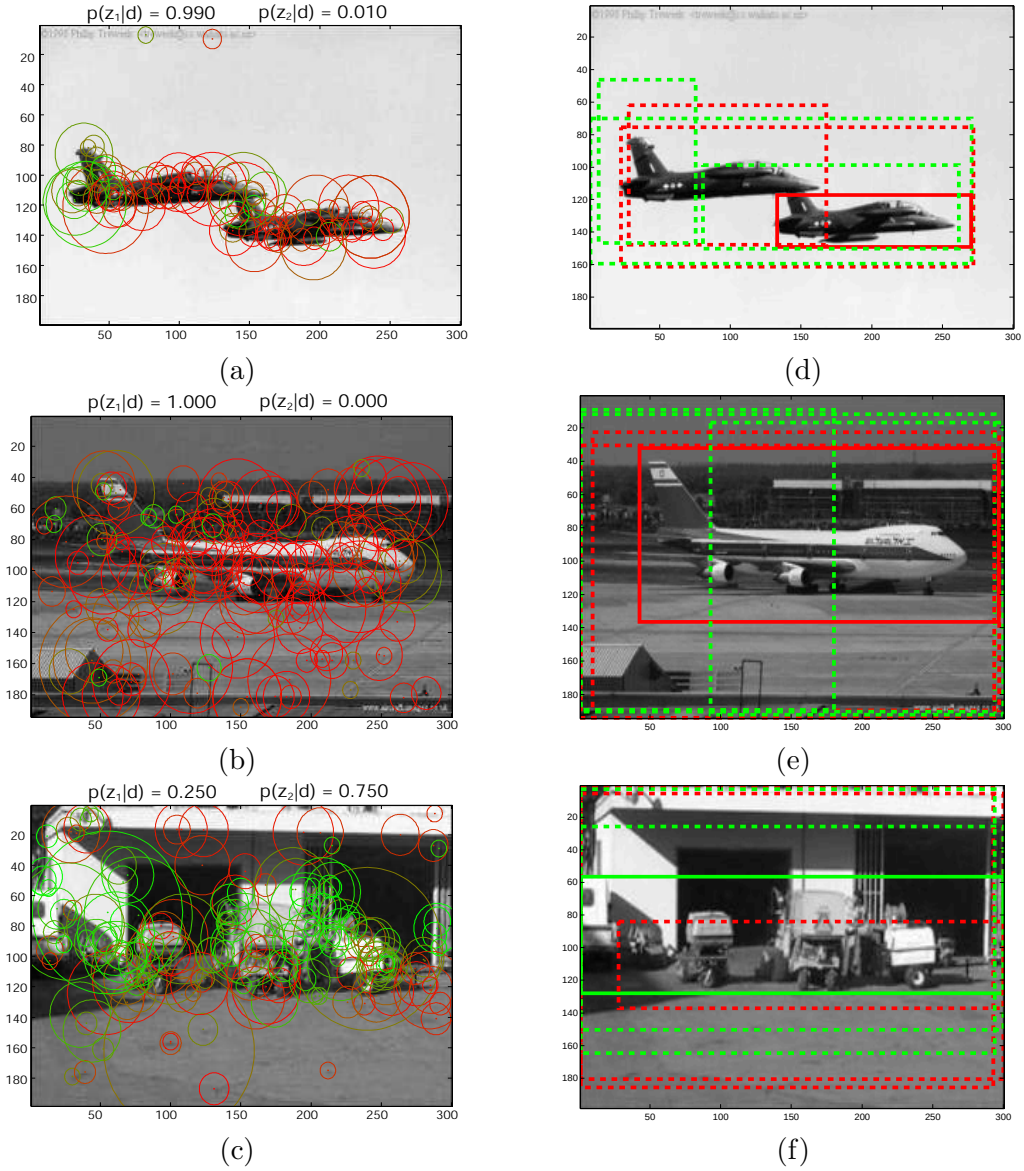


Figure 7.5: (a)–(c) Two airplane and one background image, with regions superimposed, coloured according to topic of a learnt pLSA model. Only a subset of regions are shown for clarity. (d)–(f) The same images as in (a)–(c) but showing the bounding boxes proposed by the pLSA model with dashed lines. The solid rectangle shows the centroid with highest likelihood under a TSI-pLSA model, with the colour indicating topic (the red topic appears to select airplanes). (d) shows multiple instances being handled correctly. (e) shows the object being localized correctly in the presence of background clutter.

- In recognition, the localization accuracy is improved by only finding c^* over those centroids belonging to the actual topic, rather than using all ZC centroids as in learning.

- A shortcoming of the current approach is that the method by which the bounding boxes are proposed does not utilize location information — it just uses a plain pLSA model. We

return to this point in Section 10.2.

7.4 Region detectors

pLSA based methods use histograms to model the image. Since a histogram is only good a estimate of the underlying density if constructed from a large number of samples, our approaches require a rich description on the image using many hundreds or thousands of regions. This is particularly important given size of the histograms in our models, W being $O(10^3)$. We therefore use four types of region detector to give a complete coverage of the image: (i) Kadir & Brady saliency operator [56]; (ii) Multi-scale Harris detector [77]; (iii) Difference of Gaussians, as used by Lowe [70] and (iv) Sampled Edge operator. These are described in Section 2.3. Figure 7.6 shows a single image with a few regions from each of the four detectors overlaid, coloured according to their visual word.

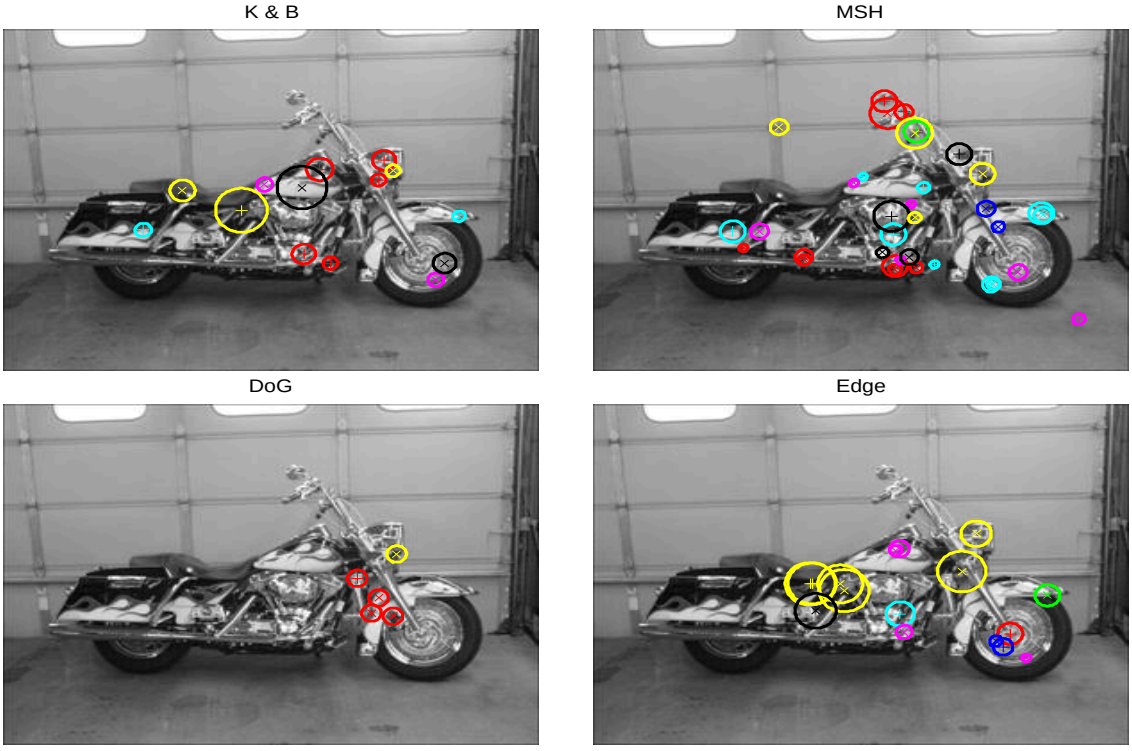


Figure 7.6: Sample image with regions from the four different detectors superimposed. Key: K & B - Kadir and Brady; MSH - Multi-scale Harris; DoG - Difference of Gaussians; Edge - Sampled Edge operator. Only regions belonging to the top 14 most distinctive words from a pLSA motorbike model are shown, each in a unique colour/symbol (+ or x) combination which is consistent across panels.

7.5 Implementational details

Having outlined the three approaches that we will investigate (pLSA; ABS-pLSA and TSI-pLSA), we now give specific details. All images are first converted to grayscale and resized to a moderate width (300 pixels in our experiments). No further normalization of any kind was carried out.

On average, around $N = 700$ regions per image were found, with Kadir & Brady and the difference of Gaussians giving around 100 per image; the sampled edge detector giving 175, and the multi-scale Harris giving 350.

Having found a large set of regions, we represent them using a SIFT descriptor, using 72 dimensions rather than the usual 128 (instead of using a 4×4 spatial grid, a 3×3 one was used), resulting in larger histogram bins which are more appropriate for object categorization. The regions did not have their orientation normalized before histogramming, making them orientation variant.

The regions are then vector quantized using a fixed codebooks of visual words, pre-computed using k-means from a large set of images drawn from the training sets of many different categories. A separate codebook was formed for each feature type and then combined to give W visual words in total. In our experiments, we used $W = 350$. Regions could be quantized to any word, e.g. we did not restrict edge regions to only be allocated to the sub-section of the codebook formed from edge regions alone.

Mindful of the need to keep the number of parameters to a reasonable level, the two approaches with spatial densities used a grid of moderate coarseness. The sub-window used in the experiments had a 6×6 grid, giving $X = 37$.

The proposed model has a large number of parameters: assuming $X = 37$, $W = 350$, $D = 500$, $N = 700$ and $Z = 8$, we have $109,200$ parameters which are estimated from $350,000$ data points, giving a data/parameter ratio of just over 3, the minimum sensible level. We return to this point in Section 7.7.

Training a TSI-pLSA model with $Z = 8$, $D \sim 500$ and the aforementioned parameters takes roughly 30 minutes using a Matlab implementation. ABS-pLSA takes approximately the same time. pLSA takes around half a minute. 100 iterations of EM were used.

Category	pLSA	ABS-pLSA	TSI-pLSA	CM
Airplane	17.7	13.2	4.7	6.3
Cars Rear	2.0	0.2	0.7	2.3
Face	22.1	11.5	17.0	8.3
Guitar	9.3	10.0	14.4	6.3
Leopard	12.0	12.0	11.0	11.0
Motorbike	19.0	6.0	7.0	3.3
Wrist watch	21.6	7.7	15.5	-

Table 7.2: Comparison of the 3 pLSA-based methods trained in an unsupervised manner and the Constellation Model (using standard settings, taking the lowest error rate obtained in Chapter 6) on 7 Caltech datasets. The task is classification, with the figures being the error rate at point of equal-error on an ROC curve. The error margins are roughly $+/-2\%$.

7.6 Caltech experiments

Classification experiments were performed on 7 of the Caltech datasets, using pLSA, ABS-pLSA and TSI-pLSA: airplanes, cars rear, faces, leopards, motorbikes, wrist watches and guitars. The same training and testing images were used as in the experiments of Section 6.1. $Z = 2$ topics were used for all datasets. In training a set of background images, equal in size to the foreground training set was used in conjunction with positively labelled examples, however in the manner of Sivic *et al.* [100], no image labels were given — the models had to learn which images belonged to each component. We return to this point in Section 7.7.1.

Examining Table 7.2, the addition of some kind of location information would seem to lower error rates for the majority of categories. The difference between ABS-pLSA and TSI-pLSA is less pronounced. In some of the Caltech datasets, such as wrist watch, the pose is very consistent so the ABS-pLSA method suffices. For others, such as airplanes, the pose is more variable thus TSI-pLSA shows a significant improvement over ABS-pLSA.

Comparing to the Constellation Model, the error rates are similar but favour the parts and structure approach. For faces and guitars the Constellation Model is clearly superior. Other categories have similar error rates for the two methods.

The categorization performance for TSI-pLSA was also tested using 5 of the Caltech datasets. The confusion table shown in Figure 7.7 compares favourably with the performance of the Constellation Model under the same conditions: TSI-pLSA has a mean diagonal of 92.0% compared to 83.1%. Visualising the models learnt is difficult given the high dimensionality of the topic distributions. In Figures 7.8–7.12 we show example test images with regions from the

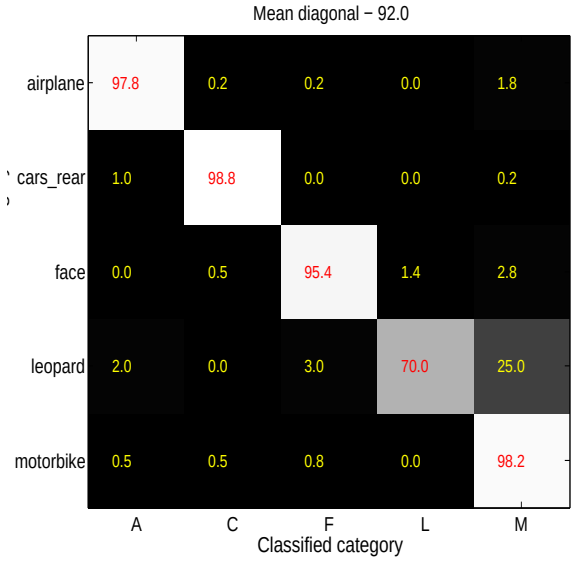


Figure 7.7: Confusion table for TSI-pLSA on 5 Caltech datasets. By comparison, the Constellation Model achieves a mean diagonal of 83.1%.

most common few words superimposed, along with their corresponding location densities (i.e. $\sum_w p(w, x|z)$, x_{bg} not being shown). For some categories, definite structures on the objects are picked out, for example the horizontal edges on the rear of the cars (see Figure 7.8) or the spots of the leopards (see Figure 7.10). However, it is noticeable that a large portion of the regions do not lie on the object but on the background of the image. This is reflected in high entropy of the location densities.

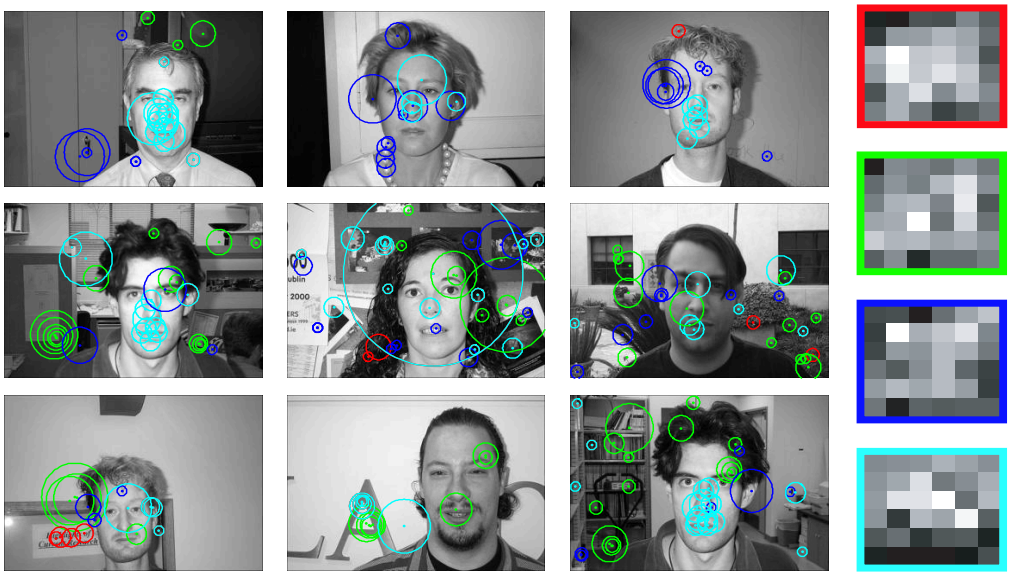
The localization performance of TSI-pLSA was also tested. Figures 7.13-7.17(a) show recall-precision curves for TSI-pLSA and the Constellation Model on 5 Caltech datasets. A baseline measure is included in the plots, whereby the putative bounding box is taken to be the size of the image and its likelihood is equal to that of the TSI-pLSA model. Examples of correct and incorrect localization are shown in Figures 7.13-7.17(b).

In localization, the TSI-pLSA model performs comparably to the Constellation Model, except for the face dataset where it consistently selects an overly large bounding box. On motorbikes the same behaviour is also exhibited but this time it gives a beneficial performance as the motorbikes typically fill most of the frame. Aside from the overly easy motorbikes, TSI-pLSA beats its baseline method comfortably.

From the results on the Caltech datasets, some tentative conclusions can be drawn. While the addition of some form of location information is helpful, the datasets themselves are not

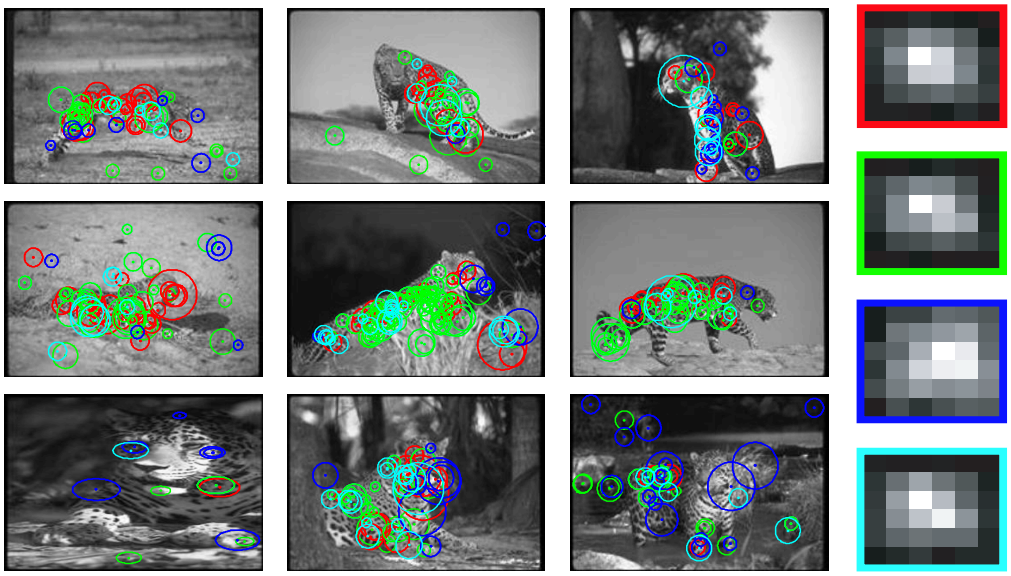
(a) (b)

Figure 7.8: Visualisation of the TSI-pLSA model for cars (rear). (a) Test examples overlaid with regions from the 4 most common words for the motorbike topic. The colours of the regions correspond to the visual word of their appearance: red, green, blue and cyan for the 1st to 4th most common words respectively. (b) Location densities of the subwindow for each of the words, the colours corresponding to those in (a). Black corresponds to low probability; white to high. The green regions pick out the horizontal edge structure along the rear of the cars. Note the correspondingly orientated location density



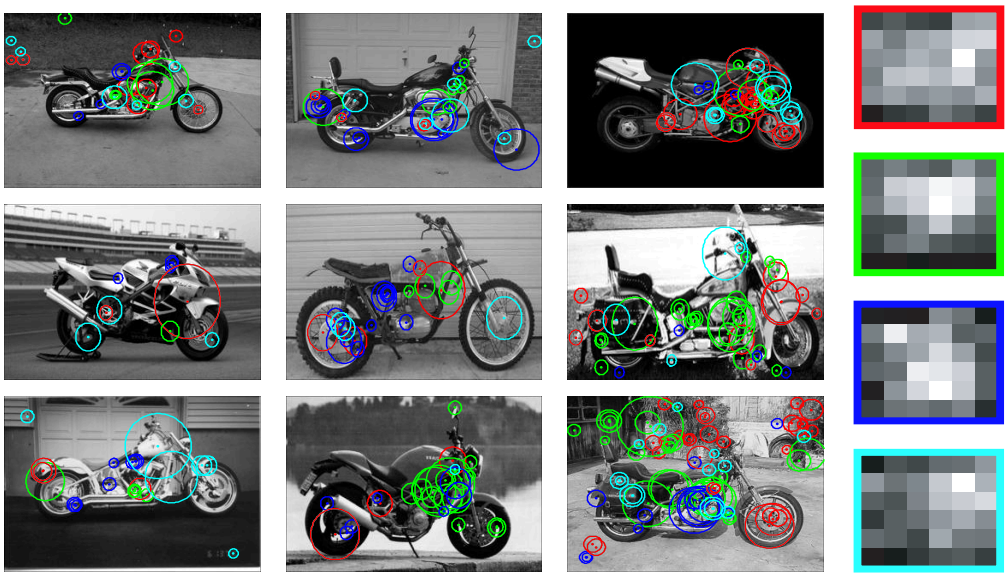
(a) (b)

Figure 7.9: Visualisation of the TSI-pLSA model for faces. While the cyan regions are frequently present on the nose and mouth, both they and other regions occur frequently in the background of the images. This is also reflected in the flat nature of the location densities.



