Efficient visual search of images and videos

Robotics Research Group
Department of Engineering Science
University of Oxford



Supervisor:
Professor Andrew Zisserman

Josef Sivic
Worcester College

2006

Abstract

This thesis investigates *visual* search of videos and image collections, where the query is specified by an image or images of the object. We study efficient retrieval of particular objects, human faces, and object classes.

Particular objects are represented by a set of viewpoint invariant region descriptors, so that recognition can proceed successfully despite changes in viewpoint, illumination and partial occlusion. Efficient retrieval is achieved by employing methods from statistical text retrieval, including inverted file systems, and text and document frequency weightings. This requires a visual analogy of a word – 'a visual word' – and it is provided by vector quantizing the region descriptors.

We also develop a representation for 3D and deforming objects, suitable for retrieval, based on multiple exemplars naturally spanning (i) different visual aspects of a 3D object and thereby implicitly representing its 3D structure, or (ii) different appearances of a deforming object. Multiple exemplar models are built automatically from real world videos, using novel tracking and motion segmentation techniques.

For retrieval of faces of a particular person in video, we focus on close-to-frontal faces delivered by a face detector, and develop a specialized visual vocabulary for faces by vector quantizing the appearance of facial features. Faces in the video are associated into face sets by tracking, and the multiple exemplar representation naturally models different appearances (such as closed and open eyes) within the set. This representation is also compact, to enable efficient retrieval.

To retrieve visual object classes, we build a new visual vocabulary of quantized local regions, which is tolerant to some amount of intra-class deformation. We employ a probabilistic latent variable model from statistical text analysis. In text, this model is used to discover topics in a corpus using the bag-of-words document representation. Here, we treat object categories as topics. We apply the probabilistic model in the visual domain, and show that models of visual object classes can be learnt from an unlabelled image collection without supervision. The learnt models are then applied to object class retrieval.

We demonstrate rapid retrieval in entire feature length movies despite significant amount of background clutter, variations in camera viewpoint and lighting conditions, and partial occlusion.

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.

Josef Sivic, Worcester College

To Linda

Acknowledgements

I would like to thank my supervisor Professor Andrew Zisserman. This thesis would not have been possible without his enthusiasm, guidance, and encouragement.

I am very grateful to a number of people who directly or indirectly influenced the work in this thesis: Alyosha Efros, Mark Everingham, Rob Fergus, Andrew Fitzgibbon, Bill Freeman, Krystian Mikolajczyk, Bryan Russell and Frederik Schaffalitzky. I must also thank Mike Brady and David Lowe for providing very thoughtful comments and suggestions.

I am indebted to Mark Everingham and James Philbin for proofreading parts of this thesis.

I would also like to thank all the people in the Oxford Vision Lab for creating a friendly working environment.

My thanks must go to Vaclav Hlavac and Tomas Pajdla who introduced me to computer vision while I was an undergraduate at the Czech Technical University in Prague.

Most importantly, I would like to thank my parents and family for their support and understanding.

My final and very special thanks goes to Linda for all her patience, help, support and inspiration.

Contents

Table of Contents **i**

1 Introduction **1**
1.1 The objective . . . 1
1.2 Motivation . . . 2
1.3 Why is it difficult? . . . 5
1.4 Contributions . . . 8
1.5 Thesis outline . . . 9
1.6 Publications . . . 10

2 Literature review **11**
2.1 Recognition of object instances . . . 11
2.1.1 Geometry based methods . . . 12
2.1.2 Global appearance methods . . . 16
2.1.3 Local appearance methods . . . 16
2.1.4 Affine covariant regions . . . 18
2.1.5 Local descriptors . . . 22
2.1.6 Representing 3D objects using local patches . . . 26
2.2 Recognition of object categories . . . 28
2.2.1 Sliding window classifiers . . . 28
2.2.2 Geometry-free methods . . . 30
2.2.3 Geometry based methods . . . 31
2.3 Codebooks in recognition . . . 33
2.3.1 Codebooks for Texture Classification – Textons . . . 34
2.3.2 Codebooks for object instance recognition . . . 34
2.3.3 Codebooks for recognition of object classes . . . 35
2.4 Efficient indexing for object recognition . . . 36
2.5 Image retrieval . . . 37
2.6 Text retrieval and statistical text analysis . . . 38
2.6.1 A standard text retrieval system . . . 38
2.6.2 Topic discovery models . . . 41

3 Efficient visual search for particular objects **44**
3.1 Introduction . . . 44
3.2 Viewpoint invariant object description . . . 45
3.3 Building a visual vocabulary . . . 46
3.3.1 Implementation . . . 47

3.4 Visual indexing using text retrieval methods . . . 48
3.4.1 Term frequency–inverse document frequency weighting . . . 49
3.4.2 Stop list . . . 50
3.4.3 Spatial consistency . . . 51
3.5 Object retrieval using visual words . . . 56
3.6 Experiments . . . 60
3.6.1 Vocabulary investigation . . . 67
3.6.2 Comparison of term frequency weighting and ranking methods . . . 73
3.7 Conclusions and discussion . . . 80

4 Object Level Grouping for Video Shots 81
4.1 Introduction . . . 81
4.2 Tracking affine covariant regions . . . 83
4.2.1 Basic tracking . . . 85
4.2.2 Short range track repair by region propagation . . . 85
4.2.3 Discussion . . . 89
4.3 Application I: Using multiple exemplars for retrieval . . . 90
4.3.1 Collating search results from multiple queries . . . 94
4.4 Object extraction by robust sub-space estimation . . . 95
4.4.1 Basic motion grouping using homographies . . . 96
4.4.2 Aggregating segmentation over multiple frames . . . 96
4.4.3 Object extraction . . . 100
4.4.4 Object extraction results . . . 102
4.5 Long range track repair . . . 106
4.6 Application II: Object-level video matching . . . 106
4.6.1 Retrieval performance . . . 107
4.7 Conclusions and discussion . . . 113

5 Video shot retrieval for face sets with face specific visual vocabulary 115
5.1 Introduction . . . 115
5.2 Obtaining sets of face exemplars by tracking . . . 117
5.2.1 Associating detected face exemplars temporally . . . 118
5.3 Representing and matching sets of face exemplars . . . 122
5.3.1 Facial feature location . . . 122
5.3.2 Representation of single faces . . . 123
5.3.3 Representation of face sets . . . 124
5.3.4 Matching face sets . . . 125
5.4 Retrieving sets of faces across shots – Video Google Faces . . . 128
5.5 Experimental evaluation . . . 128
5.5.1 Evaluation on ground truth . . . 128
5.5.2 Results on another movie . . . 133
5.6 Conclusions and discussion . . . 137

6 Discovering object classes in image collections 138
6.1 Introduction . . . 138
6.2 The topic discovery model . . . 139
6.3 Obtaining visual words . . . 142
6.4 Experiments . . . 143

6.4.1 Topic discovery . . . 145
6.4.2 Classifying new images . . . 148
6.4.3 Segmentation . . . 150
6.4.4 MIT image dataset results . . . 151
6.5 Discussion: What is being learnt by pLSA . . . 155
6.6 Application: object class retrieval . . . 158
6.6.1 Retrieval within the Caltech database . . . 161
6.6.2 Retrieval from a movie . . . 161
6.7 Conclusions and discussion . . . 164

7 Discussion **170**
7.1 Contributions of the thesis . . . 170
7.2 Recent developments . . . 171
7.3 Future work . . . 172

A Appendix **175**
A.1 Derivation of pLSA cost function . . . 175
A.2 Processed movies . . . 178

Chapter 1

Introduction

1.1 The objective

Our goal is to enable *visual* search of videos and large image collections. In particular, our aim is to retrieve those images or video shots containing a user-specified object with the ease, speed, and accuracy with which Google retrieves text documents (or web pages) containing particular words. The object is given by a user-specified region of interest in a single query image, and images and videos are searched on their visual content. This is in contrast to currently available text-based image and video search engines like 'Google Images' [157] or 'Google Videos' [158], where the query is given by few textual words, and videos and images are searched based on their surrounding text. Although these systems are relatively successful, their search potential is limited by the information given in the text. Objects and people not mentioned in the text are not retrieved.