(a) (b)

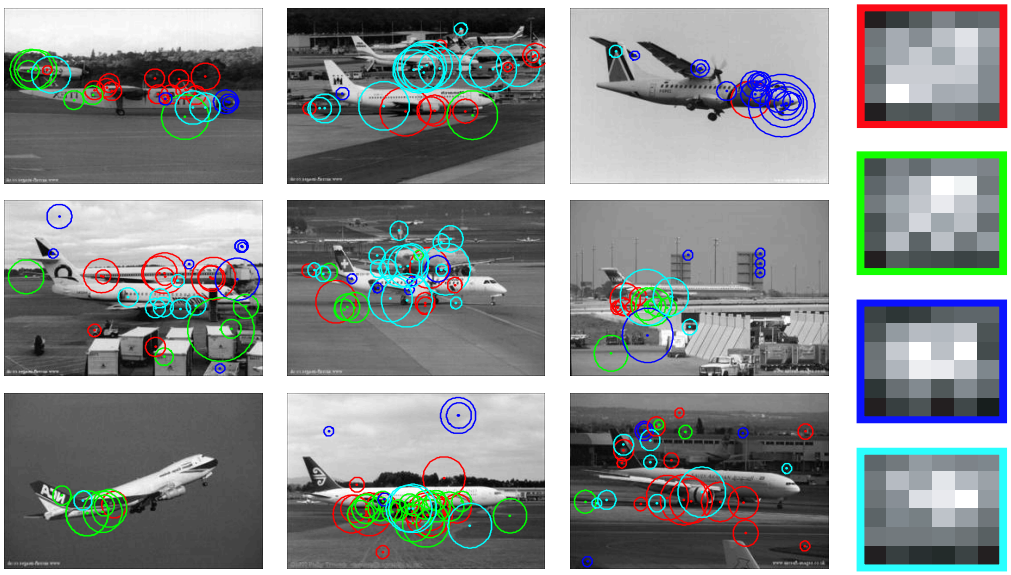
Figure 7.10: Visualisation of the TSI-pLSA model for leopards. The red and green regions pick out the distinctive spotted fur of the cat.



(a) (b)

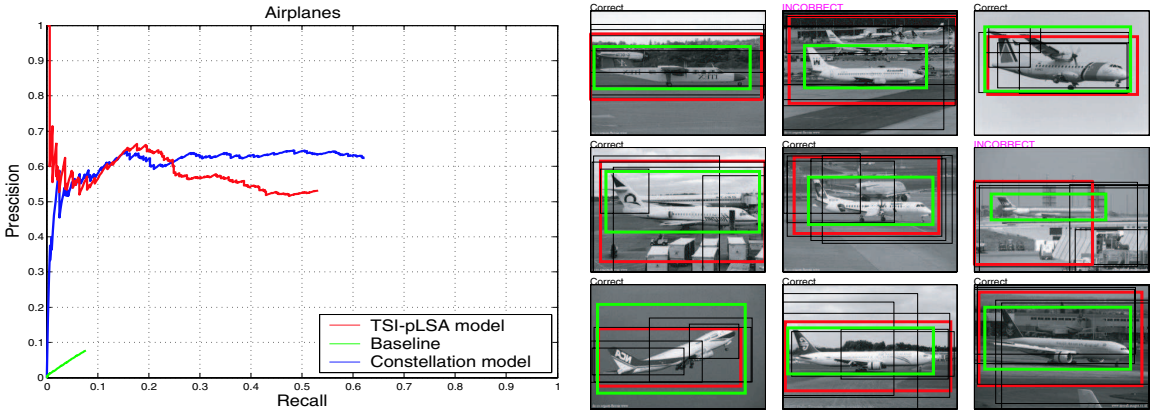
Figure 7.11: Visualisation of the TSI-pLSA model for motorbikes. The regions are somewhat variable in their location, but pick out the front handlebar-fork-wheel structures. The green regions pick out the engine block structure of the bike, having a characteristic texture due the the highlights and shadows from the chrome components.

sufficiently challenging to reveal the difference between ABS-pLSA and TSI-pLSA, as seen by the similar error rates in classification. However there is a more fundamental explanation for the lack of improvement of TSI-pLSA over ABS-pLSA. Looking at the localization examples,



(a) (b)

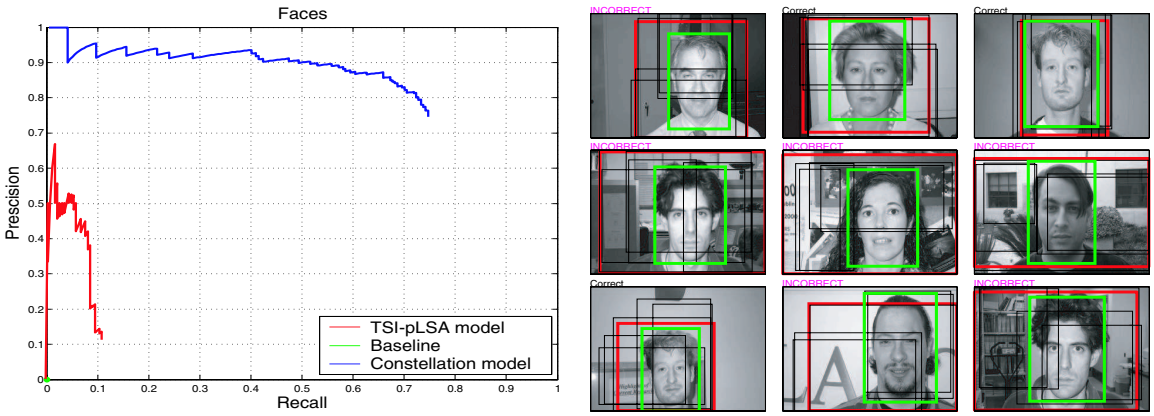
Figure 7.12: Visualisation of the TSI-pLSA model for airplanes. It is somewhat unclear what the model is learning since many of the regions commonly occur in the background of the images.



(a) (b)

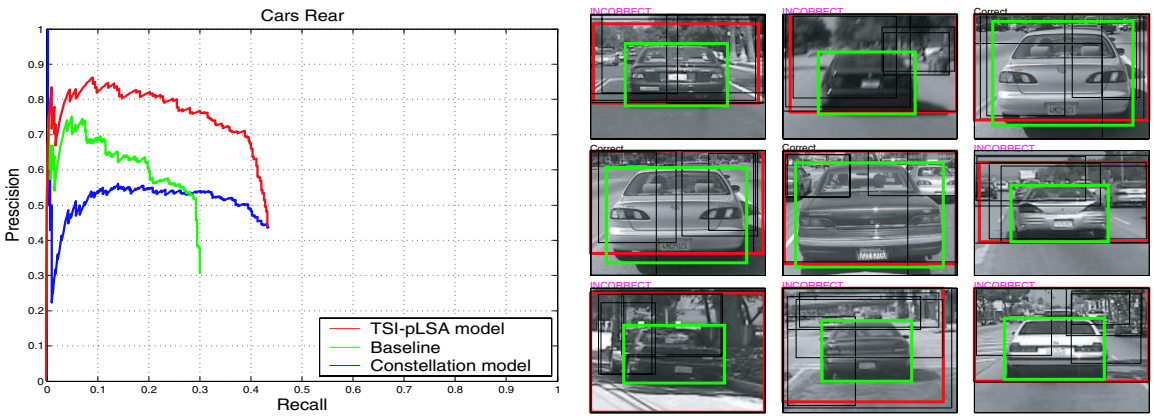
Figure 7.13: (a) The recall-precision curve for airplanes. The performance of TSI-pLSA is shown in red; the baseline (see text for explanation) in green and the scale-invariant Constellation Model in blue. (b) Examples of attempts at airplanes localization. The putative bounding box is shown in red while the ground truth is shown in green. The thin black boxes are proposed bounding boxes not selected by the algorithm.

it is clear that in most cases the bounding box is very large, often nearly the whole image. The exception to this is where there is no background clutter, which leads to the suspicion that the model is learning not just the object but its characteristic background. Evidence for this includes:



(a) (b)

Figure 7.14: (a) The recall-precision curve for faces. (b) Localisation examples. Note some of the unused bounding boxes are superior to the selected one (e.g. middle bottom image). This issue is discussed in the text.



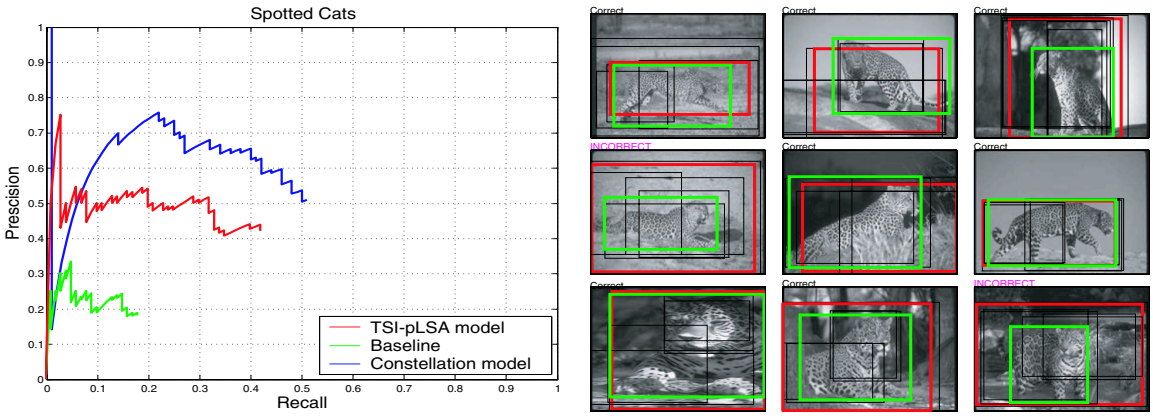
(a) (b)

Figure 7.15: (a) The recall-precision curve for cars (rear). (b) Localisation examples.

- The mechanism for proposing bounding boxes often proposes superior bounding boxes, localizing the object more tightly (see Figure 7.14), only to score less well than larger boxes which incorporate more background.

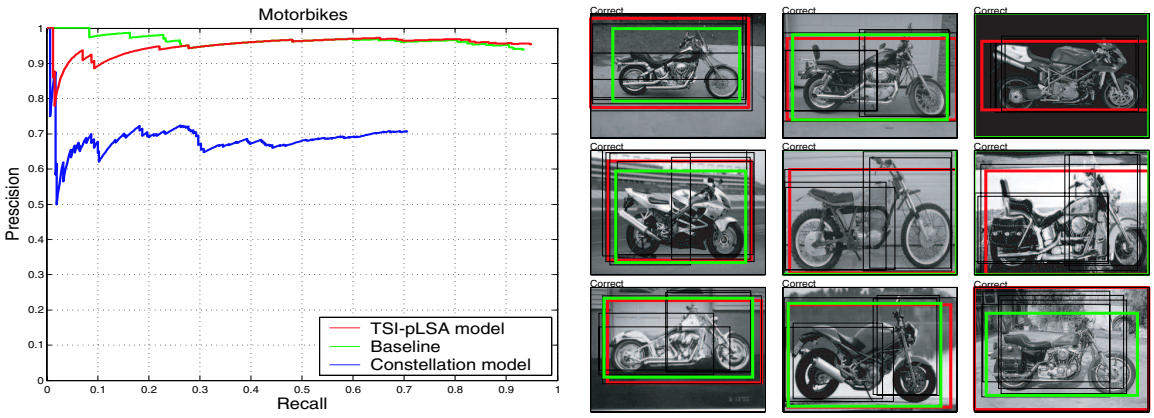
- The flat location densities and seemingly random location of regions belonging to common words in Figures 7.8–7.12.

- While the performance in a foreground/background classification task is similar to the Constellation Model, in categorization it is significantly better. The discrepancy in performance between the two tasks supports our argument in the following way: the latter



(a) (b)

Figure 7.16: (a) The recall-precision curve for leopards. (b) Localisation examples.



(a) (b)

Figure 7.17: (a) The recall-precision curve for motorbikes. (b) Localisation examples.

task's performance would be boosted if the model were using the background statistics of each of the datasets since each classes' different background would reduce confusions between the objects themselves. In classification, the background of each object class will have similar statistics to some of the images from the background dataset, so learning the background around the object will not help in this case.

- The phenomenon of learning the object and its background together was also observed by Sivic *et al.* [100], although only when using plain pLSA.

Such issues are a hazard of using purely generative model. The Constellation Model only avoids the problem by having such a sparse representation that no parts can be spared to model the background. In Chapter 8, we introduce more challenging datasets where the background

is more weakly correlated to the object so that this problem is obviated.

7.7 Model investigations

The model has a number of settings which should ideally be determined by experiment. However, some of these have been investigated in related work such as Fei-Fei and Perona and Sivic *et al.* [34, 100] and we use their findings to guide our design choices. These include:

- Feature detectors - We use wide range of types used by [34, 100], designed to be somewhat complementary in nature. Given the large number found per image, their output may be regarded as smart sampling of the image rather than giving a small set of distinctive regions as for the Constellation Model.

- Feature description - Fei-Fei and Perona found SIFT to be a superior descriptor to PCA hence we adopt SIFT.

- Codebook size - Both [34, 100] find that optimal performance is achieved in the range $500 - 3000$. Too few and the model cannot adequately describe the appearance of all regions found. Too many and the entries becomes too specific and unable to generalize well. Our use of 350 is on the low end of the range above, mainly to keep the number of parameters in the model to a reasonable level, given the joint nature of the word-location density.

7.7.1 Weakly Supervised versus Unsupervised learning

In the experiments of Section 7.6, no image labels were provided for learning. We now add this information to the learning scheme to see the difference in performance. The labels may be added by assigning a topic to each of the classes (e.g. 1 for foreground and 2 for background) and then fixing $P(z|d)$ in learning to a binary-valued matrix holding the image labels, the 1 for each row denoting the class. The effects of this weak supervision are shown in Table 7.3, comparing to the unsupervised results of Table 7.2 and the Constellation Model (which was trained using image labels). The addition of image labels significantly reduces the error rates for pLSA and ABS-pLSA but has little effect on the error rates for TSI-pLSA. The may be due to the imperfect placement of proposed bounding boxes in training.

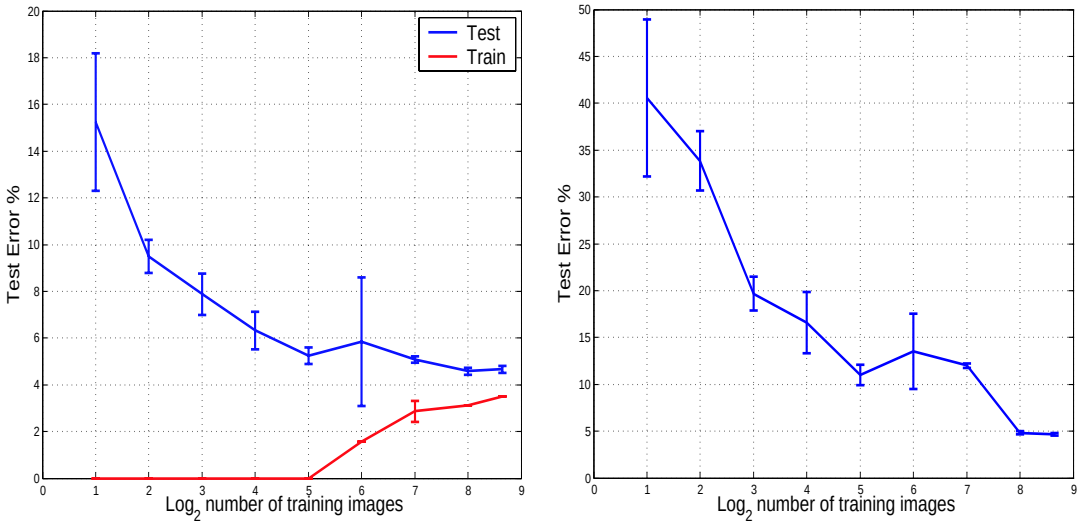
	pLSA	pLSA	ABS-pLSA	ABS-pLSA	TSI-pLSA	TSI-pLSA	CM
Category	S	U	S	U	S	U	S
Airplane	8.4	17.7	4.8	13.2	5.2	4.7	6.3
Cars Rear	1.0	2.0	3.3	0.2	3.3	0.7	2.3
Face	16.4	22.1	6.2	11.5	15.0	17.0	8.3
Guitar	7.5	9.3	2.5	10.0	8.9	14.4	6.3
Leopard	8.0	12.0	7.0	12.0	7.0	11.0	11.0
Motorbike	7.7	19.0	2.7	6.0	5.7	7.0	3.3
Wrist watch	12.7	21.6	8.8	7.7	13.3	15.5	-

Table 7.3: Comparison of the 3 pLSA-based methods trained with weak-supervision (S) and without (U) and the Constellation Model on 7 Caltech datasets. The task is classification, with the figures being the error rate at point of equal-error on an ROC curve. The error rate is reduced significantly for pLSA and ABS-pLSA if labels are used in training. In contrast, with the exception of Guitars, the error rates for TSI-pLSA are unchanged or even a little worse between the two training methods.

7.7.2 Over-fitting

As mentioned, the model has a large number of parameters that must be estimated from training data. In our datasets the size of the training set is typically $O(10^2)$ – small enough for over-fitting to be a problem. Figure 7.18(a) shows the training and test errors in classification as the number of training examples D is varied for the airplane dataset. In this instance, in contrast to the experiments in Section 7.6, training uses image labels. The plot shows that for the size of training sets used in our experiments, there still is some over-fitting present, though it is relatively small, around 1% above the Bayes error.

By comparison with Figure 7.18(a), if training is performed in an unsupervised manner, a set of different curve is obtained, shown in Figure 7.18(b). Comparing to the supervised training case, the test error is much larger for small numbers of training examples, but very similar (both $\sim 5\%$) for large numbers of examples, where the two components of the model have locked onto the two classes.



(a) (b)

Figure 7.18: The test and training errors plotted as the number of positive training images increases from 2 to 400. Note that x-axis is $\log_2$ scale. (a) Supervised training. (b) Unsupervised training. Note the different scales on the y-axis between (a) and (b). The test errors are similar in both plots once more than 250 or so examples are used.

Chapter 8

More challenging data

In this chapter we evaluate the Constellation Model and TSI-pLSA methods on the Fawlty Towers and PASCAL datasets, before comparing both the methods to a variety of rival approaches. While the Caltech datasets are useful for understanding the operation of the model, they do not provide a sufficiently challenging evaluation of localization performance. By contrast, both the Fawlty Towers and PASCAL datasets contain object instances which are a small portion of the image, hence provide a challenging localization task.

We consider two different evaluation paradigms, which we now define: *closed world* and *open world*. The former is where a large collection of images containing the object of interest are taken and split into two disjoint sets, one being used to train the model and the other used for testing. Since all images in the collection are likely to come from similar sources, there may be a close statistical relationship between the training and testing sets. The object instances may exhibit a bias towards certain object poses or not be a truly representative sample of the object class (e.g. in a collection of car images, the cars may be all US or all European rather than a mix of both). Additionally, the background statistics may be similar (e.g in the Caltech leopard dataset the animals frequently at present in jungle scenes — there are no examples of leopards in a concrete zoo environment, for example). These correlations and biases mean that only a limited generalization ability is required for models to perform well and there is the previously mentioned problem of learning the background as well as the foreground object.

By contrast, in the open world world scenario the training and test sets have been gathered from quite different sources, hence the object of interest should be the only consistent element

between the two. Consequently it should be harder to perform well in an open world setting that a closed world one.

8.1 Fawlty Towers datasets

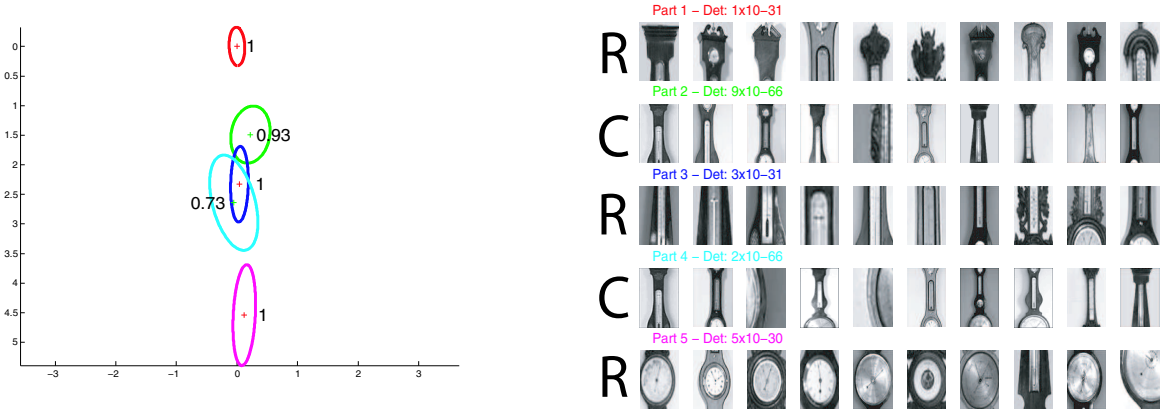
We apply both methods to the Fawlty Towers datasets, as introduced in Section 3.3. The two classes of object that we test with are barometers and cars viewed from the front. The evaluation is done in an open world setting, since we train the models for two categories on images that are statistically different to the Fawlty Towers test data; the training images being hand-selected from Google's image search (see Figure 3.4 for some examples). By evaluating in this way, we can check to ability of the models to be used in recognition away from data that is very similar to the training data (as is the case for the Caltech datasets).

8.1.1 Constellation Model

As the objects in the test images are small, many hundreds of detections will be needed to give good coverage and ensure the object is not missed. This necessitates the use of the star model, since N is $O(10^2)$. We again take the opportunity to evaluate different types of feature in heterogeneous models, using combinations of Kadir and Brady regions and curve features.

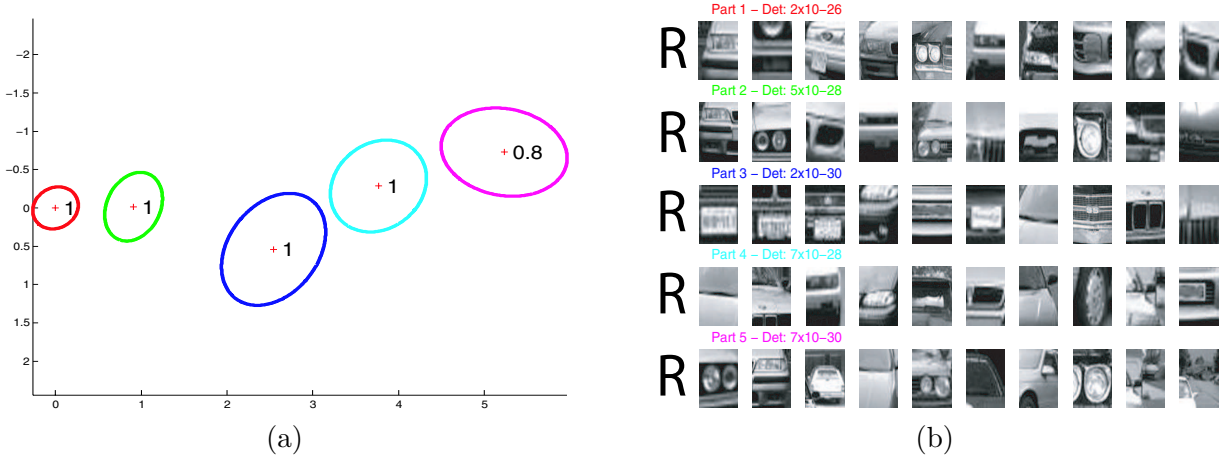
We first train models for each class on the 15 manually collected images from the Internet. Since the exemplars are on uniform backgrounds and have no background clutter, a smaller number of images may be used although the problem of over-fitting still remains due to the number of model parameters. No pre-processing of any kind is performed on them. Kadir & Brady region and Curves feature were extracted from the images. The standard model settings (see Section 6.1) were used, except than the minimum feature scale for the Kadir & Brady regions was raised to 15 pixels in radius, as the instances fill the frames. A scale-invariant star model is then trained with the following settings: $P = 5$ parts and $N = 20$ detections/feature-type/image, for three different combinations of features: all regions, all curves, and 3 regions and 2 curves. Figures 8.1 and 8.2 show two of the models learnt.

In recognition, N is increased to 200 for the regions and 80 for the curves (this being the total number extracted by the curve detector per image). A wider scale range of 8–40 pixels in radius for the Kadir & Brady detector is used given the variability in object size (the



(a) (b)

Figure 8.1: Barometer model consisting of 3 Kadir & Brady region (R) parts and 2 curves (C). (a) Star structured shape model; (b) Samples from the appearance density.



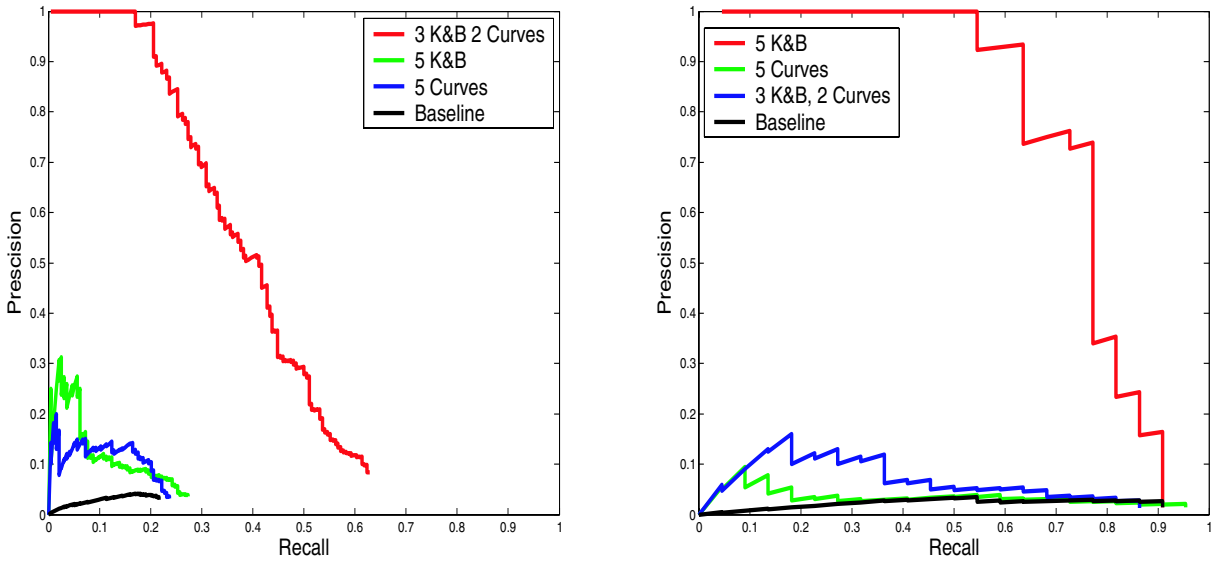
(a) (b)

Figure 8.2: Car front model using 5 Kadir & Brady region (R) parts. (a) Star structured shape model; (b) Samples from the appearance density.

barometer and cars are very different sizes). The performance is evaluated using recall-precision curves, with the criterion for a positive match being that the centroid of the best hypothesis lies within the ground truth bounding box of the object. Models of the three different feature-type combinations are evaluated on the episode "A touch of class", their recall-precision curves shown in Figure 8.3. For the barometers, the combination of curves and regions performs best, while the all-region model works best for the cars. If inappropriate combinations of feature-types are used a drastic drop in performance is observed. For both classes, a baseline method was evaluated, to put the results in context. The baseline consisted of taking a single cropped training example and using it as a normalized-correlation template, using a scale-pyramid to

search over scale as well. As Figure 8.3 shows, the baseline performs poorly compared to the optimal combination of feature types. Note that both objects occupy a very small fraction over the total image area of all frames — 0.3% and 0.6% for barometer and cars respectively, thus achieving high precision is difficult.

In Figure 8.4, the barometer model and baseline method are tested on three different episodes to see how well the model generalizes across different instances (the actual barometer changes from episode to episode). For all episodes, the performance of the constellation model far exceeds that of the baseline method. The variability in performance between the episodes may be partially explained by the changing background environment - for example in "The wedding party", one of the female characters has a dress with lots of bright vertical stripes on, which generates lots of curve features resulting in many false alarms. Figures 8.5 and 8.6 show localization examples for cars and barometer respectively. The magenta dots and curves show the centres of regions and curves that belong to the background, while the coloured curves and circles show the best hypothesis within the image. The red rectangle shows the ground truth location of the object.



(a) (b)

Figure 8.3: Recall-precision curves for different combinations of feature types. (a) Barometers; (b) Car front.

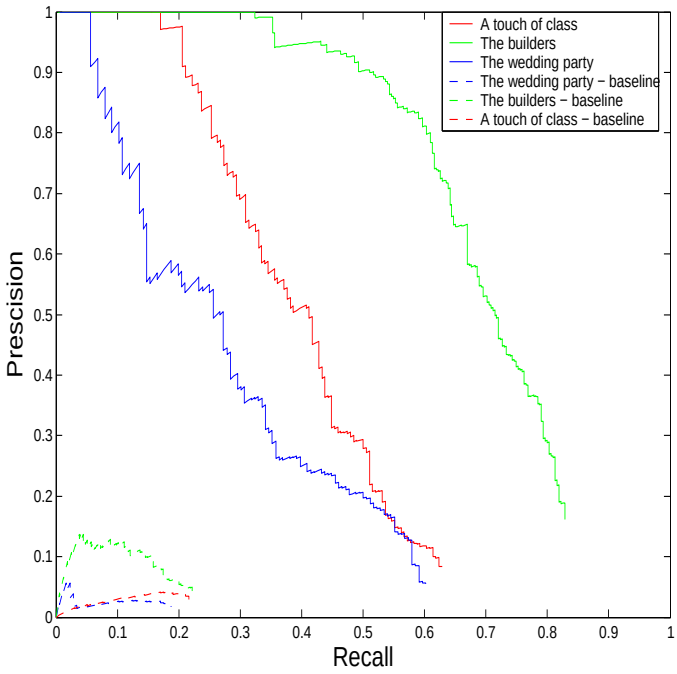


Figure 8.4: Comparison of localization performance for barometers across three episodes of Fawlty Towers, using a 3 region, 2 curve model. See text for an explanation of the baseline method.

Figure 8.5: Examples of cars being localized by a Constellation Model in the Fawlty Towers episode "A touch of class".

Figure 8.6: Examples of barometers being localized by a Constellation Model in the Fawlty Towers episode "A touch of class".

8.1.2 TSI-pLSA experiments

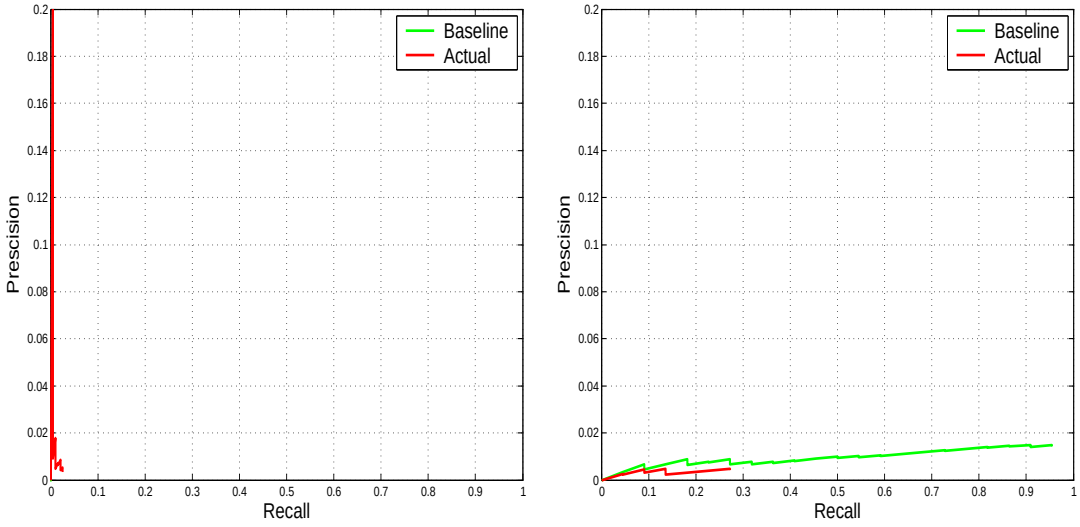
We apply TSI-pLSA to the Fawlty Towers data, using the same settings as described in Section 7.5 with a 2 topic model. Each of the 15 training images was resized so its major axis was 300 pixels in size. Training from so few images poses a slight problem due to over-fitting (see Figure 7.18) therefore we train in a supervised manner, using 15 images drawn from the Caltech background to act as negative examples. According to Figure 7.18(a), this should mean we are $2-3\%$ over the Bayes error.

In recognition, the images were left at their original size (720 by 576) in view of the small size of the object instances. For the barometers, the bounding boxes were proposed by $k = 10$ Gaussian components rather than the usual 2 to help pick out the small object instances. For cars front, the standard $k = 2$ was used since they are typically larger. For evaluation, the same localization criterion as the experiments in Section 8.1.1 are applied.

Recall-precision curves for the two objects on the episode "A Touch of Class" are shown in Figure 8.7. Both methods perform very poorly in comparison to both the Constellation Model and the normalized correlation baseline, shown in Figure 8.3. Included in Figure 8.7 are another

simple baseline where the bounding box is taken to be the whole image and its score being given by TSI-pLSA summing over all centroids in the image. This is the same baseline as used in the Caltech experiments and it indicates how much harder the localization task is for Fawlty Towers data is than for Caltech data, where it performed reasonably well.

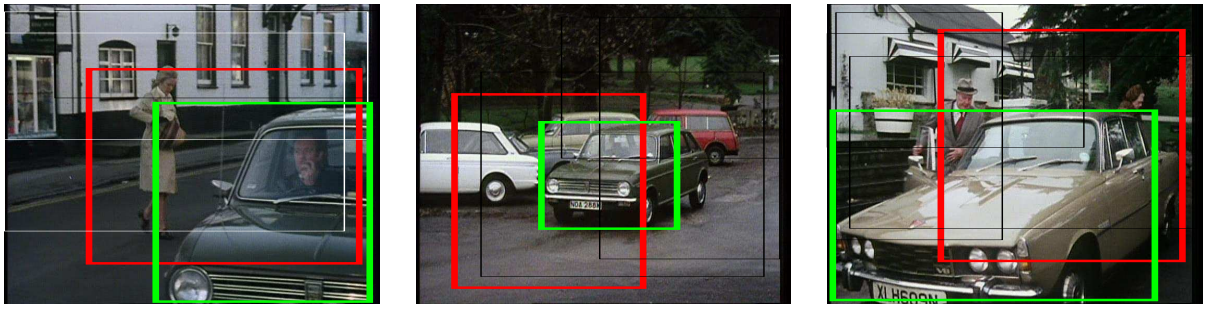
Figure 8.8 explores the reasons for the poor performance. Examples of detection (or lack thereof) are shown in Figure 8.8(a) and (b). (c) shows a sample barometer image in which the regions on the barometer, while correctly classified, are drowned out by the mass of other regions belonging to the same topic, meaning the boxing box proposal scheme fails to find the object.



(a) (b)

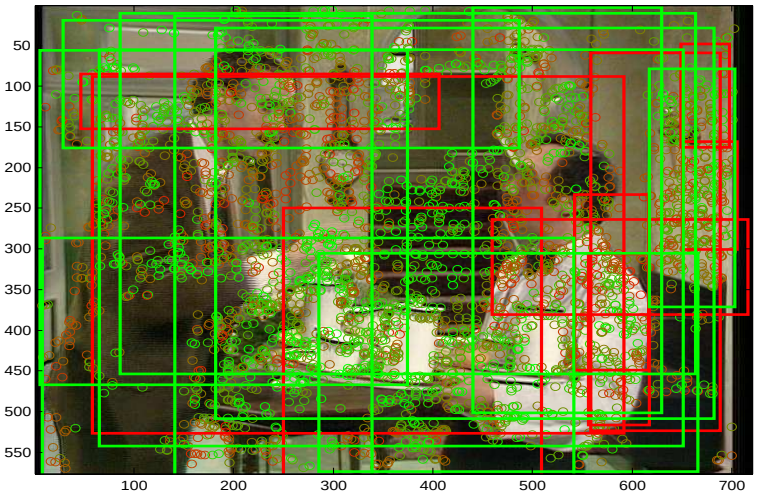
Figure 8.7: Recall-precision curves for (a) barometer and (b) car front models on the Fawlty Towers episode "A Touch of Class". Note the y-axis only extends up to 0.2 so that the curves are visible. The performance of the TSI-pLSA models is shown in red, while the simple baseline is shown in green. TSI-pLSA performs poorly for both objects.

In summary, the Fawlty Towers data is challenging for pLSA-based methods since the actual object occupies a small fraction of the image thus its contribution to the histogram representing each image will be drowned out by the large amount of background clutter unless the topic is highly discriminatory, which it is not in this case. The performance may be improved by (i) using a more accurate model of the background, perhaps adapted to the Fawlty Towers data and (ii) using a top-down scheme to propose the object bounding boxes, thus preventing the signal being swamped by the background.



(a)

(b)



(c)

Figure 8.8: Attempts at localization for (a) car front and (b) barometer. The ground truth box is shown in green. The chosen box is in red, while unused putative boxes are shown in black. In the examples no proposed bounding boxes are close to the true ones. (c) shows a frame containing a barometer overlaid with the centres of all regions in the image, as indicated by coloured circles. The colour indicates the weighting of the visual word belonging to each region between the two topics (red - barometer; green - background). Note that while the regions on the barometer belong to the barometer topic (they are strongly red), they are too small a clump for Gaussian mixtures scheme to lock onto them, given that many other regions in the image belong to the barometer topic.

8.2 PASCAL experiments

The PASCAL object recognition challenge [31] is a competition designed to evaluate object recognition systems on a range of challenging datasets in classification and localization roles.

The datasets are reviewed in Section 3.4. We evaluate both the Constellation Model and the pLSA-based approaches on two of the four PASCAL datasets, car and motorbike (people and bicycles being the other two). Each class has a fixed training and validation set and two test sets, which we refer to as 1 and 2, the latter being more challenging. We first apply the Constellation Model in Section 8.2.1, followed by pLSA-based methods in Section 8.2.2. A benefit of using these datasets is that many other approaches have evaluated their algorithms on the same data, so fair comparisons between approaches can be made. We round off the chapter by plotting the performance of both our two methods against 12 other approaches in Section 8.2.3.