Although some existing image and video search methods consider visual information, their performance is usually limited by the basic image and video features extracted (for example colour histograms). In contrast, we want to retrieve small objects in images with background clutter, changing camera viewpoint and lighting conditions. The main focus of this thesis is on efficient retrieval of particular objects. We also consider retrieval of visual object classes such as 'airplanes' (as opposed to a particular instance of an airplane). Finally, as humans are often interested in searching for other people, we also address the task of finding a particular person in video, using facial features.

Examples of successful visual searches in the entire movie 'Pretty Woman' [Marshall, 1990] are shown in figures 1.1, 1.2, 1.3 and 1.4.

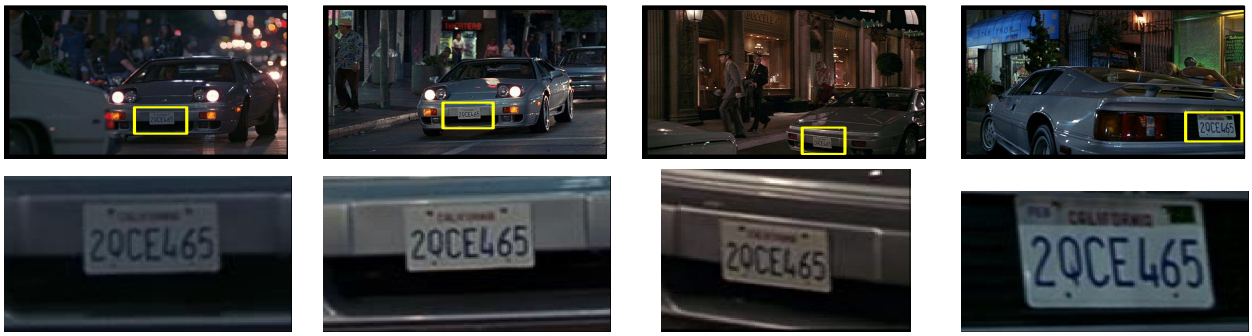
1.2 Motivation

The amount of visual information available in digital form is rapidly growing. Almost everyone now takes digital pictures and videos and builds a personal photo collection. For example, the author has about 8,000 personal digital images and videos, which amounts to about 7GB of data. TV and film production companies have large archives of digital video footage, and older footage is being transferred into digital form. With the world-wide spread of the Internet, the amount of easily accessible visual information is rapidly growing. For example, Google Images claims to index about 880 million images, and AOL claims to index about four million videos. The on-line video sharing website YouTube.com claims that 35,000 new videos are uploaded by its users every 24 hours. The significance of the fast visual search of images and videos is very much apparent, and possible applications include:

- **Product placement search.** Example query would be 'Give me all clips with this particular poster or logo.'
- **Image and video search.** An automatic searching tool which would immediately return all desired locations, objects or people in a vast amount of visual data. It could be used in the movie post-production industry, personal photo and video collections or Internet image and video archives. The query would be specified by a keyframe/shot of the desired location, or by outlining the desired object or person in the keyframe.
- **Intelligent fast-forward.** One could imagine a new user interface to DVDs, allowing fast-forwarding to shots taken at the same location or containing a particular person or an object. One could for example jump automatically between all shots taken in the 'Hogwarts dining hall' in Harry Potter films, or automatically fast-forward to the next appearance of Scarlett Johansson in the 'Lost In Translation' movie.



Query Query close-up

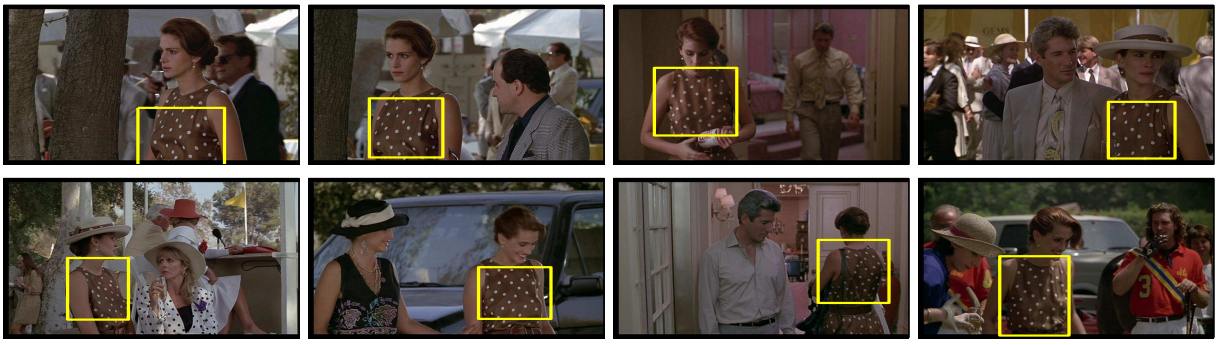


Four retrieved shots shown by a single frame (top) and object close-up (bottom)

Figure 1.1: Example search for a particular object – 'car plate'.

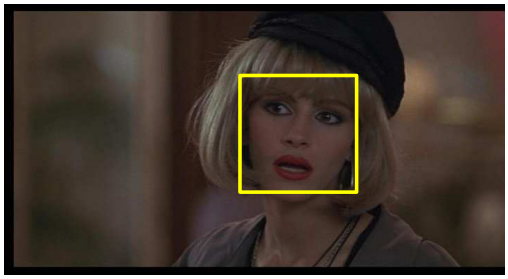


Query



The first 8 retrieved shots shown by a single frame

Figure 1.2: Example search for a particular object – 'Polka dot dress'.



Query



The first 12 retrieved shots shown by a single frame

Figure 1.3: Example search for a face of a person.



Query image



Retrieved frames from the movie 'Pretty Woman'.

Figure 1.4: Example search for an object class – 'airplanes'.

- **Automatic museum guide.** Using a small wearable camera, one could discover facts about a painting just by looking at it.

- **Mobile phone applications.** With the current spread of mobile phones with digital cameras, one can imagine taking a picture of a famous building, say the Prague Castle, and using this picture to automatically retrieve useful facts about the building from an annotated architecture database.

- **Surveillance.** The volume of surveillance data is huge and a fast searching technique would make the data easily accessible. An example query could be: Given an image of a particular car, search for the car in an archive of surveillance video.