8.2.1 Constellation Model

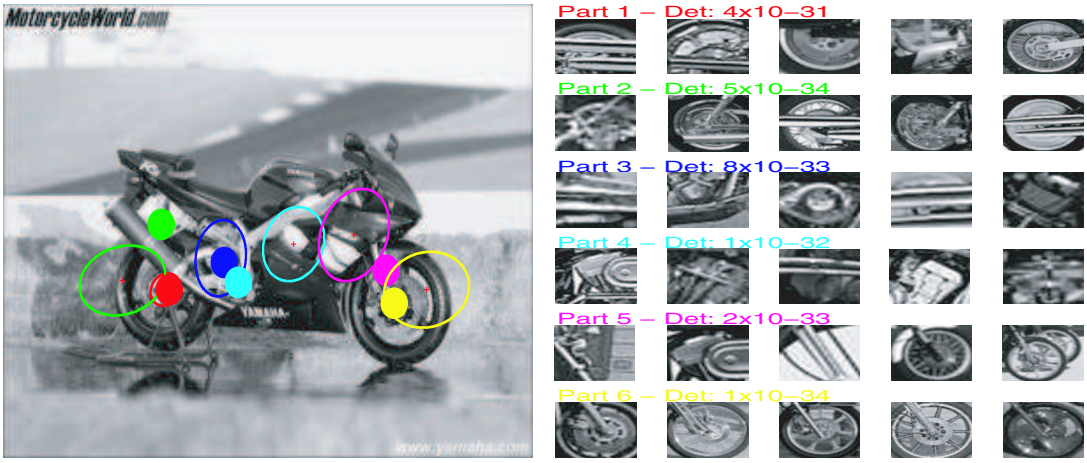
The PASCAL datasets contain far more pose variation that the Caltech or Fawlty Towers images. The pose of the motorbikes is fairly consistent, mainly being left-facing and right-facing side views. For the cars, the pose is more varied with side, front, rear and top views all present. As the Constellation Model can only handle a single viewpoint, this complicates matters so we detail the experiments for each category separately.

Motorbikes

For this class, a single aspect model was learnt with the motorbike facing to the right, in the hope that it generalizes well to left facing instances and other aspects. The small size of the training set, 107 images, means that the training and validation sets were merged to give a training set of 214 images. A variety of 6 part star models with different combinations of Kadir & Brady; multi-scale Harris and Curve features (using the PCA-based curve representation) and $N = 30$ detections/feature-type/image were trained using the standard Constellation Model settings (see Section 6.1). The model performing best on the training (and validation) set was selected as the final classifier and is shown in Figure 8.9(a) & (b). The model uses only Kadir & Brady features, picking out the wheels and engine blocks of the bike.

Since the relation between the bounding box produced by the best hypothesis and the ground truth bounding box is not learnt automatically, we estimate the mean offset in translation and scale between the two boxes from the manual annotations of the training set. The calculated offsets are then applied to predicted bounding boxes when evaluating test images.

The model was applied to test set 1 and 2 with $N = 30$ regions per image. Detection examples on test set 2 are shown in Figure 8.10. The algorithms performance is compared to a variety of other approaches and baselines in Figure 8.17 and Figure 8.18 where its performance is competitive with the leading approaches on the harder test set 2.



(a) (b)

Figure 8.9: (a) Shape model for PASCAL motorbikes, superimposed on an image. All parts use Kadir & Brady regions. The coloured dots indicate the best hypothesis in the image, with the red part acting as the landmark in the star model. The coloured ellipses show the mean location and variance of each part. (b) Samples from appearance density of model.

Cars

For the cars category, two separate models were trained, one for front and rear views and one for side views. The training and validation sets for cars consisted of 272 images, which when sorted into side and front/rear views gave only 130 images or so per viewpoint — insufficient to train an effective model, given their variable nature and the lack of supervision in our scheme. Therefore models were trained on the Caltech cars rear dataset and the UIUC cars side dataset, careful to ensure that no images were used that were present in either of the PASCAL test sets.

6 part star models were trained with a variety of feature types (the same repertoire as for motorbikes) and the final model selected according to its performance on the PASCAL training and validation sets. Standard model settings were used with $N = 30$ regions/feature-type/image. The models for each viewpoint are shown in Figure 8.11. The best cars rear model used a combination of Kadir & Brady regions and multi-scale Harris, while the cars side model used Kadir & Brady regions exclusively. As with the motorbikes, for each model, the mean

Figure 8.10: Examples of motorbikes from the PASCAL test set 2 being localized. The best hypothesis is indicated by the coloured circles and the background regions centres' by the magenta dots. The ground truth bounding box is shown in green. A correct putative box is shown in red while an incorrect one is shown in blue.

offsets in translation and scale between the putative bounding boxes and ground-truth boxes are computed from the PASCAL training data.

Combining the models of the two different aspects is problematic. If they both used the same feature-types they could be combined in a mixture model, however they use different sets of features. Our solution was to re-weight the likelihoods from each model, so that their mean on the PASCAL training data was the same. The weighting coefficients were then fixed for testing. Thus, given a novel image, the likelihood for each model was computed, multiplied by the respective weighting factor and then the bounding box picked by the model having the highest re-weighted likelihood. The ROC and RPC curves for the models of each view and their combination are shown in Figure 8.12, with localization examples on test set 2 shown in Figure 8.13. Looking at the curves in Figure 8.12, localization and classification performance

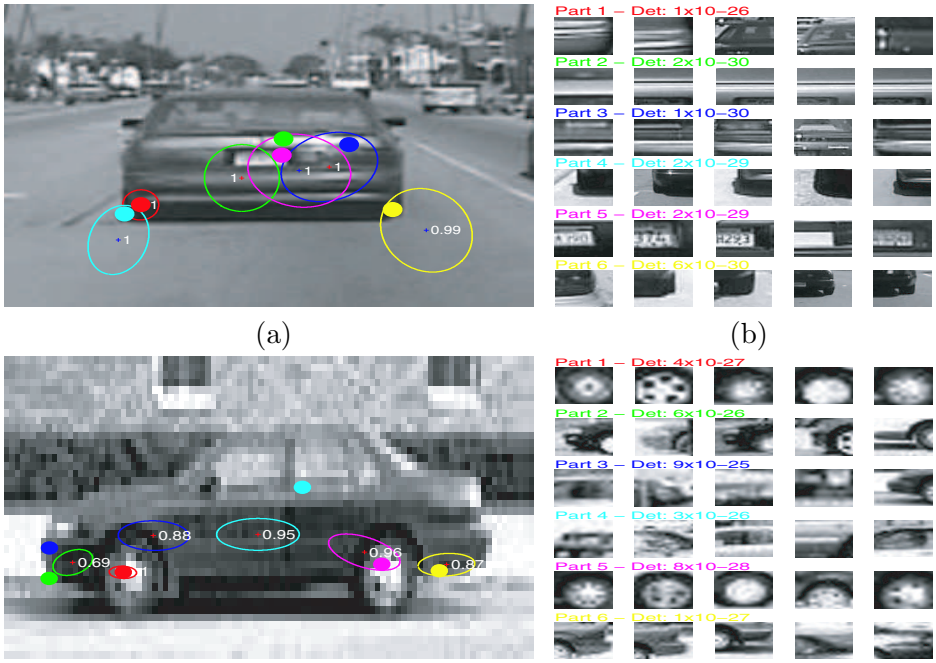


Figure 8.11: (a) Cars rear shape model, trained on Caltech cars rear dataset for evaluation on PASCAL data. The model is a heterogeneous combination of Kadir & Brady regions (red, green and blue parts) and multi-scale Harris regions (cyan, magenta and yellow). (b) Samples from appearance density of model. The model picks out the corners where the shadow of the car meets the road; the license plate and generic horizontal edge features on the rear of the car. (c) Car side model, trained on UIUC data for evaluation on PASCAL data. This model uses Kadir & Brady features exclusively. (d) Samples from the appearance density of the model. The model selects the front and rear bumpers as well as the wheels.

do not seem to be correlated. We return to this point in Chapter 10. On the harder test set 2, the combined model performs worse than the rear model in localization, the degradation due to the poor performing side model.

In Figures 8.15 and 8.16 we compare the models to other algorithms in both classification and localization. For test set 1 we use the combined model, while for test set 2 we use just the rear model since the features it uses (intersections of the cars' shadow with the road and generic horizontal edges) are also stable across a range of intermediate three-quarter views. Strictly speaking, we cannot compare our approach to the others since the models were not trained on the PASCAL training data, however the training images we use confer no particular advantage other than there being more of them. The other approaches require more supervision than ours therefore can be trained on fewer images. Additionally, our approach constitutes an open-world test scenario, thus demonstrates the generalization ability of our models. On test set 1 the

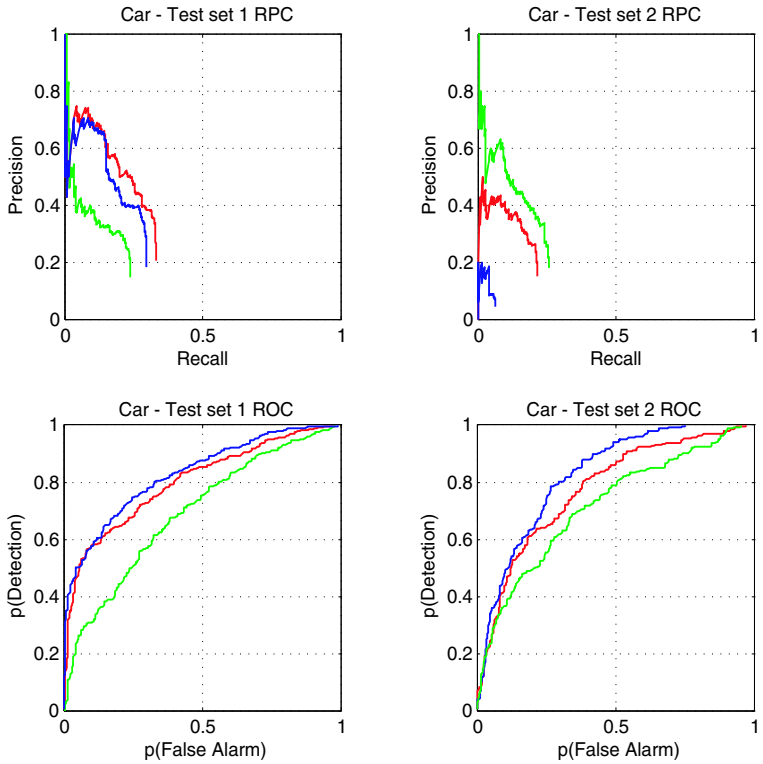


Figure 8.12: Recall-precision curves (1st row) and ROC curves (2nd row) for the models in Figure 8.11 evaluated on the PASCAL Cars test set 1 (1st column) and test set 2 (2nd column). The side model is shown in blue; the rear model in green and the combination of the two models in red. The side model outperforms the others in classification on both test sets. The combined model outperforms each individual model in localization on test set 1, but on test set 2 performs worse than the rear model due to the poor performance of the side model.

model performs relatively poorly compared to the others, although it easily beats the baselines. On test set 2 however, the performance of in localization is comparable to the system of Leibe and Schiele [65], despite requiring less supervision.

8.2.2 TSI-pLSA

We apply all 3 pLSA-based methods to the PASCAL data. The validation sets are merged with the training sets for each category to boost the number of training images. As with the Caltech datasets, learning was performed in an unsupervised manner, with foreground and background training (and validation) examples being presented to the algorithms as one corpus, containing 4 distinct objects (motorbike, car, people and bicycle). A variety of settings were tried for the number of topics: $Z = \{4, 5, 6\}$, the motivation being that some classes are best modelled by more than one topic owing to their variability.

Figure 8.13: Localization examples of the rear aspect Constellation Model on test set 2 of PASCAL Cars. The best hypothesis is indicated by the coloured circles and the background region's centres by the magenta dots. The ground truth bounding box is shown in green. A correct putative box is shown in red while an incorrect one is shown in blue.

The classifier for use in testing was either a single topic or equally-weighted pair of topics, selected by their performance on the training and validation sets. More formally, for a test image t, the classifier used is $P(d^t|z^*)$ in the case of a single topic or $\frac{P(d^t|z_1^*)+P(d^t|z_2^*)}{2}$ in the case of two topics, z^* or z_1^* and z_2^* being the chosen topic(s).

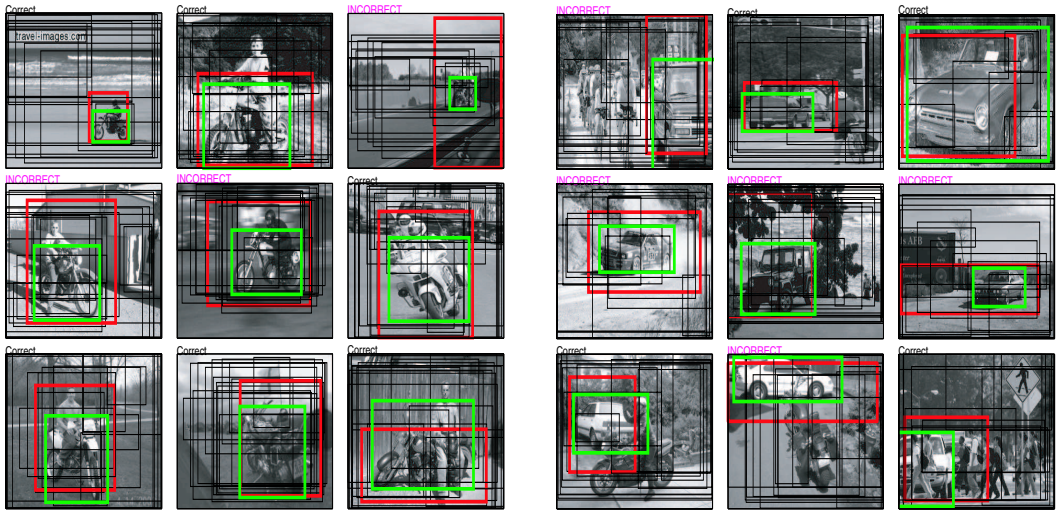
The classification results shown in Table 8.1 use a 5 topic car model and a 6 topic motorbike model for all three methods. For test set 1 of both classes, the addition of location drops the error rates significantly. However, on the more challenging test set 2, TSI-pLSA outperforms ABS-pLSA, demonstrating limitations of using absolute position on challenging data. Localisation examples on test set 2 are shown for both classes in Figure 8.14.

8.2.3 Comparison to other methods

The ROC and RPC curves obtained by the the Constellation Model and TSI-pLSA are plotted against other entrants to the PASCAL Challenge in Figures 8.15 – 8.18. Overall both methods perform well, comfortably beating all baseline methods and on occasion rivalling the best

Dataset	pLSA	ABS	TSI	CM
Car 1	21.8	12.4	15.3	25.4
Motorbike 1	19.4	8.3	9.8	9.7
Car 2	31.7	33.0	25.8	25.1
Motorbike 2	33.7	30.2	25.7	27.7

Table 8.1: The results of the pLSA-based methods on the PASCAL datasets for classification. Values given are error rate at point of equal error on an ROC curve. TSI-pLSA significantly outperforms both ABS-pLSA and pLSA. The increased difficulty of test set 2 is demonstrated by the increased error rates over test set 1. The Constellation Model performs comparably to the pLSA methods, except for Car test set 1 where it is significantly worse.



(a) (b)

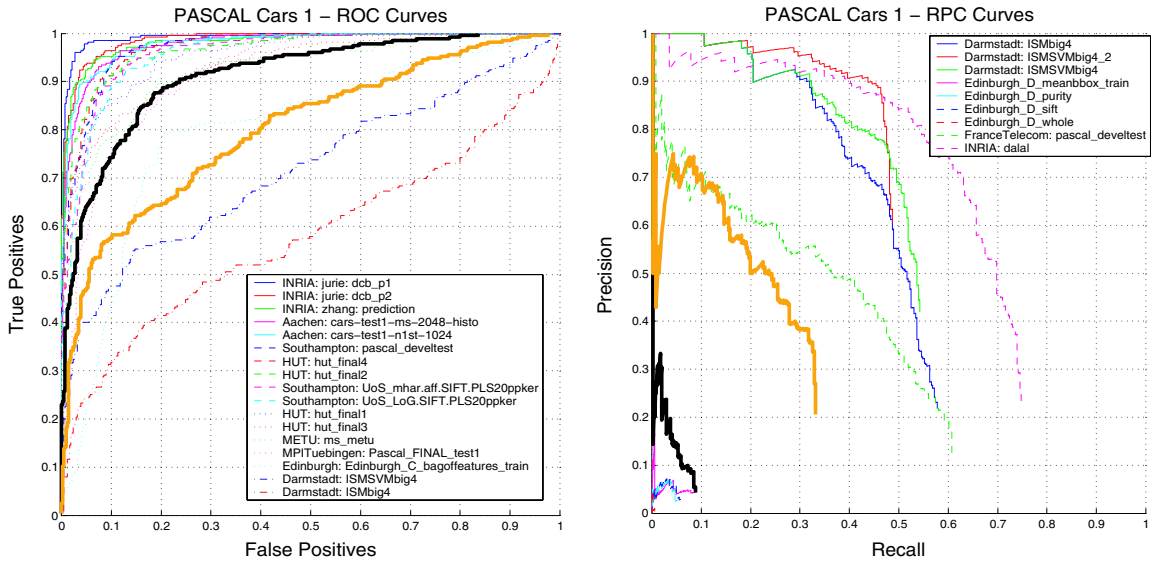
Figure 8.14: A mix of correct and incorrect localization examples for TSI-pLSA running on test set 2 of (a) cars and (b) motorbikes. The putative bounding box is shown in red and the ground-truth box in green. The thin black rectangles show other bounding boxes proposed but not picked by the model.

methods. The only poor performance is on test set 1 for cars where both methods lag behind rival approaches for both classification and localization. Compared to other methods, both the Constellation Model and TSI-pLSA seem to be more competitive on the more challenging test set 2, where the Constellation Model rivals one of the leading approaches, that of Leibe and Schiele [65]. Although TSI-pLSA performs poorly in the localization task compared to the Constellation Model and other leading approaches, its classification performance is more competitive. On test set 2 in classification, TSI-pLSA is the best method on cars and 4th overall on motorbikes.

The challenging nature of the datasets is illustrated by the fact that on test set 2 of both

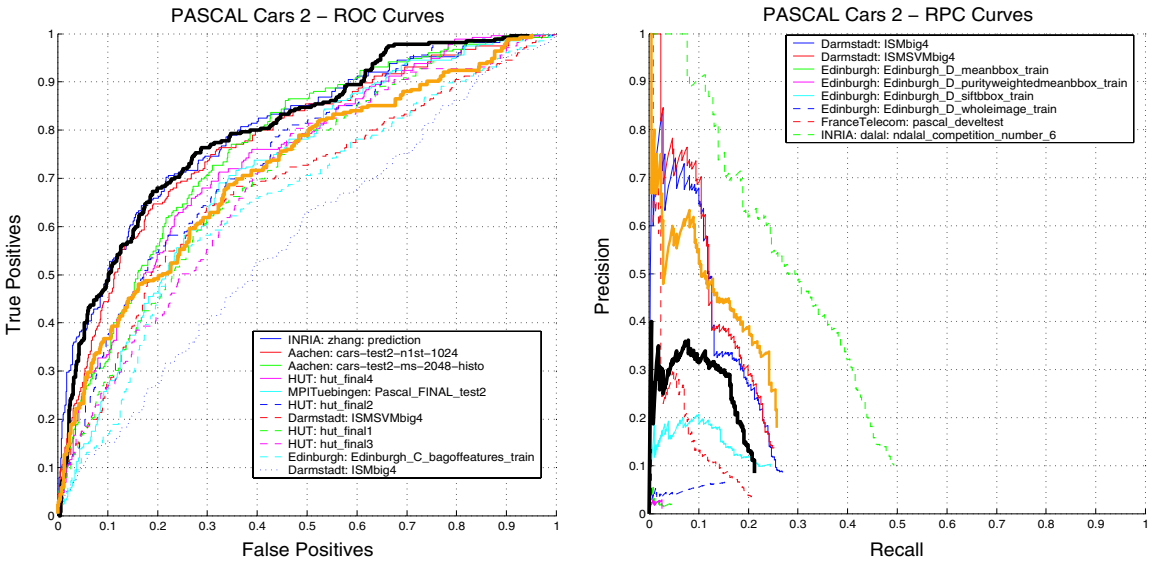
classes, the best classification error was 20% and the best localization performance was only 60% recall.

It is interesting to note that both the Constellation Model and the Leibe and Schiele approach, which are among the best performing methods in localization, perform poorly in classification. While no other approaches are evaluated in both tasks, the suspicison is that the leading approaches in classification are using more than just the statistics of the object to achieve their performance. This phenomenon is also observed in Section 7.6.



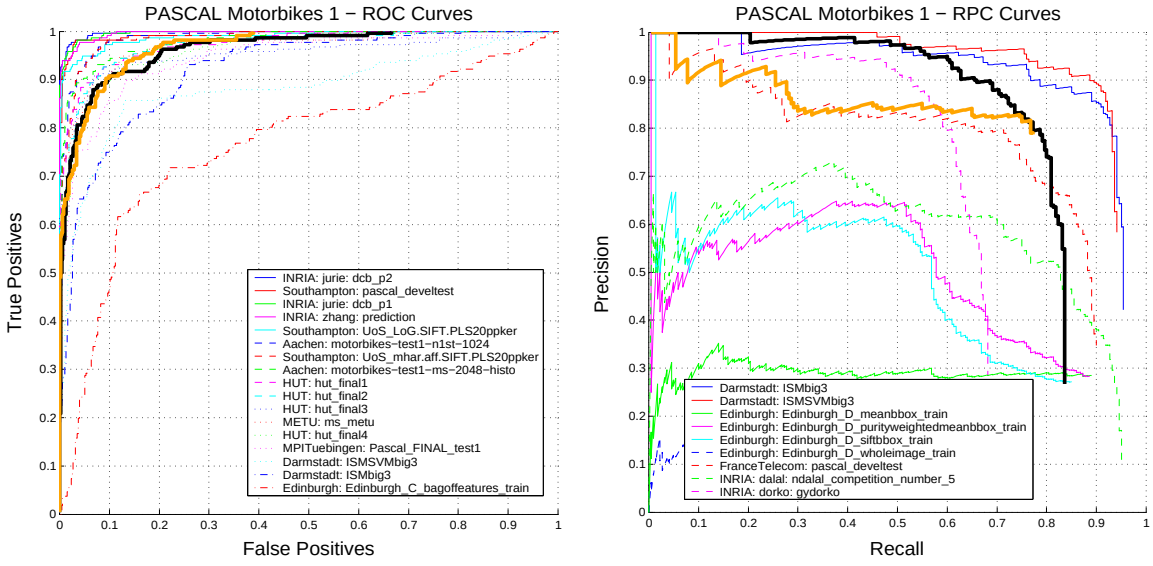
(a) (b)

Figure 8.15: (a) ROC and (b) RPC curves for the Constellation Model (thick orange curve) and TSI-pLSA (thick black curve) compared to other entrants to the PASCAL Challenge on test set 1 for Cars. Both approaches perform poorly being beaten by many methods, although they are superior to the baselines.



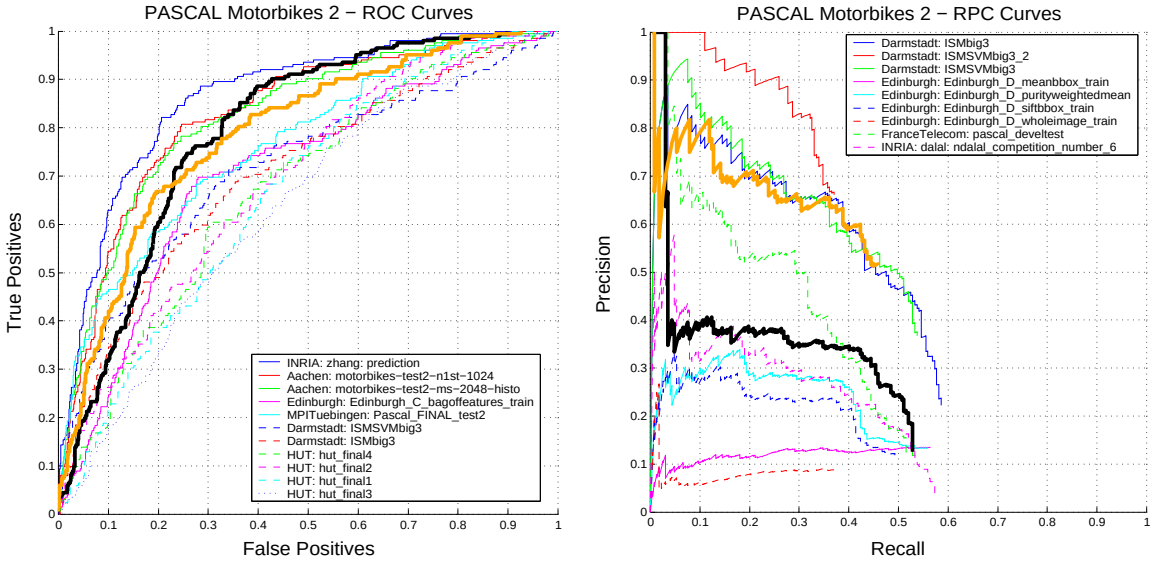
(a) (b)

Figure 8.16: (a) ROC and (b) RPC curves for the Constellation Model (thick orange curve) and TSI-pLSA (thick black curve) compared to other entrants to the PASCAL Challenge on the more challenging test set 2 for Cars. These plots tell a different story to that of Figure 8.15. TSI-pLSA has the leading classification performance across all methods, while the Constellation Model is comparable to leading methods.



(a) (b)

Figure 8.17: (a) ROC and (b) RPC curves for the Constellation Model (thick orange curve) and TSI-pLSA (thick black curve) compared to other entrants to the PASCAL Challenge on the more challenging test set 1 for Motorbikes. While the classification performance is middle-of-the-road, the localization performance of both methods is respectable, especially for TSI-pLSA which is only beaten the Leibe and Schiele's scheme.



(a) (b)

Figure 8.18: (a) ROC and (b) RPC curves for the Constellation Model (thick orange curve) and TSI-pLSA (thick black curve) compared to other entrants to the PASCAL Challenge on the more challenging test set 2 for Motorbikes. Both methods give a respectable ROC performance. The localization performance of the Constellation Model is up with the best approaches. TSI-pLSA is significantly worse but ahead of the baseline methods.

Chapter 9

Contaminated data experiments

Thus far in the thesis, the training images have been labelled — we know they contain an instance of the object we wish to learn. However, for many real-world applications this may not be the case. For example, you might want to organise your home photo collection by learning visual models of common objects and places, without having the user manually annotate each frame. In this chapter, we address the problem of learning from *contaminated* data where a substantial portion of the training set images do not contain the object of interest, consisting instead of totally unrelated objects. From a machine learning perspective, the labels of the training data are considered to be noisy — while we expect them to be positive, there is a non-zero probability that they may be negative.

We apply both our methods to the output of Google's Image Search [46] which provides a large quantity of images for a given keyword. As described in Section 3.5, the images returned are highly polluted with the portion of good examples varying between 20% and 50%. Since images can be obtained for any given category, then if we can directly learn from such noisy contaminated data, it enables us to automatically learn a classifier for whatever visual category we wish, eliminating the need to manually gather large training sets as all current approaches do.

We investigate two different scenarios involving images from Google. In the first, we train models directly on Google images and use them to re-rank the training images, using the likelihood of each image under the model learnt. Hopefully the good images will score highly, while the junk (visually unrelated) images score poorly resulting the first few re-ranked pages

containing more good examples than they originally did, thus improving the performance of Google's Image search.

The second, more ambitious scenario, is to train models on the Google images and then to test them on the Caltech datasets, to see how they compare to existing methods, trained on manually prepared (uncontaminated) data. This open world evaluation requires stronger models than the first scenario since they will be used in a more general setting. We apply both scenarios to the Constellation Model and pLSA-based methods in turn.

9.1 pLSA-based methods

Although the original application of pLSA-based methods was to contaminated data in the text community, there are two issues to consider when training our models on images from Google: (i) the optimal number of topics, Z ; (ii) which subset of these topics should be used to form a classifier for use in testing. A larger number of topics will result in more homogeneous topics at the expense of their ability to generalize. Given the varied nature of images obtained from Google, a large number of topics might seem appropriate, but this raises the issue of how to pick the topics corresponding to the good images, while ignoring topics which model the junk images within the dataset. Hence the two issues above are linked.

Our solution to the latter problem is to use a validation set which is gathered automatically. We rely on the empirical observation (as seen in Figure 9.1) that the first few pages returned by Google tend to contain more good images than those returned later on. The idea is that we assume the images from these first pages are positive examples, and hence may be used as a validation set to make model selection choices in our experiments. The catch is that the drop off in quality of Google's search is so steep that only the first few images of the first page are likely to be good examples.

Using Google's automatic translation tool [47] we obtain the translations of the user's keyword in the following languages: German, French, Spanish, Italian, Portugese and Chinese. Since each translation returns a different set of images, albeit with the same drop off in quality, we automatically download the first few images from each different language, and combine to give a validation set of a reasonable size without a degradation in quality.

Using 7 different languages (including English), taking the first 5 images we can obtain a

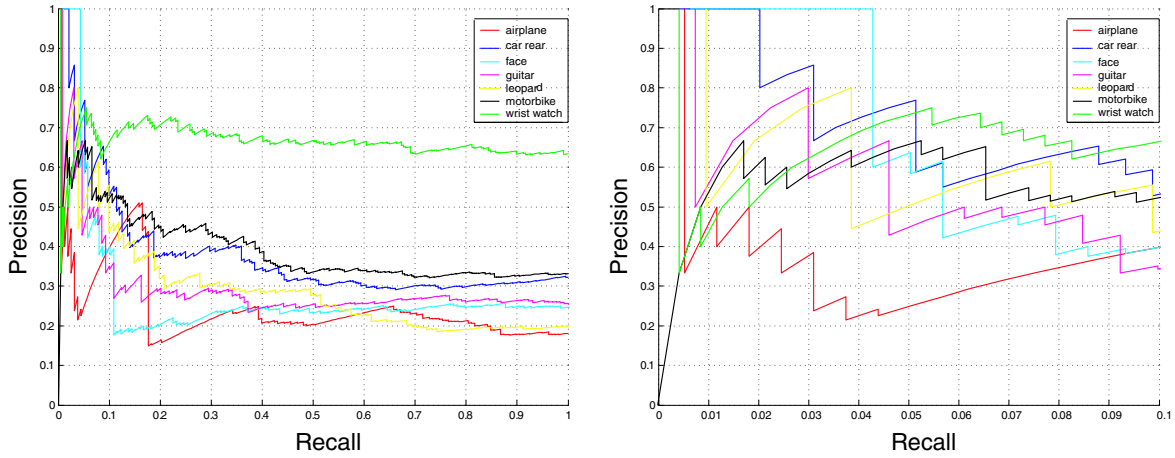


Figure 9.1: Recall precision curves of the raw output of Google's image search for the 7 keywords. Good labels count as positive examples while Intermediate and Junk labels are negative examples. (a) shows recall from 0 to 1, while (b) zooms in, showing the recall from 0 to 0.1. As each dataset is roughly 500 images in size, with the fraction of good images around 25% meaning that 0.1 recall corresponds to around 12 good images. Note the precision drops rapidly as the recall increases, levelling out at 20–30% for most categories.

validation set of up to 35 images (since languages may share the same word for a category and we reject duplicate images). Note that this scheme does not require any supervision. Figure 9.2 shows the validation sets for airplane and motorbike.

For a given keyword, the validation set can used to pick topics from a model. In view of the small size and imperfect quality of the validation set, we limit ourselves to picking a single topic from the larger model. While this is a sub-optimal strategy, it is all that can be reliably done given the lack of labelled data.

The number of topics to use in experiments was determined empirically. Figure 9.3 shows the test performance on the Caltech face and cars rear datasets as the number of topics was varied, when training from their respective Google datasets. The plot shows that beyond $Z = 10$, the validation set is unable to reliably pick the best topic and also that the performance of the best topic does not increase much. This is not surprising as with a large number of topics, the good images are likely to be modelled by several topics. In view of these issues, a stable peak is picked at $Z = 8$ (see Figure 9.3). The value is small enough for the validation set to reliably pick the correct topic, while being large enough to ensure that at least one component fits the good images (which maybe as low as 15%, roughly 1/8th of the training set). This value was

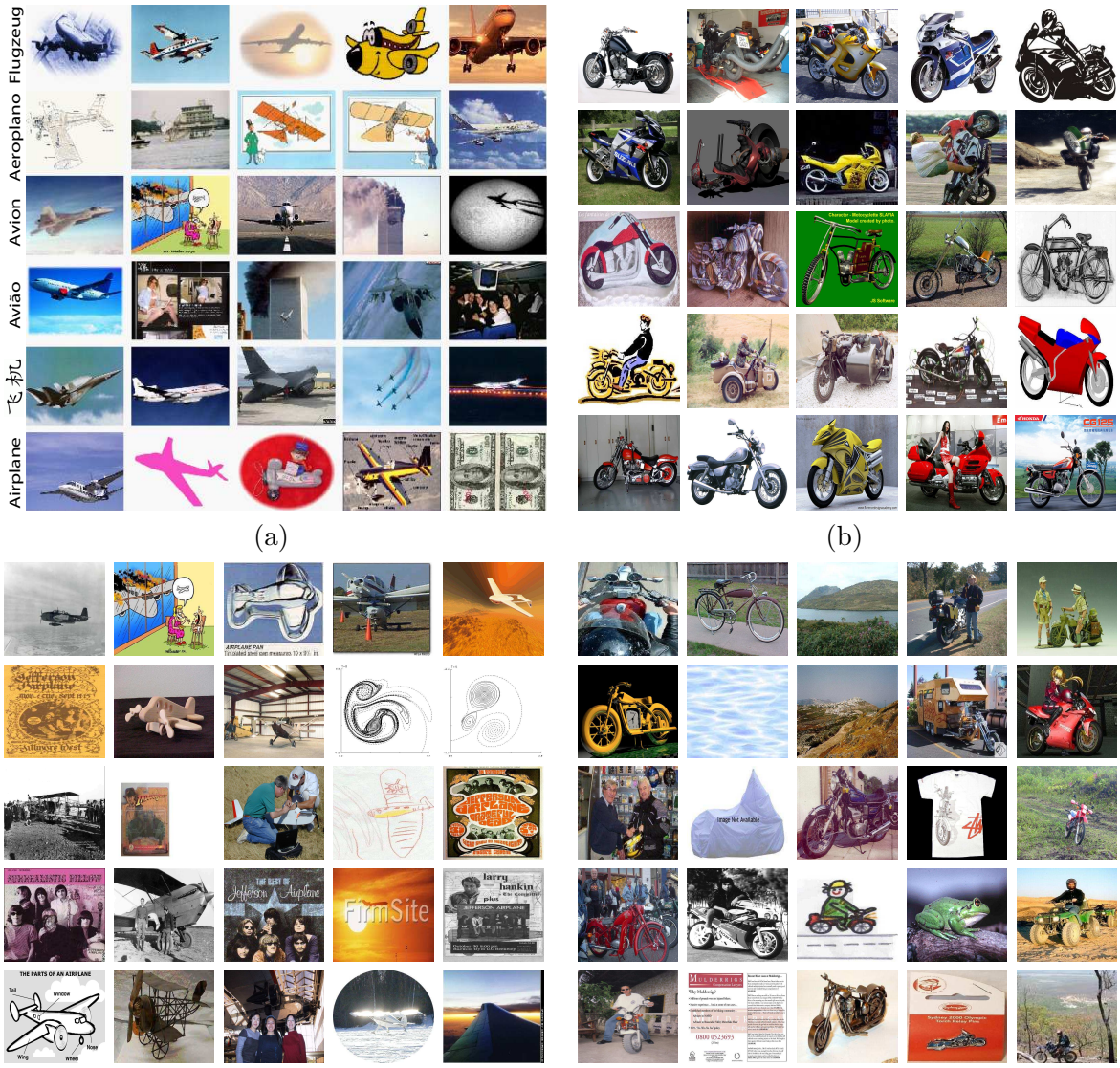
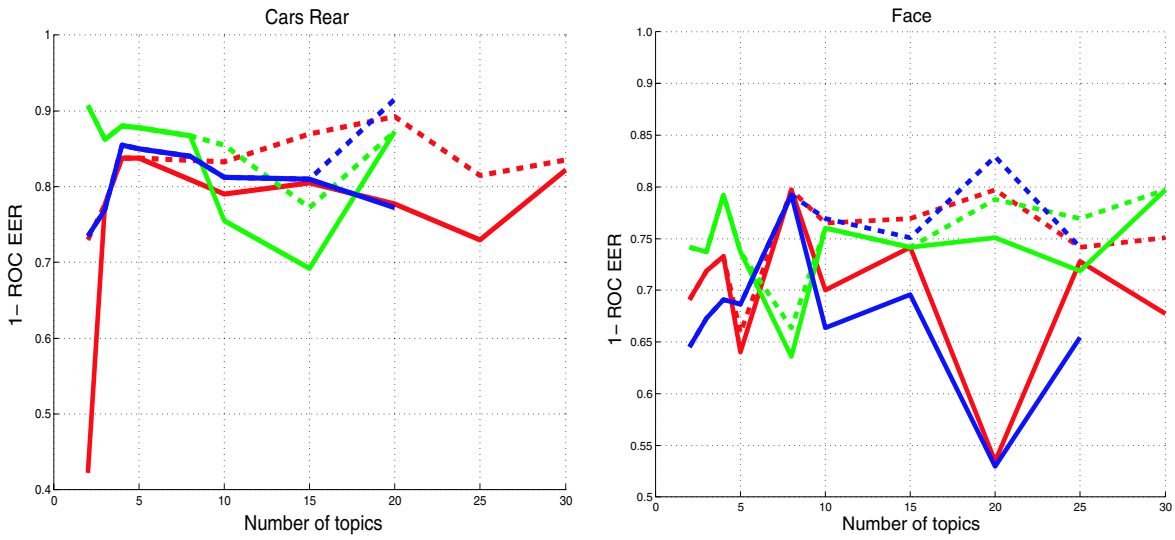


Figure 9.2: (a) The entire validation set for "airplane" obtained automatically using Google's translation tool and Google's image search. The text by each row shows the translated keyword used to gather that particular row. The quality of the images is noticeably higher the random sample of images from the dataset, shown in (c). (b) the entire validation set for motorbikes. Again, the portion of good images is considerably higher than a random page of images, shown in (d).

then used for all experiments involving Google data.

Having trained an 8 topic model, each topic is run across the validation set and the single topic z^* that performed best is picked to be the classifier used in testing (i.e. for test image t, its score is $p(d^t|z^*)$).



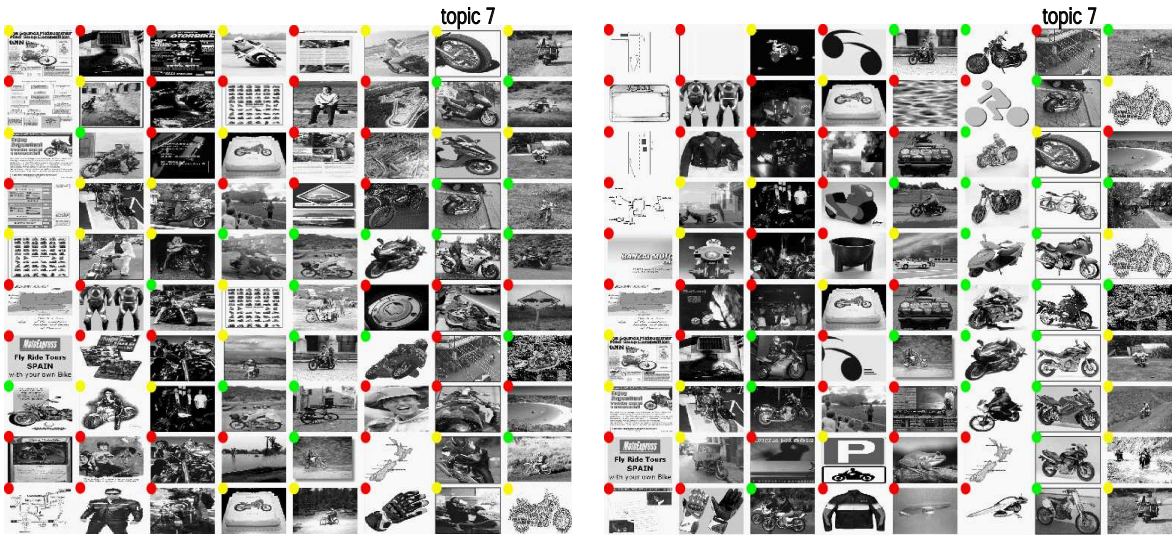
(a) (b)

Figure 9.3: How the number of topics affects the performance of models trained on Google data and tested on the appropriate Caltech test dataset for (a) cars rear and (b) face. Key: red - pLSA; green - ABS-pLSA; blue - TSI-pLSA. The solid curves show the performance of the single topic automatically chosen by the validation set, while the dashed lines show the performance of the best topic in the model. Thus, if the dashed and solid curves of the same colour are coincident, then the validation set has correctly picked the best topic. For < 10 topics, the dashed and solid curves are for the most part coincident.