1.3 Why is it difficult?

The range of possible applications for visual search is exciting. However, there are several challenges which need to be addressed:

- The imaged appearance of an object, a scene, or a person, can change significantly due to changing camera viewpoint and scene lighting. Examples are shown in figures 1.5 and 1.6. In addition, as shown in figure 1.8, objects can be partially occluded.

- As shown in figure 1.9, 3D objects can be imaged from various viewpoints, showing different visual aspects.

- When searching for a class of objects, say motorbikes, we need to overcome intra-class variations. Examples are shown in figure 1.10.

- When searching for faces of a particular person, we have to deal with changing facial expression, in addition to viewpoint and lighting variations, and partial occlusion. Examples are shown in figure 1.7.

- A successful retrieval system must be able to deal with large amounts of data. In this thesis, we will present results on feature-length movies. A typical feature-length movie has about 1,000 shots, and 100,000–150,000 frames.



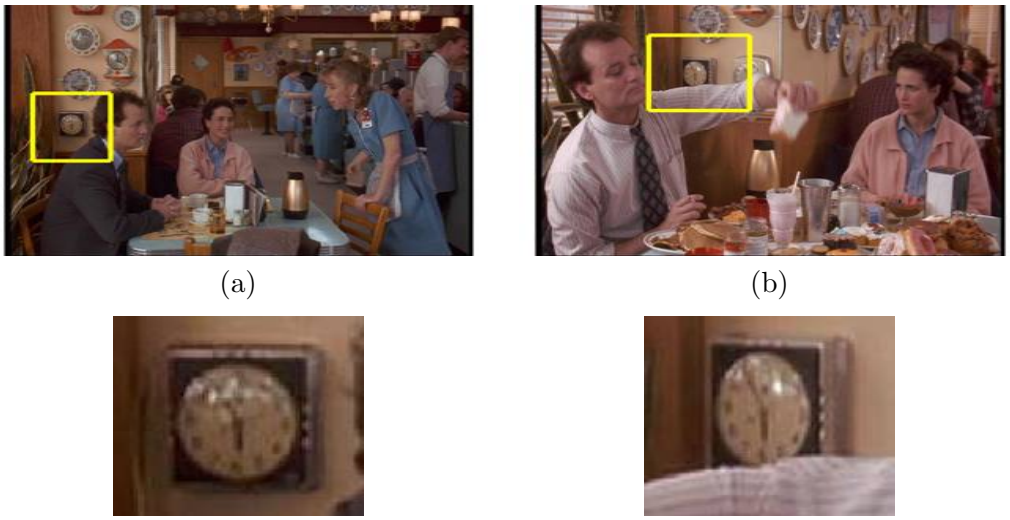
Figure 1.5: **Challenges of visual search – camera viewpoint.** The top row shows the same object ('Phil' sign) in two different frames from the movie 'Groundhog Day'. The bottom row shows object close-ups. Note the change in scale and foreshortening effects due to changing camera viewpoint. Note also that the images contain a significant amount of background clutter, and the object takes up only a small portion of the frame.



Figure 1.6: **Challenges of visual search – illumination.** The same object (digital clock) appears at various points in 'Groundhog Day'.



Figure 1.7: **Challenges of visual search on faces – expression and lighting.** Four example faces of the same person from the movie 'Groundhog Day'.



(a) (b)

(c) (d)

Figure 1.8: **Challenges of visual search – partial occlusion.** (a) and (b) show the same object (clock) outlined in yellow in two different frames from 'Groundhog Day'. (c) and (d) show corresponding object close-ups. Note that the clock in (b) and (d) is partially occluded by the actor's arm.



Figure 1.9: **Challenges of visual search – visual aspect.** Various views of the same 3D object (ambulance van) that appear in the movie 'Run Lola Run'.



Figure 1.10: **Challenges of visual search – intra-class variations.** Three instances of the same object class (motorbikes). Note that all the shown motorbikes have similar parts (wheels, handlebar, saddle, fuel tank) but that the appearance of those parts can vary.

1.4 Contributions

The contribution of this thesis can be divided into two main themes, summarized below. A more detailed account of contributions appears in section 7.1(p170).

1. Visual words

Inspired by the success of textual search, we develop a novel image representation, where an image is described as a collection of visual 'nouns', as analogues to words. These descriptors are built by quantizing local image patches, and are developed with a controlled degree of invariance to certain appearance transformations. For particular objects, we quantize the appearance descriptors of affine covariant regions, which enables retrieval despite variations in camera viewpoint and lighting conditions. Objects are represented by a collection of visual words, so that partial occlusion is also handled. For object classes, the quantization brings some amount of tolerance to within-class deformations. Special attention is paid to retrieval of faces of a particular person, for which we develop a face-specific visual vocabulary by quantizing appearance descriptors of facial features.

The benefit of quantization is that all descriptors assigned to the same visual word are deemed matched. As a result, matches are effectively pre-computed, so that at run-time, frames and shots containing any particular object or face can be retrieved with no delay. This means that any object (or face) occurring in the video can be retrieved even though there was no explicit interest in the object when descriptors for the video were built and matched.

This new object representation by visual words also allows us to take search methods and probabilistic models from text retrieval and statistical text analysis and employ them for efficient visual search. In particular, we investigate application of two text models in the visual domain. The first, a 'bag-of-words' model, enables efficient search and indexing for large image collections. The second, probabilistic Latent Semantic Analysis (pLSA), fits a latent variable generative probabilistic model in order to discover topics in text corpora. Here we treat images as documents, and visual object classes as topics. We show that models of visual object classes can be learnt from an unlabelled image collection in an unsupervised way. The learnt models are then used for retrieval of object classes in feature length movies.

2. Representation of 3D and deformable objects for efficient retrieval

We develop a novel representation of 3D and deformable objects by sets of exemplars naturally spanning different visual aspects of a 3D object (e.g. the front, side and back of a van) or the different appearances of a deforming object (e.g. a speaking, smiling and blinking face). Each exemplar is an image region represented by a collection of visual words.

We extend the standard paradigm of image based retrieval, where the query is defined by a region within a single image, to retrieval at an *object-level*, where a query object is defined by several exemplars over multiple images. We investigate two scenarios. In the first one, individual object exemplars perform independent queries, which are then collated into a final result. In the second one, multiple exemplars are compactly represented by a single distribution over visual words, and the retrieval is performed by a single query.

To automatically obtain multiple exemplar object models from real-world videos, we employ three complementary cues. First, the *temporal continuity* cue is exploited by tracking affine covariant regions in the video. Second, the *common motion* cue is exploited by segmenting scenes into independently moving objects. For example, the front, side and back of a moving van, although never visible together in a single frame, can be associated because they move together as a rigid object. Third, the *common appearance* cue is exploited by matching objects within a shot. This allows us to associate objects across occlusions.

1.5 Thesis outline

In chapter 2 we review previous work in visual object recognition, image retrieval, and text retrieval, related to the thesis. Special attention is given to object recognition approaches using local image regions, which form the basis for our object retrieval method.