9.1.1 Improving Google's image search

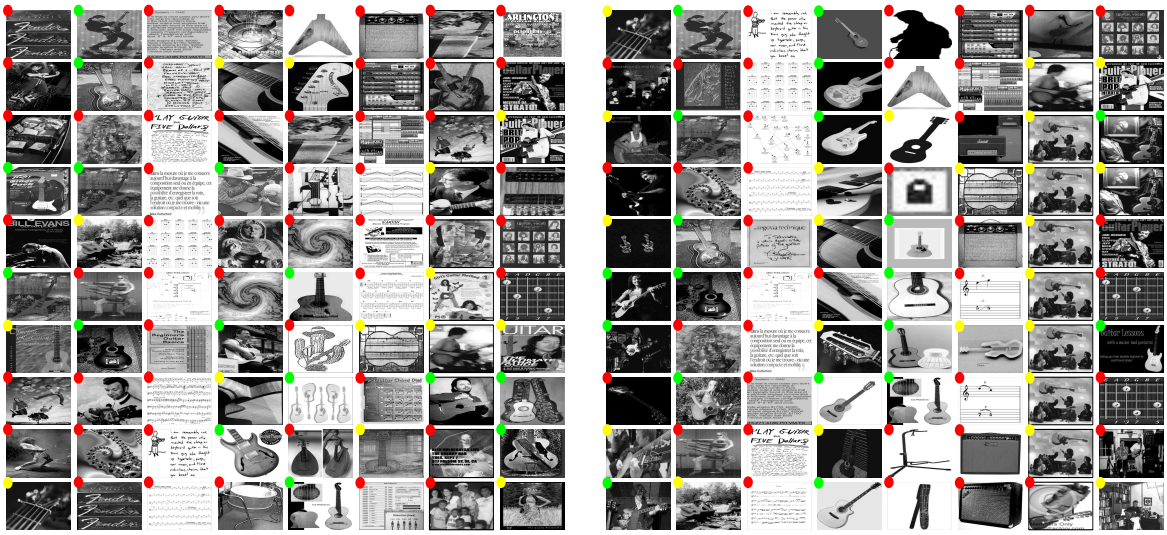
To improve Google's image search with the pLSA methods we first train a model on Google data, then choose a single topic with the validation set and use it to re-rank the entire Google dataset. The parameters and settings used are the same as in the Caltech experiments of Section 7.6.

In Figures 9.4–9.6, we can see what each topic in pLSA and TSI-pLSA has learnt from the motorbike, guitar and cars rear Google datasets. Note that each topic clusters visually consistent images: some correspond to the object; others to images of text or other junk. Some of the TSI-pLSA models seem to be clustering object instances by aspect (e.g. motorbikes) while one of the topics in the guitar model clusters the action of playing the guitar, rather than guitars. Qualitatively, the addition of location information seems to improve the consistency of each topic. The performance improvement is quantified in Table 9.1, where the precision at 15% recall (corresponding to a couple of web-pages) is recorded for the 3 pLSA methods and the raw Google ranking.



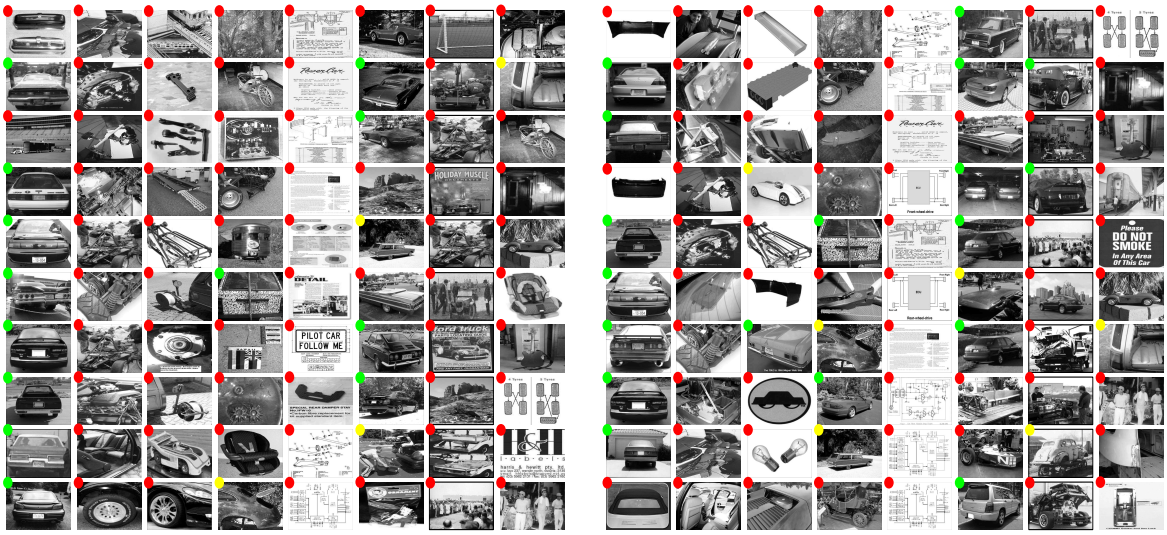
(a) (b)

Figure 9.4: (a) pLSA applied to images collected from Google using the "motorbike" keyword. Each column shows the top 10 images for a given topic. The coloured dots indicate the ground-truth label (only used for assessment purposes): green - good; yellow - intermediate; red - junk. The validation set automatically selects topic 7 as the best topic. (b) as for (a) but using TSI-pLSA instead of pLSA. Note the increased visual consistency, with different aspects of motorbikes being separated out into two different topics.



(a) (b)

Figure 9.5: (a) pLSA applied to images collected from Google using the "guitar" keyword. (b) as for (a) but using TSI-pLSA instead of pLSA. Topic 5 was chosen by the validation set for both pLSA and TSI-pLSA. In (b), topic 1 consists of people playing the guitar rather than guitars themselves, showing that images of actions can also be learnt from the Google data.



(a) (b)

Figure 9.6: (a) pLSA applied to images collected from Google using the "car rear" keywords. (b) as for (a) but using TSI-pLSA instead of pLSA. For both approaches, the first topic was automatically chosen by the validation set. In (b) note that topic 6 consists of cars at 45 degrees rather than rear on as in topic 1.

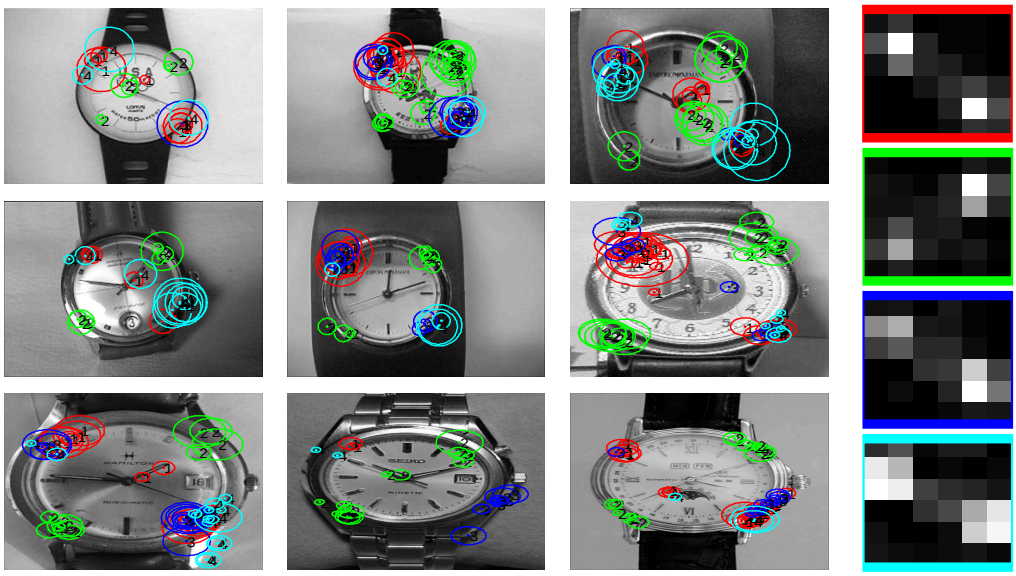
The table shows that TSI-pLSA always improves on the raw Google precision, in many cases significantly. The two other pLSA methods are more variable in their performance, outperforming TSI-pLSA for cars rear and wrist watch but performing no better or worse than the raw ranking for other categories (e.g. leopards).

Dataset	Raw Google	pLSA	ABS-pLSA	TSI-pLSA
Airplane	0.50	1.0	1.0	0.96
Cars Rear	0.41	0.88	0.94	0.76
Face	0.19	0.53	0.64	0.81
Guitar	0.31	0.39	0.40	0.49
Leopard	0.42	0.34	0.38	0.58
Motorbike	0.46	0.46	0.55	0.72
Wrist watch	0.70	0.83	0.97	0.87

Table 9.1: Precision at 15% recall for different methods of re-ranking images returned by Google's Images search. Good images are taken as positive examples, while intermediate and junk images for negative examples. The raw Google ranking is compard to the three pLSA-based methods. 15% recall corresponds to a couple of web pages worth of images.

9.1.2 Open world experiments

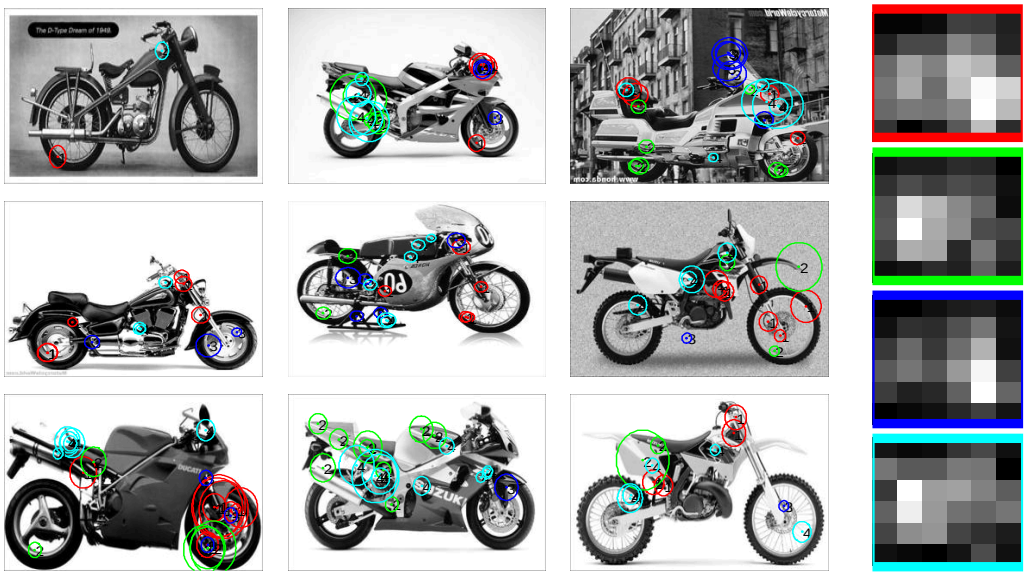
The models used to improve Google's image search may also be tested on some of the other datasets used in this thesis, enabling us to see what performance penalty is incurred when training on contaminated data. For such experiments, we use the Caltech datasets matching the keywords entered into Google's Image search. In Figure 9.7–Figure 9.9, we show TSI-pLSA models trained on Google data being tested on Caltech images. These are the same models used in Section 9.1.1. The location densities in the figures give some idea of the tightness of the model, which appears to be surprisingly good — better in some cases than the models trained directly on Caltech data.



(a) (b)

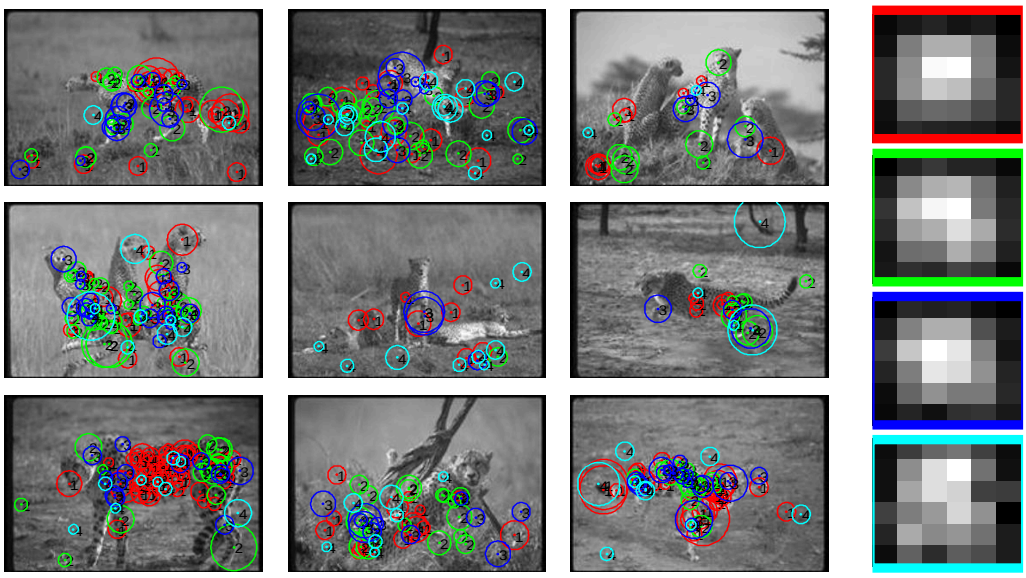
Figure 9.7: (a) Watches from the Caltech dataset, with regions superimposed that belong to the 4 most common visual words (irrespective of location) from the automatically chosen topic of the Google-trained TSI-pLSA watch model. Each colour shows regions quantized to a different visual word. The circular bezel of the watch face is picked out. Due to the rotation sensitivity of our region representation, different parts of the bezel are quantized to different words. (b) The location densities in the sub-window of the 4 most common words shown in (a). White corresponds to a high probability, black to a low one. Note their tightly constrained, multi-modal, nature.

Table 9.2 shows the classification performance of the pLSA-based methods when tested on the Caltech datasets (having been trained on Google data). For the most part, a significant drop in performance is observed when training from Google images. For around half the categories, the use of location information reduces the error significantly, although only in the case of motorbikes and airplanes is TSI-pLSA better their either of the other two approaches.



(a) (b)

Figure 9.8: As for Figure 9.7 but for motorbikes. The common words correspond to parts of the wheels of the bike and the exhaust/tail structure. The location densities look to be tighter than the model trained on Caltech data — see Figure 7.11



(a) (b)

Figure 9.9: As for Figure 9.7 but for leopards. The textured fur of the animal is captured by the most common regions. Their location densities are spread out, reflecting their diffuse nature.

Both ABS-pLSA and TSI-pLSA perform notably poorly on the guitar dataset. This may be explained by the fact that all the prepared data has the guitar in a vertical position while guitars appear at a seemingly random orientation in the Google training data. Since neither of the models using location can handle rotation they perform badly, in contrast to pLSA which

	pLSA	pLSA	ABS-pLSA	ABS-pLSA	TSI-pLSA	TSI-pLSA	CM
Category	Caltech	Google	Caltech	Google	Caltech	Google	Caltech
(A)irplane	17.7	24.7	13.2	17.2	4.7	15.5	6.3
(C)ars Rear	2.0	21.0	0.2	13.2	0.7	16.0	2.3
(F)ace	22.1	20.3	11.5	36.4	17.0	20.7	8.3
(G)uitar	9.3	17.6	10.0	62.0	14.4	31.8	6.3
(L)eopard	12.0	15.0	12.0	16.0	11.0	13.0	11.0
(M)otorbike	19.0	15.2	6.0	18.5	7.0	6.2	3.3
(W)rist watch	21.6	21.0	7.7	20.5	15.5	19.9	-

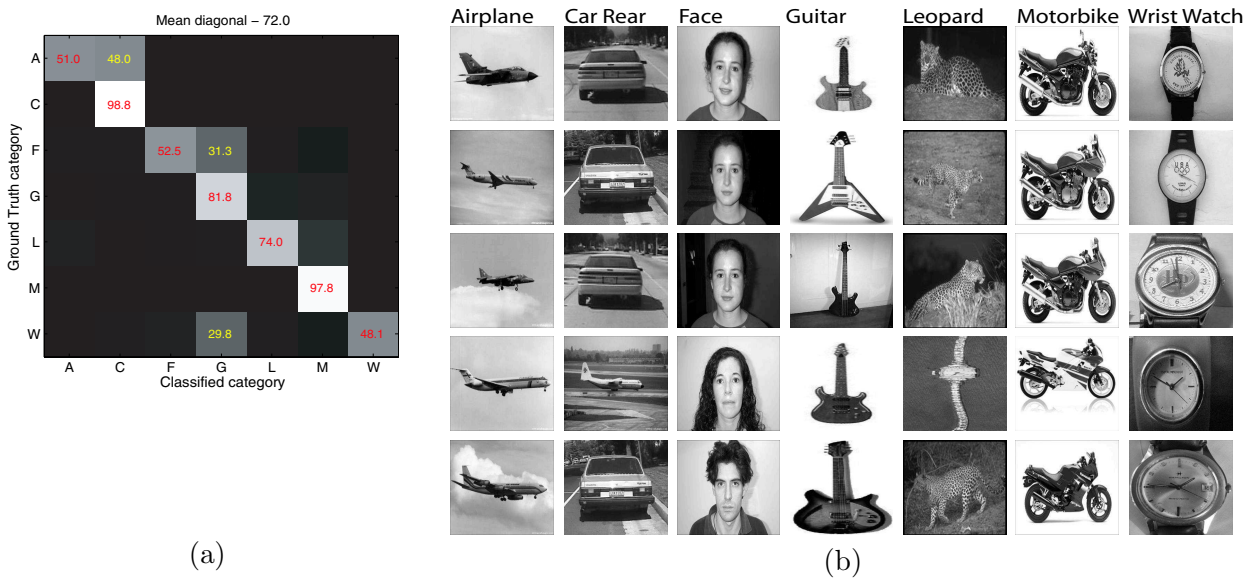
Table 9.2: Comparison of different methods trained on Caltech and raw Google data. All methods were tested on Caltech data. The task is classification, with the figures being the error rate at point of equal-error on an ROC curve. The error margins are roughly $+/-2\%$.

still performs respectably.

The confusion table of the 7 classes is shown in Figure 9.10(a). For the majority of classes the performance is respectable. Notable confusions include: airplanes being classified as cars rear (both have lots of horizontal edges); the guitar model misclassifying faces and wrist watches (due to the weak guitar model). Comparing to the confusion table for the Constellation Model (Table 6.2) and for TSI-pLSA (Figure 7.7) trained on Caltech data, once the extra two datasets have been eliminated, we see that the mean diagonal is 81.0%, comparable to the 83.1% of the Constellation Model but less than the 92.0% of TSI-pLSA trained on pure data. Figure 9.10(b) shows the top 5 images retrieved by each model from a set of 2148 images, comprising all images from all 7 classes.

9.2 Constellation Model

The Constellation Model is applied directly to the raw Google data using the star model with standard model settings, with a $P = 6$ part model and $N = 30$ detections/feature-type/image. Models using different combinations of feature-types are learnt, using a repertoire of Kadir & Brady, multi-scale Harris and Curves (using PCA-based representation) types. The best model is the one with one with the greatest area under a recall-precision curve, using the Google validation set (as described in Section 9.1) as positive examples and the Google background dataset (see Section 3.1) as negative examples.



(a) (b)

Figure 9.10: (a) Confusion table for 7 models, trained on Google data and test on the Caltech datasets. Each row is the ground-truth class, while the columns show the estimated class. (b) The top 5 images retrieved by each model from a corpus of 2148 images from all seven classes.

9.2.1 Improving Google's image search

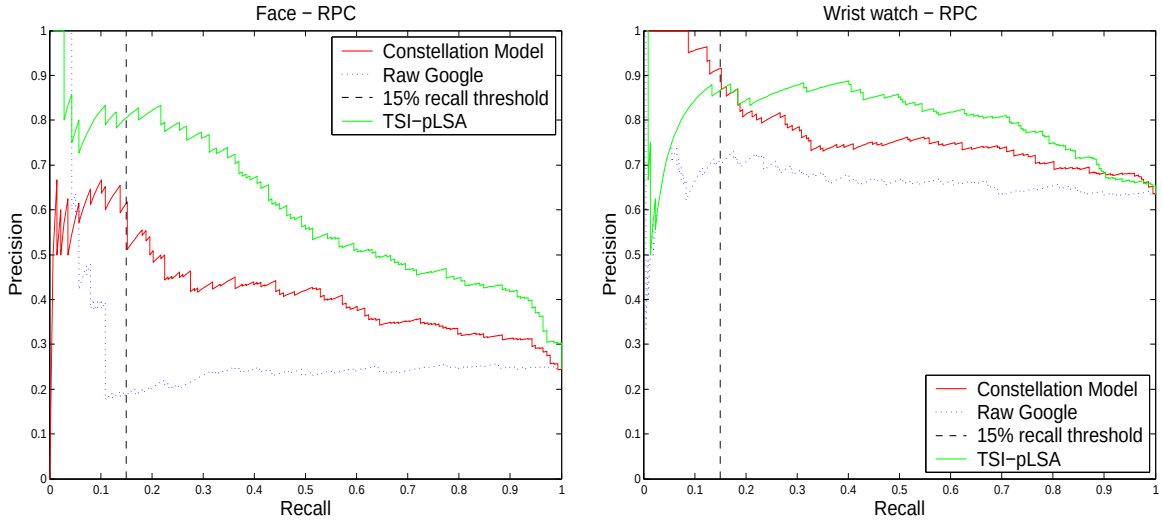
The model picked by the validation set may be used to re-rank the raw Google images in the hope of improving Google's Image search. Table 9.3 shows the performance of the best model over all feature-type combinations and the one picked using the validation set, at 15% recall. Also shown is the raw Google and TSI-pLSA performance at the same recall level. As in Table 9.1, good images are taken as positive labels, while intermediate and junk images are taken as negative labels. In the majority of cases, the performance of the Constellation Model is lower than that of TSI-pLSA, in some cases worse than the raw Google ranking.

In Figure 9.11 we show the recall-precision curves for the "face" and "wrist watch" keywords for the Constellation Model (using the automatically picked model) and TSI-pLSA, along with the raw Google ranking. Both methods can be seen to improve the overall quality of the search. It should be noted that for very low recall values, the precision is very unstable (since it depends on the labels of the just a few images), thus is not a reliable measure of quality.

In Figure 9.12 we show the face model learnt directly from raw Google data. The model shown is the one picked by the validation set and uses all three feature-types, having two parts of each type. Figure 9.13(a) shows the first 25 images in the raw Google ranking while Figure 9.13(b) shows first 25 after re-ranking by the model shown in Figure 9.12. The portion

Dataset	Raw Google	CM	CM best	TSI-pLSA
Airplane	0.50	0.39	0.39	0.96
Cars Rear	0.41	0.62	0.74	0.76
Face	0.19	0.62	0.62	0.81
Guitar	0.31	0.39	0.39	0.49
Leopard	0.42	0.23	0.34	0.58
Motorbike	0.46	0.52	0.62	0.72
Wrist watch	0.70	0.92	0.92	0.87

Table 9.3: Precision at 15% recall for different methods of re-ranking images returned by Google's Image search. Good images are taken as positive examples, while intermediate and junk images form the negative examples. The column headed "CM" is the performance of the Constellation Model automatically chosen using the validation set, while "CM best" shows the performance of the best model, across all feature-type combinations. The validation set appears to be moderately successful at picking the best model. The performance of TSI-pLSA is also given; it consistently beats the Constellation Model.

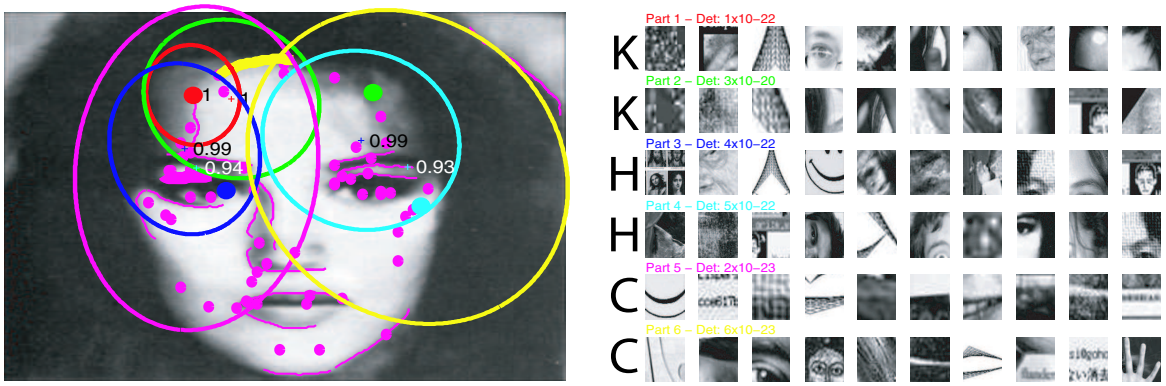


(a) (b)

Figure 9.11: Recall-precision curves for (a) "face" and (b) "wrist watch" keywords. Red — Constellation Model; green — TSI-pLSA; blue — raw Google ranking. The 15% recall threshold is shown with by the black vertical dashed line. (b) shows that even for cases where the raw Google performance is high, both the methods can still improve the quality of the search.

of good images is increased, although a couple of non-face images are still present.

In Figure 9.14 we show the wrist watch model learnt directly from raw Google data, picked by the validation set. The model uses 3 Kadir & Brady features and 3 Curve features. Figure 9.15(a) & (b) shows the shows the first 25 images before and after the application of the model in Figure 9.14. Only one non-watch image remains in the re-ranked images.



(a) (b)

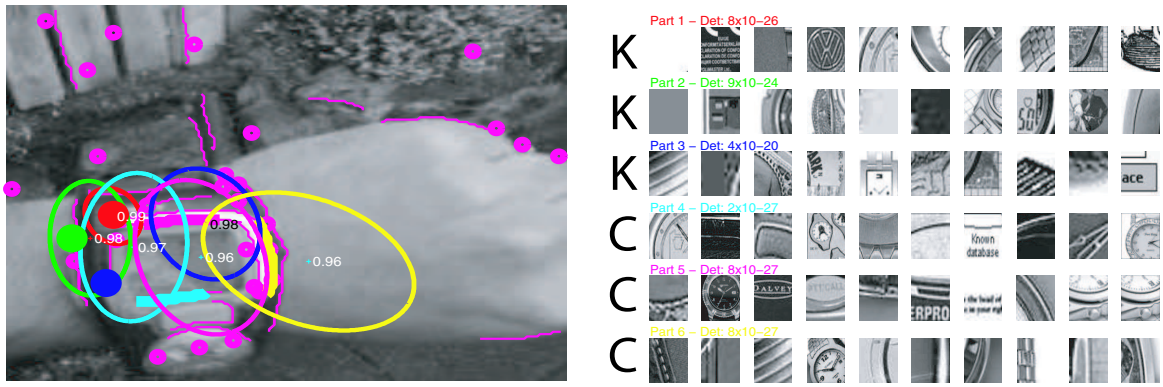
Figure 9.12: The face model learnt directly from raw Google data and automatically chosen by the validation set. The 6 part heterogeneous model used 2 Kadir & Brady features (K); 2 multi-scale Harris (H) and 2 Curves (C). (a) Shape model superimposed on an example image. The large coloured dots and thick curves show the best hypothesis in the image, with the background features shown as magenta dots and thin curves. (b) Samples from the appearance density. Note that the model is far looser than those trained on prepared datasets.

(a) (b)

Figure 9.13: (a) First 25 images returned by Google using the "face" keyword. (b) First 25 images after re-ranking by the Constellation Model trained on all images returned by the "Face" keyword. The coloured dot in the bottom right of each image denotes the ground truth label (green – good; yellow – intermediate and red – junk). Note the increase in good images in (b) as compared with (a).

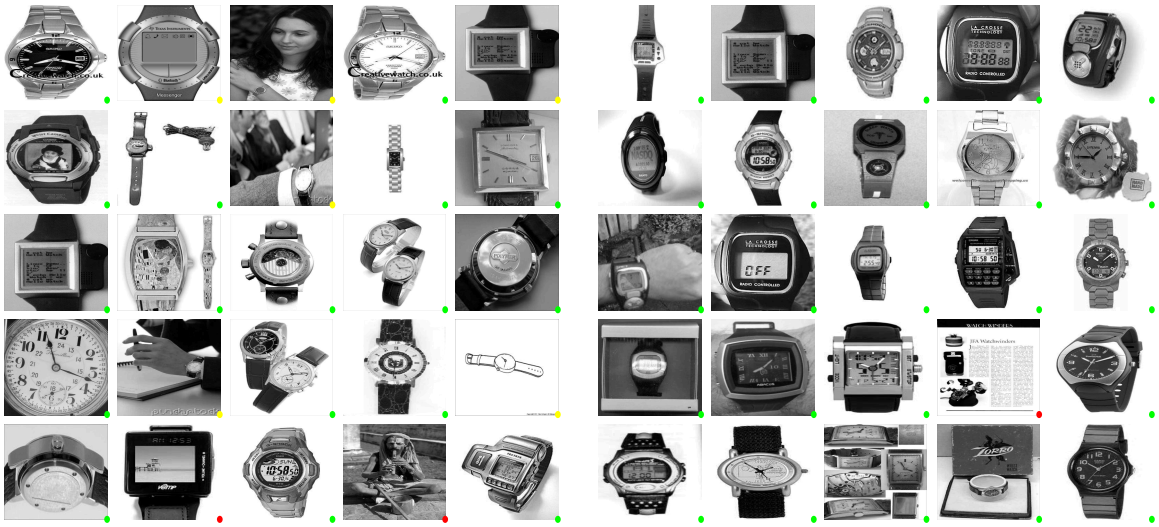
9.2.2 Open world experiments

The Constellation Models trained on raw Google data are applied to the Caltech datasets in a classification task. The results are shown in Table 9.4. The error rates are high compared



(a) (b)

Figure 9.14: The wrist watch model learnt directly from raw Google data and automatically chosen by the validation set. The 6 part heterogeneous model used 3 Kadir & Brady features (K) and 3 Curves (C). (a) Shape model superimposed on an example image. The large coloured dots and thick curves show the best hypothesis in the image, with the background features show as magenta dots and thin curves. (b) Samples from the appearance density. Although the model has a high variance in both location and appearance, it seems to pick out the bezel of the watch.



(a) (b)

Figure 9.15: (a) First 25 images returned by Google using the "wrist watch" keyword. (b) First 25 images after re-ranking by the Constellation Model trained on all images returned by the same keyword. The coloured dot in the bottom right of each image denotes the ground truth label (green – good; yellow – intermediate and red – junk). Note increase in good images in (b) as compared with (a).

to both TSI-pLSA and the Constellation Model trained on prepared data. This would indicate that the models are weak, only capable of loosely clustering of the images on which they were trained. These results are not surprising given the high variance of the models, examples of

	TSI-pLSA	TSI-pLSA	CM	CM
Category	Caltech	Google	Caltech	Google
Airplane	4.7	15.5	6.3	21.0
Cars Rear	0.7	16.0	2.3	31.8
Face	17.0	20.7	8.3	48.8
Guitar	14.4	31.8	6.3	58.3
Leopard	11.0	13.0	11.0	23.0
Motorbike	7.0	6.2	3.3	25.8
Wrist watch	15.5	19.9	-	31.7

Table 9.4: Comparison of different methods trained on Caltech and raw Google data. All methods were tested on Caltech data. The task is classification, with the figures being the error rate at point of equal-error on an ROC curve. The performance of the Constellation Model trained on Google data is poor, compared with Google-trained TSI-pLSA and Caltech-trained models. In some cases it is at, or worse than, chance level (e.g. face and guitar).

which are shown in Figure 9.12 and Figure 9.14.

9.3 Summary

The Google datasets present an extremely challenging environment within which to learn. It is therefore pleasing to see that both methods are able to find some consistency within the data, TSI-pLSA being successful and the Constellation Model less so. There are many possible variants on our testing paradigm, such as the inclusion of user-feedback, for example, which would improve the performance further.

The application of models trained on Google data in more general settings showed a difference in performance between the two methods. The models learnt with TSI-pLSA gave a respectable performance on the Caltech data, with the exception of a few classes. The Constellation Models trained on Google data did less well, in some cases performing worse than chance. However the TSI-pLSA results are encouraging, showing that useful models can be learnt using no supervision whatsoever.

Chapter 10

Discussion

In this thesis we have proposed two different approaches to object category recognition and evaluated them on a wide range of diverse datasets. Both methods showed the ability to learn categories with minimal levels of supervision and demonstrated performance competitive with leading methods in the field. We summarize the major contributions of the thesis:

- We have proposed a parts-and-structure model with probabilistic models of shape, appearance, scale and occlusion. It builds on the work of Weber *et al.* [109], adding in a probabilistic representation of appearance and is able to utilize directly the output of low-level feature detectors.
- The model is able to use as its input a combination of different feature types, combining them in one model. This enables the model to capture both the outline and textured interior of an object.
- We have devised efficient methods for learning the model and applying it in recognition to images with hundreds of features.
- We have proposed improvements to pLSA to enable the inclusion of location information in a scale and translation invariant manner.
- We have shown the TSI-pLSA model successfully learning from the raw output of an Internet Image search engine. The resulting models are robust enough to perform well on standard test sets.

The Constellation Model and TSI-pLSA are two complementary methods, reflected by the differing levels of performance on the various test sets. We examine the performance of each method in turn, proposing areas for future investigation before comparing them directly and finally highlighting some issues common to both.

10.1 Constellation Model

The ability to learn a parts and structure model automatically, only being provided with image labels, is encouraging. Although the output of the feature detectors is erratic, the model learns which parts of the object are consistently picked out by the detector. However, when training from highly variable data, a large number of images may be required for the model to find these stable parts.

Our efforts to boost the models' coverage of the object, and hence performance, had mixed results. The use of a heterogeneous set of features worked well, beating models with homogeneous features on many of the datasets. The use of the star model in learning, while enabling greater number of parts and regions and reduced learning times, did not deliver models with a significantly improved performance. Two possible reasons for this are: (i) the additional detections did not identify any further stable parts of the object and (ii) with many parts in the model, the learning scheme gets stuck in local maxima, thus is unable to make full use of the additional regions identified. In recognition however, the ability of the star model to use several hundred detections enables small objects to be successfully localized.

The current manner in which the Constellation Model localizes is somewhat fragile. A bounding box around the best hypothesis is taken to give the putative location and scale of the object in the image, but if a single part falls off the object, as illustrated in Figure 10.1, the bounding box is often dramatically distorted resulting in a failed localization.

The model currently lacks any kind of verification stage where support for the proposed model pose is checked against the image in recognition. Leibe and Schiele have such a scheme which measures the pixel level fit of the model, resulting in a significant decrease in false alarm rate. Introducing such a scheme might help avoid some of the errors illustrated in Figure 10.1.

The utility of the star model in learning is limited by the necessity of using ordering constraints on the features within hypotheses to ensure good convergence properties. The current

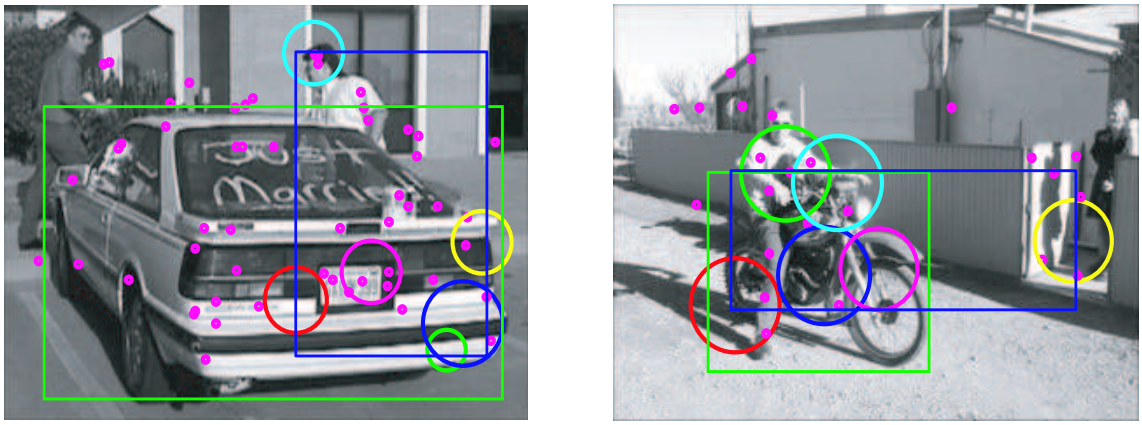


Figure 10.1: Two examples of incorrect localizations by the constellation model. The ground-truth bounding box is in green, while the bounding box of the best hypothesis is shown in blue. The two examples show cases where the best hypothesis has all parts but one on the object, with a single part falling off the object. In the example on the left, this is due to a weak overall fit of the model, while on the right this is due to no regions being present at the front on the bike, thus a region far away is used. In both cases the single off-object part causes a large deformation of the bounding box, resulting in an incorrect localization.

scheme requires the x (or y) coordinates of all non-landmark parts to be monotonically increasing, an $O(N^P)$ requirement. An avenue of further investigation would be to impose weaker constraints, for example constraining the ordering of just neighbouring pairs of parts, resulting in a more reasonable $O(N^3P)$ requirement. Alternatively, instead of imposing such hard constraints, a maximum-a-posteriori model could be learnt using a prior which encourages the separation of the parts.

The star model structure is only one form of simplification of the full model. While it has wide applicability, it should be noted that it does not model certain classes, such as humans or animals, effectively as it fails to capture the articulated nature of the body. Categories such as these are well represented by tree-structured models [35], having the same computational complexity as the star model. It would be interesting to try out different model structures in learning, perhaps selecting the most appropriate with a validation set.

The model proposed in this thesis makes no use of priors or direct supervisory information in learning, both of which would assist the process. A hierarchal Bayesian form of the model, allowing prior densities over the model parameters, was investigated by Fei-Fei *et al.* [32]. By using a prior constructed from many object categories, the ability to learn a new class from a handful of images (≤ 5) was demonstrated. More direct supervision by either manually

segmenting the images or specifying points of correspondence has not been investigated but can easily be incorporated into the learning scheme. The benefits of star models with a large number of parts might be realized with the addition of such supervision.

A further refinement to the Constellation Model is to avoid the use of features in recognition altogether. Relying on unbiased, crude region operators in learning is a necessary evil if we wish to learn without supervision: we have no prior knowledge of what may or may not be informative in the image but we need a sparse representation to reduce the complexity of the image sufficiently for the model learning to pick out consistent structure. However in recognition, the situation is different. Having learnt a model, the appearance densities model the regions of the image we wish to find. These densities can then be run over the image to obtain likelihood maps for each part at all locations and scales in the image. Then, the efficient methods of Felzenszwalb and Huttenlocher [35] allow the location density of a star model to be applied, thus finding the globally optimal match of the model over the entire image. Such a complete search overcomes many false negatives due to feature drop out, and also poor localizations due to small feature displacement and scale errors. Preliminary results of such an approach are presented in [39].

A number of approaches related to the Constellation Model have recently been published. Papers such as Holub and Perona [53] and [6, 7] have combined generative parts and structure approaches with discriminative training techniques. These hybrid schemes have the advantage of generative models that the learnt model has intuitive meaning (i.e. it looks like the object) whilst having the low error rates associated with discriminative methods, demonstrated by excellent performances on the Caltech datasets [37].

10.2 TSI-pLSA

The ability to learn from contaminated data is the main strength of the pLSA-based approaches. The models learnt from Google data show that useful classifiers can be constructed with no supervision. Although the models are weaker than approaches trained using some form of supervision, they could be used as priors in training supervised models from a small number of images.

The results on the PASCAL datasets show that TSI-pLSA can localize with moderate success

on data with significant scale and location variation, in contrast to the ABS-pLSA model. However, both methods are inferior to the Constellation Model on more challenging data such as the PASCAL test set 2 and the Fawlty Towers data. In cases where the object is very small, the large number of regions gathered from the background of the image will obscure the contribution from the object itself.