Chapter 3 introduces our method for efficient retrieval of particular objects in videos. First, we give details of viewpoint invariant image description and visual vocabulary building. Then we describe the use of text retrieval methods for visual indexing and detail ranking by the spatial layout of visual words. Finally, we perform a series of experiments testing the behaviour of the proposed approach on a set of ground truth queries from an entire feature length movie.

In chapter 4, we extend the retrieval approach described in chapter 3 to 3D objects and deforming objects. We build an object representation, consisting of multiple exemplars automatically extracted from video using motion. We develop techniques for tracking affine covariant regions and independent motion segmentation, and show examples of retrieval of 3D and deforming objects from feature-length movies.

Chapter 5 considers retrieval of faces of a particular person in video, and builds on ideas from chapters 3 and 4. The focus is on close-to-frontal faces, delivered by a face detector. Each face is represented by a collection of face-specific visual words, centred on facial features. Individual faces in the video are associated into face-sets by tracking. Face-sets are similar to the multiple exemplar object representation of chapter 4, and can capture different facial expressions. Finally, we describe a compact and efficient representation of entire face-sets, and show retrieval examples again on several feature length movies.

In chapter 6 we focus on retrieval of visual object *classes* from still image collections. First, we build a new, more general, visual vocabulary that captures some amount of intra-class variation. Then we describe the probabilistic latent semantic analysis (pLSA) model from statistical text analysis literature, and outline it's application in the visual domain to automatically build models of visual object classes from a large unlabelled image collection. We examine the built models through a series of experiments on two image collections. Finally, we apply the models to object class retrieval in movies.

Chapter 7 summarizes the contribution of the thesis, reviews recent developments in the literature, and discusses avenues for future research.

1.6 Publications

The first version of the work presented in chapter 3 was published in ICCV'03 [138]. Two invited summary papers appeared in EUSIPCO'04 [135] and PCM'04 [139]. An extension, applying ideas developed in chapter 3 to video mining, appeared in CVPR'04 [140]. The work presented in chapter 4 first appeared in ECCV'04 [136], and an extended version was published in IJCV [137]. The material presented in chapter 5 was published in CIVR'05 [133]. The work presented in chapter 6 was published in ICCV'05 [134]. An extension, building on ideas put forward in chapter 6, appeared in CVPR'06 [117].

Chapter 2

Literature review

This thesis builds on more than 40 years of research in visual object recognition. The literature review of object recognition is split into two parts: (i) recognition of object instances is reviewed in section 2.1, and (ii) object category recognition is reviewed in section 2.2. We pay special attention to recognition methods using local regions. We review affine covariant region detectors that later form the basis for our object representation. Visual search methods proposed in this thesis are based on forming a 'visual vocabulary' – a codebook of quantized local regions. Related recognition approaches, using some form of codebook of quantized local regions, are reviewed in section 2.3. Methods for efficient indexing in object recognition are reviewed in section 2.4, and relevant work in image retrieval is discussed in section 2.5. One contribution of this thesis is applying text retrieval and statistical text analysis methods to visual search. Section 2.6 reviews relevant ideas from text analysis and search, such as the vector space model, inverted file indexing and probabilistic models for discovering topics in text collections.

In some cases related papers are reviewed and discussed in relevant chapters.

2.1 Recognition of object instances

The goal of visual object recognition is to detect the presence of an object in an image, and possibly localize the object in the image and estimate its pose. This usually involves designing an object representation that can model the imaged appearance of an object under a broad class of imaging conditions, such as varying object and camera pose, scene lighting, partial occlusion, and possibly

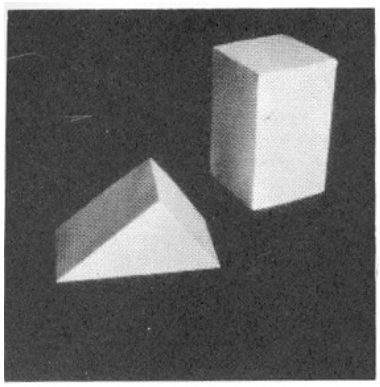


Figure 2.1: Example scene from the work of Roberts (1965) [111]. Objects are made of polyhedra on a uniform background.

deformation. Such representation should also be robust enough to deal with large amounts of background clutter.

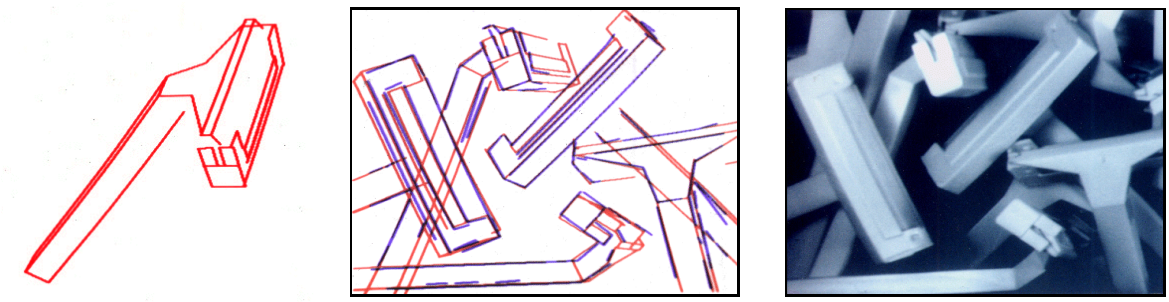
Early approaches to object recognition made strong simplification assumptions about the real world. An example is the 'Blocks world' by Roberts (1965) [111], where objects were made of combinations of polyhedra on a uniform background. An example is shown in figure 2.1. Later, objects were represented by combinations of generalized cylinders, which enabled modelling of fairly complex curved objects.

Below we review some outstanding approaches in object instance recognition. We start from early works based on 3D geometric object models and geometric invariants. Then we touch on global appearance methods that essentially represent objects by storing images taken from different viewpoints. Finally, most of this section is devoted to approaches representing objects by local image regions.

2.1.1 Geometry based methods

A common theme dominating the geometry based approaches is the use of object boundaries to represent objects. Such approaches were appealing as they were invariant to a large class of illumination changes. In practice however, these methods suffer from unreliable detection of object contours in real images.

Alignment: The basis of the alignment approach is the 'hypothesize and test' paradigm from artificial intelligence. First, a correspondence is hypothesized between an object model (e.g. a 3D



(a) (b) (c)

Figure 2.2: Object recognition example from the work of Lowe (1987) [88]. (a) 3D wire-frame object model. (b) Successful matches between the model in (a) and image shown in (c). Object instances are recognized despite severe partial occlusion.

wire-frame model) and detected image primitives (e.g. points and line segments), which gives an object pose hypothesis. In the verification step, the model is projected into the image and the entire projected object contour is compared to the evidence coming from the measured image edges.

Two examples of the alignment approach are the works of Lowe [88] and Ullman and Huttenlocher [67]. In [67], a minimal number of corresponding primitives is used to determine a hypothesis of the object pose. A 'weak perspective' camera model is assumed and therefore only three model to image correspondences are required. Lowe [88] used the idea of perceptual grouping to reduce the correspondence search. Groupings of edge segments are formed based on proximity, collinearity and parallelism. An edge grouping (e.g. a pair of parallel edge segments of similar length) creates a correspondence hypothesis with only a small number of similar model edge groups. A recognition example is shown in figure 2.2.