An inelegant feature of TSI-pLSA is the mechanism by which the bounding boxes are proposed since it makes no use of the location density of the model, relying entirely on appearance. Additionally, the scheme is only likely to work if the regions falling on the object belong to one consistent topic. What is needed is more complicated hierarchical model, such as the one proposed by Sudderth *et al.* [102], which infers the location of the object using both location and appearance information.

10.3 Constellation Model and TSI-pLSA comparison

The two approaches are quite different in nature: the Constellation Model uses a small set of parts; their location and appearance being described by continuous densities. In contrast, TSI-pLSA models with equal emphasis the entire sub-window of the reference frame, using discrete representations of location and appearance. However, some properties are common to both: they model the output of low-level feature detectors in a generative fashion.

The Constellation Model performs well in challenging localization tasks using the Fawlty Towers and PASCAL datasets. Owing to the small scale of the objects, only a limited amount of evidence for the object can be extracted from the image. In scenarios such as these, the Constellation Model remains effective, provided a sufficiently large number of regions are used to reliably pick out the small objects in the image, since its modelling power is concentrated on a few parts rather than spread out over the image as with the pLSA-based methods.

Several factors limit the localization performance of TSI-pLSA. First, the scheme for proposing the reference frame of the objects has shortcomings, as outlined in Section 10.2, meaning that it is difficult to find the object if it is small within the frame. Second, the approach is prone to learning statistics of the image instead of those of the object, making it hard to separate the object from the background. However, this may have a beneficial effect on classification performance since the model can make use of information from the surrounding scene, in addition to

the object of interest.

In contaminated data, the Constellation Model is unable to learn effectively, only giving a loose clustering of the data. It is hindered by only having one model component, in contrast to the multiple topics of the TSI-pLSA model. As the Constellation Model must account for all images in the training data, the resulting densities have a large variance and consequently little discriminatory power. In TSI-pLSA however, each topic need only model a consistent subset of the images, meaning it can learn robust object models even in heavily polluted data.

10.4 Common issues

One important issue, barely touched on in this thesis, is that of 3-D recognition; both the methods presented essentially being single viewpoint approaches. However, for both approaches there are clear ways to extend them to multiple views. The simplest approach is to assume independence between the views and to use a mixture of models. For the pLSA-based methods which are already mixture models, this means that each topic represents a different aspect of the object. Some of the TSI-pLSA models trained on Google (e.g. Figure 9.4 & Figure 9.6) show this occurring. Mixtures of Constellation Models were used by Weber *et al.* [109] for multi-view human head detection and the extension to mixtures of our incarnation of the Constellation Model is presented in Section 4.8. Of more interest however would be variants that exploit the redundancy between views [104]. One possible approach is to move to a 3-D location model, with several landmark parts used in each hypothesis to propose unambiguous transformations between the model and the scene.

When using generative models, the background around the object can be learnt as well as the object itself, as observed in Section 7.6. Such a phenomenon is likely to increase the classification performance, provided the positive images have a similar background in both test and training sets and that this background has different statistics to the negative images. However, it also means that it is difficult to localize the object. The use of discriminative training methods and diverse training data may help to avoid this problem.

In this thesis, each image is represented as a collection of regions picked out by low-level operators. While the tractability of the two approaches proposed in this thesis is dependent on a sparse representation of the image, much information from the image is being thrown away.

If the models are to offer a complete representation of the image they must account for every pixel. Recent work such as Kumar *et al.* [60] and Winn and Joijic [113] moves in this direction, combining recognition with segmentation.

On this note, we feel that the future of object recognition lies in probabilistic representations which model the image at a variety of levels: scene, object, part, right down to the pixels themselves. Although such representations are generative in nature, superior performance may be obtained by training them in a discriminative manner. The schemes must also provide a natural way of modelling the multiple views of an object, going beyond the single viewpoint paradigm that is currently in vogue.

Appendix A

Appendix

We list here all the notation used in the pLSA-based approaches (Table A.1) and the Constellation Model (Table A.2).

Variable	Size	Explanation
c	4	Latent variable specifying bounding box of sub-window in image.
C	1	Total number of proposed bounding boxes per image, equal to $K(K+1)/2$.
d	1	Image index.
D	1	Total number of images.
K	1	Maximum number of components in Gaussian mixture model used to propose bounding boxes.
N_d	Variable	Number of regions in image d.
w	1	Visual word index.
W	1	Size of visual word vocabulary.
x	1	Spatial location bin index.
x_{fg}	1	Spatial bin index within the sub-window containing the object.
x_{bg}	1	The background spatial bin.
X	1	Total number of spatial bins in model ($X_{fg}+1$).
X_{fg}	1	Total number of spatial bins within the sub-window.
z	1	Topic index.
Z	1	Total number of topics in model.

Table A.1: Overview of the notation used pLSA-based methods.

Variable	Size	Explanation
α^i	1	Area of image i.
a	1	Dimensionality of PCA space for descriptors.
$\mathbf{b}$	P	Binary vector indicating presence/absence of each model part.
$\mathbf{B}$	2^P or $P-1$	Parameter of occlusion density. Size depends on shape model being either full or star.
$\mathbf{c}_p$	a	Model parameter. Mean of descriptors of model part p.
$\mathbf{c}_{t,bg}$	a	Fixed parameter. Mean of background descriptors for feature type t.
$\mathbf{d}$	a	Descriptors of a single feature.
$\mathbf{D}^i$	T	Descriptors of all features in image i. Each element is $k \times N_t^i$.
$\mathbf{D}$	$T \times I$	Descriptors of all features all images. Each element is $k \times N_t^i$.
$\mathbf{f}$	T	Number of parts of each type present in a given hypothesis.
$\mathbf{h}$	P	Index variable allocating features to model parts.
H^i	$\prod_t^{N_t^i} C_P$	Set of all valid hypotheses in image i.
i	1	Index for images.
I	1	Total number of images.
k	1	Size of pixel patch in PCA space.
l	1	Landmark part index.
$\mathbf{M}$	T	Model parameter. Vector of means for Poisson background distribution.
$\boldsymbol{\mu}_l$	$2(P-1)$	Model parameter. Mean of locations of model parts.
$\mathbf{n}$	T	Number of background features of each type for a given hypothesis.
$\mathbf{N}^i$	T	Vector of total number of features of each type within image i.
p	1	Index for model part.
P	1	Number of parts in model.
r	1	Scale range of background distribution.
R	1	Ratio of class posterior densities.
s	1	Scale of a single feature.
$\mathbf{S}^i$	T	Scales of all features in image i. Each element is $1 \times N_t^i$.
$\mathbf{S}$	$T \times I$	Scales of all features in all images. Each element is $1 \times N_t^i$.
Σ_S	$P \times P$	Model parameter. Covariance matrix of scales of model parts.
Σ_l	$2(P-1) \times 2(P-1)$	Model parameter. Covariance matrix of locations of model parts.
$\mathbb{S}$	-	Denotes scale invariance.
t	1	Index for feature type.
$\mathbf{t}$	$(P-1)$	Model parameter. Mean of part scales.
T	1	Number of types of feature.
$\mathbb{T}$	-	Denotes translation invariance.
θ_{fg}	Variable	The parameters of the foreground model.
θ_{bg}	Variable	Fixed values of background model parameters.
U_l	$(P-1) \times (P-1)$	Model parameter. Variance matrix of part scales.
V_p	$a \times a$	Model parameter. Variance matrix of descriptors of model part p.
$V_{t,bg}$	$a \times a$	Fixed parameter. Variance matrix of background descriptors of type t.
$\mathbf{x}$	2	Location of a single feature.
$\mathbf{X}^i$	T	Locations of all features in image i. Each element is $1 \times N_t^i$.
$\mathbf{X}$	$T \times I$	Locations of all scales in all images. Each element is $1 \times N_t^i$.
ω	1	Mixture component index.
Ω	1	Total number of mixture components.

Table A.2: Overview of the notation used in the Constellation Model.

Additional comments on the table:

1. Depending on the appearance representation used, $\mathbf{d}$ may be of length a or $a+2$, the latter being if energy and residual terms (see Section 4.4.1) are added to each descriptor vector. If $\mathbf{d}$ is of length $a+2$ then correspondingly, $\mathbf{c}_p$, $\mathbf{c}_{t,bg}$ will also be of size $a+2$ and V_p, $V_{t,bg}$ will be $(a+2) \times (a+2)$ matrices.

A.1 Thesis statistics

- Total number of advisors – 2.
- Total number of institutions – 2.
- Total time – 5 years.
- Total pages – 193.
- Total number of words – 53327.
- Total number of figures – 266.
- Total number of tables – 17.
- Total number of images used in experiments – 21688.
- Number of conference/workshop papers – 7.
- Number of journal papers – 2.
- Number of prizes – 1.
- Total computation time – 192 $\times 10^6$ seconds or 2232 days or 6.1 years (on 2.8Ghz Intel Xeon CPUs).
- Total number of floating point operations – 5.79×10^{17} flops or 0.6 exaflops (assuming 3 gigaflops/second per cpu).
- Integral memory usage – 1.2 $\times 10^{16}$ byte seconds or 12 petabyte seconds.
- Total hard disk usage – 474GB.
- Total cost of electricity used for computation – ∼600GBP or $1100 (assuming 0.6A/cpu @ 240V; 0.08GBP/kWh; excluding RAID usage and cooling which could double the value).
- Total size of the thesis in PDF format – 51MB.
- Total size of the thesis is PS format – 726MB.
- Largest talk time per month on phone bill – 3886 minutes (most of it to Beijing, China).
- Largest In'n'Out burger eaten: 6×6 animal style.

All the above computation statistics are derived from the use of the VGG cluster in Oxford, operational since 2003.

Bibliography

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, pages 113–130, 2002.

[2] Agarwal, S. and Awan, A. and Roth, D. UIUC Car dataset. `http://l2r.cs.uiuc.edu/~cogcomp/Data/Car`, 2002.

[3] G. Agin. *Representation and description of curved objects*. PhD thesis, Stanford University, 1972.

[4] Y. Amit and D. Geman. A computational model for visual selection. *Neural Computation*, 11(7):1691–1715, 1998.

[5] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1300–1305, 1997.

[6] J. Amores, N. Sebe, and P. Radeva. Fast spatial pattern discovery integrating boosting with constellations of contextual discriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 2, pages 769–774, 2005.

[7] A. Bar-Hillel, T. Hertz, and D. Weinshall. Object class recognition by boosting a part-based model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 1, pages 702–709, 2005.

[8] K. Barnard, P. Duygulu, N. de Freitas, D. Forsyth, D. Blei, and M. Jordan. Matching words and pictures. *JMLR*, 3:1107–1135, February 2003.

[9] P. Belhumeur, J. Hespanha, and Kriegman D. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[10] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA*, volume 1, pages 26–33, June 2005.

[11] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[12] I. Biederman. *An Invitation to Cognitive Science, Vol. 2: Visual Cognition*, volume 2, chapter Visual Object Recognition, pages 121–165. MIT Press, 1995.

[13] T. Binford. Visual perception by computer. *Proc. IEEE Conference on Systems and Control*, 1971.

[14] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[15] R. C. Bolles and R. Horaud. 3DPO: A three-dimensional part orientation system. In T. Kanade, editor, *Three Dimensional Vision*, pages 399–450. Kluwer Academic Publishers, 1987.

[16] E. Borenstein. and S. Ullman. Class-specific, top-down segmentation. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, pages 109–124, 2002.

[17] J. M. Brady, J. Ponce, A. Yuille, and H. and Asada. Describing surfaces. In *International Symposium on Robotics Research*, pages 5–16. MIT Press, 1985.

[18] M. Burl and P. Perona. Recognition of planar object classes. In *Proc. Computer Vision and Pattern Recognition*, pages 223–230, 1996.

[19] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. European Conference on Computer Vision*, pages 628–641, 1996.

[20] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proceedings of the European Conference on Computer Vision*, pages 628–641, 1998.

[21] M.C. Burl, T.K. Leung, and P. Perona. Face localization via shape statistics. In *Int. Workshop on Automatic Face and Gesture Recognition*, 1995.

[22] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[23] A. Carlson, C. Cumby, J. Rosen, and D. Roth. The snow learning architecture, uiucdcs-r-99-2101. Technical report, Dept. of Computer Science, UIUC, 1999.

[24] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[25] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 1, pages 10–17, 2005.

[26] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[27] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA*, pages 886–893, 2005.

[28] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *JRSS* **B**, 39:1–38, 1976.

[29] G. Dorko and C. Schmid. Object class recognition using discriminative local features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Review(Submitted), 2004.

[30] Everingham, M. and Van Gool, L. and Williams, C. and Zisserman, A. PASCAL Visual Object Challenge Datasets. `http://www.pascal-network.org/challenges/VOC/voc/index.html`, 2005.

[31] Everingham, M. and Van Gool, L. and Williams, C. and Zisserman, A. PASCAL Visual Object Challenge Results. `http://www.pascal-network.org/challenges/VOC/voc/results_050405.pdf`, 2005.

[32] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, pages 1134–1141, October 2003.

[33] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *Workshop on Generative-Model Based Vision*, 2004.

[34] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA*, volume 2, pages 524–531, June 2005.

[35] P. Feltzenswalb and D. Hutenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61:55–79, January 2005.

[36] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073, 2000.

[37] R. Fergus and P. Perona. Caltech Object Category datasets. `http://www.vision.caltech.edu/html-files/archive.html`, 2003.

[38] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for google images. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, pages 242–256. Springer-Verlag, May 2004.

[39] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 1, pages 380–387, 2005.

[40] R. Fergus, P. Perona, and P. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, 2003.

[41] R. Fergus, M. Weber, and P. Perona. Efficient methods for object recognition using the constellation model. Technical report, California Institute of Technology, 2001.

[42] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 1, pages 40–54, 2004.

[43] M.A. Fischler and R.A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computer*, c-22(1):67–92, Jan. 1973.

[44] M. M. Fleck, D. A. Forsyth, and C. Bregler. Finding naked people. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, UK*, LNCS 1065, pages 591–602. Springer-Verlag, 1996.

[45] D. Forsyth and M. Fleck. Body plans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 678–683, 1997.

[46] Google Image Search. `http://www.google.com/imghp`, 2005.

[47] Google Translation Tool. `http://translate.google.com/translate_t`, 2005.

[48] W. E. L. Grimson and T. Lozano-Pérez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[49] C. J. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference, Manchester*, pages 147–151, 1988.

[50] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the determination of minimum cost paths. *IEEE Transactions on SSC*, 4:100–107, 1968.

[51] S. Helmer and D. Lowe. Object recognition with many local features. In *Workshop on Generative Model Based Vision 2004 (GMBV), Washington, D.C.*, July 2004.

[52] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999, Berkeley, CA, USA*, pages 50–57. ACM, 1999.

[53] A. Holub and P. Perona. A discriminative framework for modeling object classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 1, pages 664–671, 2005.

[54] D.P. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proceedings of the 1st International Conference on Computer Vision, London*, pages 102–111, 1987.

[55] M. Jerrum and A. Sinclair. The Markov chain Monte Carlo method. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*. PWS Publishing, Boston, 1997.

[56] T. Kadir and M. Brady. Scale, saliency and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[57] T. Kadir and M. Brady. Scale Scaliency Operator. `http://www.robots.ox.ac.uk/~timork/salscale.html`, 2003.

[58] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, volume 2, pages 506–513, June 2004.

[59] M. Kirby and L. Sirovich. Applications of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108, Jan 1990.

[60] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Obj cut. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, pages 18–25, 2005.

[61] M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. v.d. Malsburg, R.P. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans. Comput.*, 42(3):300–311, Mar 1993.

[62] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson. Object recognition by affine invariant matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[63] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[64] Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *cvpr*, volume 2, pages 97–104, 2004.

[65] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[66] T. Leung and J. Malik. Contour continuity and region based image segmentation. In *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, LNCS 1406, pages 544–559. Springer-Verlag, 1998.

[67] T.K. Leung, M.C. Burl, and P. Perona. Finding faces in cluttered scenes using random labeled graph matching. *Proceedings of the 5th International Conference on Computer Vision, Boston*, pages 637–644, June 1995.

[68] T.K. Leung, M.C. Burl, and P. Perona. Probabilistic affine invariants for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 678–684, 1998.

[69] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):77–116, 1998.

[70] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 1150–1157, September 1999.

[71] D. Lowe. Local feature view clustering for 3D object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, pages 682–688. Springer, December 2001.

[72] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[73] D. G. Lowe. The viewpoint consistency constraint. *International Journal of Computer Vision*, 1(1):57–72, 1987.

[74] K.V. Mardia and I.L. Dryden. "Shape Distributions for Landmark Data". *Advances in Applied Probability*, 21:742–755, 1989.

[75] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 384–393, 2002.

[76] LeCun, Y. MNIST Database of Handwritten digits. http://yann.lecun.com/exdb/mnist/index.html, 2004.

[77] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 525–531, 2001.

[78] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *CVPR*, volume 2, pages 257–263, 2003.

[79] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 1, pages 69–82. Springer-Verlag, May 2004.

[80] P. Moreels, M. Maire, and P. Perona. Recognition by probabilistic hypothesis construction. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 2, pages 55–68, 2004.

[81] J. L. Mundy and A. J. Heller. The evolution and testing of a model-based object recognition system. In *Proceedings of the International Conference on Computer Vision*, pages 262–282, 1990.

[82] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. *Int J. of Comp. Vis.*, 14:5–24, 1995.

[83] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100). Technical Report CUCS-006-96, Columbia University, February 1996.

[84] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 2, pages 71–84, 2004.

[85] Opelt, A. and Fussenegger, M. and Pinz, A. and Auer, P. Graz Object Category datasets. `http://www.emt.tugraz.at/~pinz/data`, 2005.

[86] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 1223–1228, 1998.

[87] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient limit. *Perception*, 14:449–470, 1985.

[88] J. Ponce and J. Brady. Towards a surface primal sketch. *Three dimensional machine vision*, pages 195–240, 1987.

[89] M. Pontil, S. Rogai, and A. Verri. Recognizing 3-d objects with linear support vector machines. In *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, pages 469–483, 1998.

[90] M. Riesenhuber and T. Poggio. Models of object recognition. *Nature Neuroscience*, pages 1199–1204, 2000.

[91] I. Rigoutsos and R. Hummel. A Bayesian approach to model matching with geometric hashing. *Comp. Vis. and Img. Understanding*, 62:11–26, Jul. 1995.

[92] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *Proc. Computer Vision and Pattern Recognition*, pages 272–280, 2003.

[93] C. Rothwell, D. Forsyth, A. Zisserman, and J. Mundy. Extracting projective structure from single perspective views of 3D point sets. In *Proceedings of the 4th International Conference on Computer Vision, Berlin*, pages 573–582, 1993.

[94] C. Rothwell, A. Zisserman, D. Forsyth, and J. Mundy. Planar object recognition using projective shape representation. *International Journal of Computer Vision*, 16(2), 1995.

[95] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, Jan 1998.

[96] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 1, pages 414–431. Springer-Verlag, 2002.

[97] B. Schiele and J. L. Crowley. Object recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36(1):31–50, 2000.

[98] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[99] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1759, 2000.

[100] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. Technical Report A. I. Memo 2005-005, Massachusetts Institute of Technology, 2005.

[101] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, pages 1470–1477, October 2003.

[102] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Proceedings of the IEEE International Conference on Computer Vision, Beijing*, page To appear, 2005.

[103] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, November 1991.

[104] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, pages 762–769, 2004.

[105] M. Turk and A. Pentland. Eigenfaces for recognition. *J. of Cognitive Neurosci.*, 3(1), 1991.

[106] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10), 1991.

[107] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[108] M. Weber. *Unsupervised Learning of Models for Object Recognition*. PhD thesis, California Institute of Technology, Pasadena, CA, 2000.

[109] M. Weber, W. Einhauser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *Proc. 4th IEEE Int. Conf. Autom. Face and Gesture Recog., FG2000*, pages 20–27, March 2000.

[110] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2101–2108, June 2000.

[111] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *Proc. 6th Europ. Conf. Comp. Vis., ECCV2000*, volume 1, pages 18–32, June 2000.

[112] M. Welling. An expectation maximization algorithm for inferring offset-normal shape distributions. In *Tenth International Workshop on Artificial Intelligence and Statistics*, 2005.

[113] J. Winn and N. Joijic. Locus: Learning object classes with unsupervised segmentation. In *Proceedings of the IEEE International Conference on Computer Vision, Beijing*, page To appear, 2005.