Due to use of the full 3D object model, alignment approaches could deal with significant partial occlusion. Some amount of background clutter could be also handled. The challenge is the correspondence search. Three approaches (interpretation trees, pose clustering and RANSAC), which address the difficult correspondence search, are reviewed next.

Interpretation trees: The *interpretation tree* [54, 58] approach searches the space of all possible correspondences. Imagine there are N image primitives and M model primitives. Each image primitive can be assigned to (or labelled as) any of the model primitives, thus, without using any constraints, we have N^M possible labellings or assignments. The interpretation tree tries to search this space. The idea is to apply constraints and heuristics to heavily prune the tree. For example,

a model line segment could be matched to an image line segment only if the lengths of the two line segments are similar.

Pose clustering: Another approach, developed by Stockman [144] and Thompson and Mundy [148], is *pose clustering*. The idea is to discretize the space of possible transformations (object poses) and cast votes for transformations supported by k-tuples of hypothesized correspondences. The intuition is that correct correspondences should result in similar transformations (up to noise), and should therefore create a 'peak' in the discretized transformation space. In the general case, and using notation above, the complexity is $(MN)^k$, where k is the number of correspondences needed to determine the transformation, e.g. 2 points for similarity transformation (translation, scale and rotation). Difficulties with the pose clustering approach are: (i) the quantization effects resulting from discretizing the transformation space, and (ii) that the number of bins grows exponentially with the transformation complexity. For example, in the case of 2D (planar) affine transformation with 6 degrees of freedom and 10 bins per degree of freedom, 10^6 bins would be needed. Pose clustering was successfully used in the impressive object recognition system of Lowe (1999) [85]. A similar approach was recently used in the successful object category recognition work of Leibe and Schiele (2004) [79].

RANSAC: Perhaps the most successful correspondence search method is the *RANdom SAmple Consensus* algorithm of Fischler and Bolles (1981) [51]. RANSAC is a robust method for model fitting to noisy data with outliers. The main idea is to randomly sample a minimal set of correspondences, and compute the aligning transformation. Transformations are scored by the consensus with the remaining correspondences. The idea is that the correct transformation should have a large support from other correct correspondences, whereas an incorrect transformation would be consistent with only a small number of correspondences. In the alignment context, RANSAC explores the space of model to image transformations, and the search is guided by the potential correspondences. A good summary of RANSAC is given by Hartley and Zisserman [60]. RANSAC is an active research area even 25 years after the original paper by Fischler and Bolles.

In all three of the above mentioned approaches, it is highly beneficial to reduce the number of potential hypothesis which need to be tested. This can be done by perceptual grouping, suggested

by Lowe [88]. Another approach is to 'identify' features on the object model and in the image, and only hypothesize matches between features with similar identifiers. Such identifiers can be computed from local geometric structure (e.g. geometric invariants, which are reviewed next), or local appearance (reviewed in section 2.1.3).

Geometric hashing and geometric invariants: The basic idea of geometric hashing, proposed by Lamdan and Wolfson [77, 176], is to extract geometric features from a set of model object images and store them in an indexing structure, for example a hash table. In recognition, a set of features is extracted from a novel image and matched against the indexing structure. If a novel image feature 'lands sufficiently close' to a stored feature of one of the object models, a possible match is declared. The key elements are that: (i) measured features are based on geometric structure rather than imaged appearance, and (ii) measurements are invariant to a certain required class of transformations. For example, Lamdan and Wolfson describe objects by sets of points. Invariance to a 2D affine transformation is achieved by storing point coordinates in a normalized frame defined by a triplet of points, and a redundant representation is obtained by choosing every possible triplet as a basis.

Another example is the object recognition system developed by Rothwell *et al.* [114, 115], where projective invariants of small groups of points, lines and curves are used to index into a database of objects. The system can deal with large viewpoint changes because of the projective invariants used, and also some amount of occlusion, as several invariants are extracted from an object. Another advantage of this indexing based approach is that it scales sub-linearly with the number of objects in the database. This is in contrast to the alignment methods described above, where all database objects have to be matched to each novel image in turn. One of the main drawbacks here is that single view projective invariants only exist for certain classes of objects, such as planar objects or solids of revolution. Another drawback, which holds for most geometric approaches based on object contours mentioned in this section, is the difficulty of extracting reliable edge information from real world images. Appearance based methods, which avoid this issue, are described next.

2.1.2 Global appearance methods

Another class of methods represents objects by storing global appearance information. Viewpoint and lighting invariance is achieved by storing many images captured from different viewpoints and under different illuminations. Recognition of a novel image then proceeds by finding the most similar image in the database. The object pose and illumination can be determined from the pose and illumination of the most similar database image (for which it is known through the capturing process). An example of this approach is the work of Murase and Nayar [98]. In their work, a more compact model (than storing the original images) is obtained by using eigenspace [155] representation.

The appeal of this approach lies in its simplicity. The drawbacks are: (i) the large amount of training data required, and (ii) problems with partial occlusion and background clutter, due to the global appearance representation used. It should be noted though that similar 'non-parametric' approaches, which essentially store many training examples, can be quite powerful for some tasks. An example is the work by Efros in action recognition [36].

The method of Murase and Nayar, described above, is an example of view-based object representation, as it avoids building an explicit geometric model of an object. It is somewhat similar to an earlier work on aspect-graph object representation [73]. The difference is that in [98], the viewing sphere is sampled densely, without any knowledge of the object structure. This is in contrast to the aspect-graph, where two neighbouring nodes on the viewing sphere differ in some significant way, for example a previously occluded facet becoming visible.

In chapter 4 of this thesis, we consider a view-based representation of 3D objects for retrieval. The difference is that each object view is represented by the appearance of local regions, thus allowing it to handle partial occlusions.

Another global appearance method is the work of Schiele and Crowley [123], where images are represented by histograms of local texture descriptors.

2.1.3 Local appearance methods

Local appearance methods are at the heart of some of the most successful object instance recognition systems to date. The basic idea is that objects are represented by the appearance of hundreds of



(a) (b) (c)

Figure 2.3: Object recognition example from the work of Lowe [85, 87]. (a) Model images for two objects. (b) Novel scene image. (c) Recognition result. Matched local regions and model image bounding boxes are overlaid.

local patches. First, a database of objects is built by storing local appearances in the form of a descriptor vector. In recognition, local regions are extracted from a novel image and matched to the object database using their appearance descriptor values. This initial set of local region matches is then disambiguated using semi-local or global geometric constraints. For example if the object is planar, we can require that all local regions are mapped by a planar homography.

The method can naturally handle partial occlusion as objects are represented by multiple patches. Robustness to background clutter is achieved by making the appearance descriptors discriminating and applying geometric constraints. Viewpoint and illumination invariance is achieved by careful design of local region detectors and descriptors, which we describe below.

Local region based systems can represent and reliably recognize a surprising variety of real-world objects, provided the objects are at least lightly textured. The key to the good performance is: (i) repeatability of local regions, which means that a local region describing a particular part of the object can be reliably re-detected over a range of camera viewpoints, illumination conditions and camera noise; (ii) robust and discriminative description of local regions, i.e. local regions with different appearances can be reliably discriminated in the descriptor space, and descriptor computation is robust to noise which might occur in the region detection process; (iii) redundant object description by hundreds of local regions, which can cope with occasional missing or mismatched regions and partial occlusion.

An example of local region based representation is the work of Schmid and Mohr [126]. Their

application is retrieval of images. Each image is described by local circular patches based at Harris [59] interest points. To achieve rotation invariance, each patch is described by a set of differential rotation invariants. Scale invariance is achieved by representing each Harris corner with several patches at different scales. A database of images is represented by an index of descriptors, and images are retrieved by matching on descriptor values. Descriptor based matches are then disambiguated using a weak local geometry test, where a region match is required to have another region match in its vicinity. A similar verification test is used in chapter 3 of this thesis for object retrieval.

Another work using local regions was the paper by Lowe [85]. Objects are represented by a set of circular regions placed at local extrema of Difference of Gaussian (DoG) operator in both position and scale. Rotation invariance is achieved by computing the patch descriptor relative to a dominant image gradient orientation within the patch. The resulting circular patches at various scales are described by 'SIFT' descriptors, which capture orientations of image gradients on a coarse spatial grid within the patch. Invariance to affine pixel intensity changes within each patch is achieved by an appropriate intensity normalization. The SIFT descriptor is reviewed in more detail in section 2.1.5. Recognition of a novel image proceeds by first extracting regions and their descriptors, and then fast approximate nearest neighbour matching to a database of pre-extracted descriptors of model objects. Pose clustering (using similarity transformation) is used to suggest potential candidate objects in the novel image. Finally, the presence of an object is verified by fitting a 2D affine transformation. Impressive close to real-time recognition performance was demonstrated on a database consisting of tens of objects and scenes with multiple objects with significant occlusion, background clutter and ranges of scales and rotation.

Despite the success of the above methods, the range of applicable camera viewpoints and object poses is still limited by invariance to only similarity transformations (i.e. translation, scale and rotation). Methods overcoming this limitation are described next.

2.1.4 Affine covariant regions

Viewpoint limitations of methods described in the previous section stem from the use of fixed shape circular support regions, illustrated in figure 2.4. The key idea of affine covariant region detectors

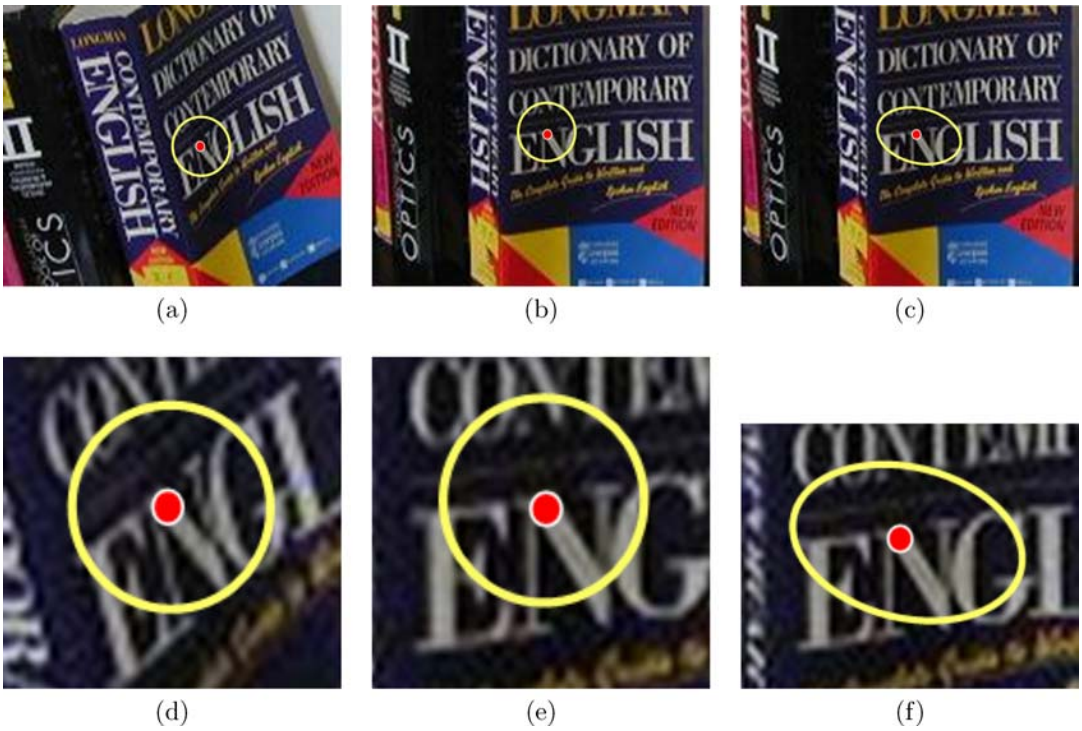


Figure 2.4: **Limitation of circular support regions under large viewpoint changes.** (a) First viewpoint. (b)-(c) second viewpoint. The circular region in (b) does not cover the same object surface patch as the circular region in (a). What is needed is a deformation of the circular region in (b) by an anisotropic scaling to the ellipse shown in (c). Note that regions in (a) and (c) cover approximately the same surface patch on the book. (d)-(f) close-ups of (a)-(c). Figure reproduced from [96].

is that the shape of the region is automatically adapted to underlying image intensities in a single image in such a way that regions detected *independently* in each image correspond to the same 3D surface patch. Note that the method of Lowe [85], described in the previous section, uses pixel intensities in a single image to determine the scale of the circular patch (by choosing regions that are local extrema of the DoG operator response). Patch orientation (rotation) was chosen based on local image gradient orientations. The innovation underpinning affine covariant regions lies in automatically determining the *shape* of the local region. Note that regions described in this section are called 'affine covariant' because their size and shape transforms *covariantly* with a 2D affine image transformation. An affine transformation is a reasonably good local approximation to transformations arising from viewpoint changes for locally planar (or at least smooth) surfaces.

Several affine covariant region detectors have been proposed in the literature. For example methods based on: (i) iterative region shape adaptation about an interest point [12, 93, 119], (ii)

selecting stable areas from an intensity watershed segmentation [91], (iii) parallelogram growing starting from an interest point [156], (iv) exploring intensity profiles along lines emanating from a local intensity extrema [156], and (v) searching over elliptical region support for extrema of a saliency measure [70]. In the following we review in more detail methods (i) and (ii), as these methods are used in this thesis for describing object appearance. A comprehensive review of affine covariant region detectors, and a comparison of their performance appeared in Mikolajzcyk *et al.* [96].

Shape adapted affine covariant regions

The first approach, investigated by Baumberg [12], Mikolajczyk and Schmid [93] and Schaffalitzky and Zisserman [119], is based on Harris [59] interest points detected over a range of image scales. Only interest points, which are local maxima over scale evaluated by a Laplacian of Gaussian operator, are kept. An iterative procedure is then used to adapt the shape of the circular point neighbourhood of each interest point to the skew normalized frame, where *second moment matrix* of image gradients is isotropic. The second moment matrix, $\mathtt{M}_I$, is the covariance matrix of gradients of image intensities over the local region (neighbourhood) $\Omega \subset \mathcal{R}^2$,

$$\mathtt{M}_I = \int_\Omega (\nabla I)\ (\nabla I)^\top\ \frac{dx\ dy}{|\Omega|} = \int_\Omega \begin{pmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{pmatrix} \frac{dx\ dy}{|\Omega|}, \tag{2.1}$$

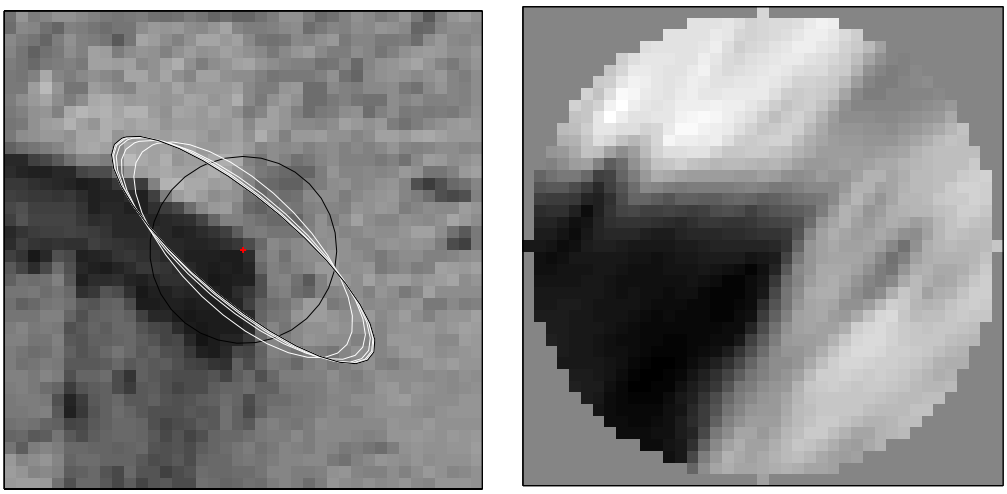
where I_x and I_y denote $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ respectively, and $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)^\top$ denotes image gradient.

Suppose images I and J are related by a local (no translation) affine geometric transformation, $I(\mathbf{x}) = J(\mathtt{A}\mathbf{x})$, then the second moment matrix of image I, $\mathtt{M}_I$, transforms as

$$\mathtt{M}_J = \mathtt{A}^{-\top} \mathtt{M}_I \mathtt{A}^{-1}. \tag{2.2}$$

Making $\mathtt{M}_J$ the unit matrix, we can write $\mathtt{M}_I = \mathtt{A}^\top \mathtt{A}$, from which $\mathtt{A}$ can be computed up to an arbitrary rotation.

Each iteration of the shape adaptation algorithm then proceeds as follows: (1) gradients of image intensities are estimated using a circular symmetric Gaussian smoothing kernel, (2) a new transformation matrix $\mathtt{A}$ is computed to make the second moment matrix isotropic, and (3) the



(a) (b)

Figure 2.5: **Detecting shape adapted affine covariant regions [12, 119]**. (a) The iterative algorithm starts with a circular neighborhood of a Harris corner and adapts its shape to make the second moment matrix of image gradients isotropic. Ten iterations are shown. The final ellipse is shown in black. The scale of the initial circular neighborhood (in black) was chosen as a local maxima of the Laplacian over scale. (b) The elliptical image patch transformed into the skew normalized frame.

image is transformed to a new skew normalized frame. Figure 2.5 illustrates ten iterations of the shape adaptation. The original idea of iterative shape adaptation using the second moment matrix is due to Lindeberg and Garding [84], and implementation details are explained by Schaffalitzky and Zisserman in [119]. Mikolajczyk and Schmid [93] extend this approach to search not only over the shape of the region but also over its scale and position.

Maximally stable regions

Another affine covariant region detector was developed by Matas *et al.* [91]. An intensity image is thresholded with all possible thresholds, $t \in S$, where $S = \{0 \dots 255\}$, resulting in sequences of nested contiguous regions. Let $Q_1, \dots, Q_i$ be such a sequence of nested regions, i.e. $Q_i \subset Q_{i+1}$. Region Q_{i^*} is declared *maximally stable* iff

$$q(i) = \frac{|Q_{i+\Delta} \setminus Q_{i-\Delta}|}{|Q_i|} \tag{2.3}$$

has a local minimum at i^*. Δ is a parameter, $\setminus$ denotes set difference, and $|.|$ denotes cardinality, i.e. the region area in the case of a discrete image. Intuitively, the method is looking for regions

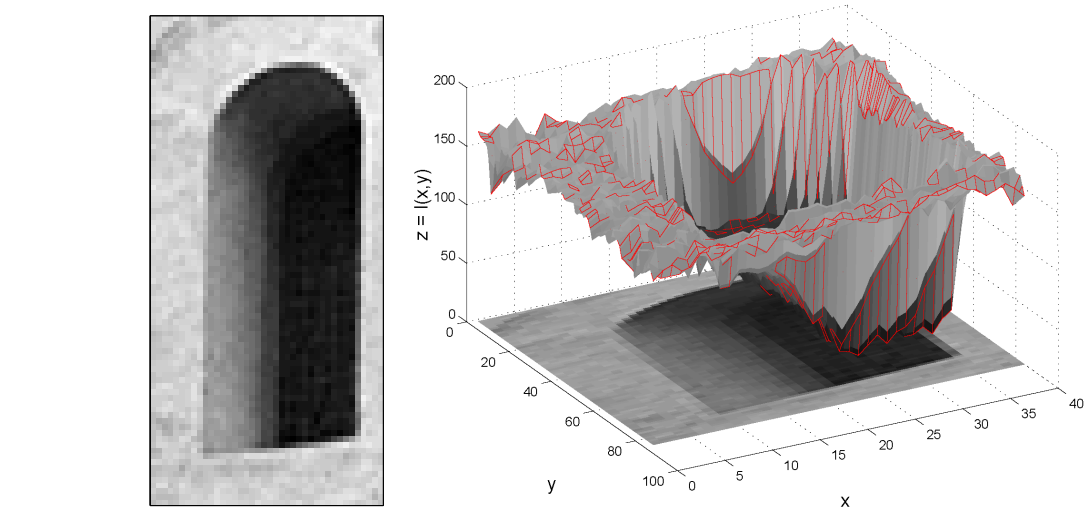
with an approximately stationary area for varying threshold values t. The method is illustrated in figure 2.6, and an example of two maximally stable regions covering the same scene element is shown in figure 2.7. Note that the maximally stable region detector outputs arbitrary shaped image regions. In this thesis, maximally stable regions are represented by ellipses with the same shape moments up to the second order. The same representation was used by Mikolajczyk *et al.* [96] and Schaffalitzky and Zisserman [121].

2.1.5 Local descriptors

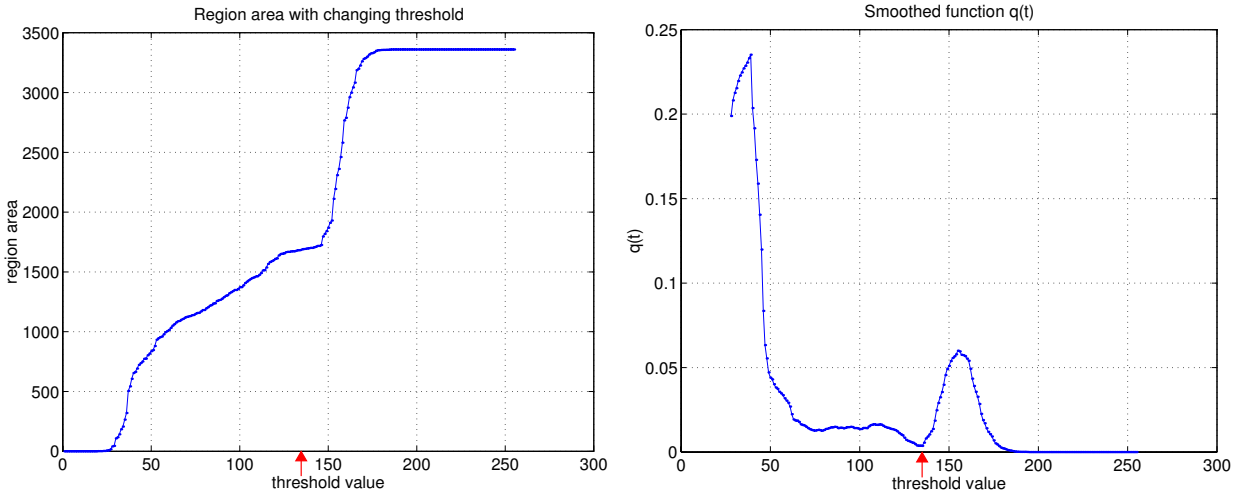
After detecting local regions in an image, the usual procedure is to extract a descriptor from each region, which is then used for object matching. Some methods for extracting local image descriptors are reviewed next. A fairly comprehensive review of local region descriptors and experimental comparison of their performance is given by Mikolajczyk and Schmid [94].

Raw image intensities: The simplest method to describe the detected local region is to use raw pixel intensities within the region. To deal with affine photometric transformations, $I' = aI + b$, the common practice is to normalize image intensities within the patch by subtracting the mean and scaling the variance to 1. The problem with using raw pixel intensities is the high dimensionality of the resulting descriptor: even a small patch of 12×12 pixels gives a descriptor with 144 dimensions. Similarity between patches is usually measured by cross-correlation or sum of squared differences. These measures are sensitive to patch localization errors, which often occur.

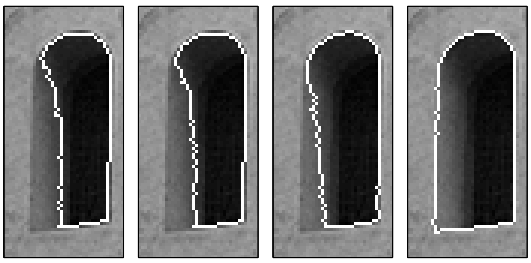
Filter responses: To reduce the dimensionality of the descriptor, a common approach is to apply a bank of filters over the image patch and store the filter responses, instead of the pixel values. Mikolajczyk and Schmid [92] use steerable filters [53], which are Gaussian derivatives steered in the direction of prevailing gradient orientation in the image patch. Steering the derivatives makes the descriptor invariant to image rotation. The quality of the representation depends on the number of coefficients stored. At the same time, higher order derivatives of the image function tend to be noisy.



(a)



(b)



(c)

Figure 2.6: Illustration of the method for detecting maximally stable regions by Matas *et al.* [91]. (a) Original sub-image (left) and its intensity function in 3D (right). (b) Area (left) and $q(t)$ function given by equation (2.3) (right) of the largest connected region with changing value of threshold t. (c) The largest regions within the image corresponding to thresholds at local minima of q, located at values 75, 79, 100, 135 (from left to right). Note that the threshold corresponding to the visually correct window outline lies at the most prominent local minimum at the value of 135 (indicated by a red arrow).

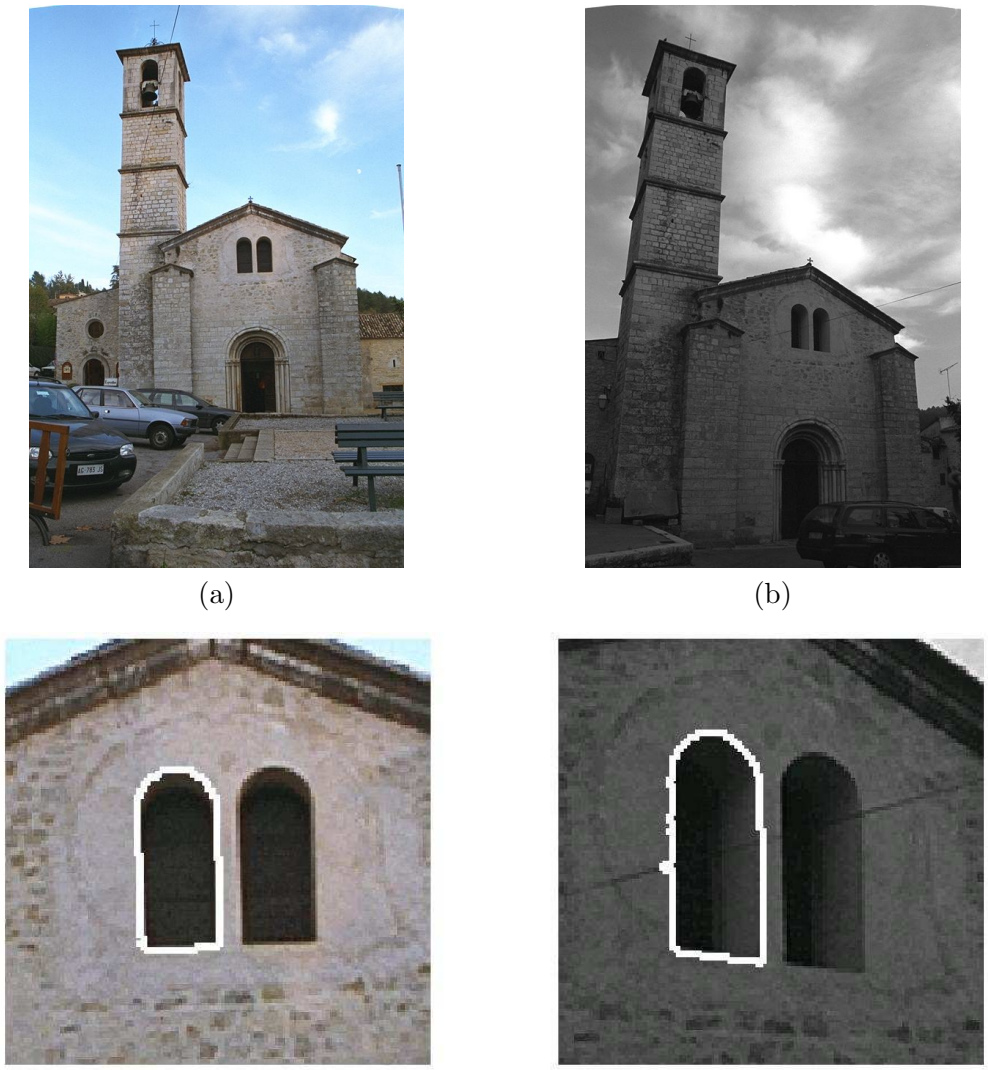


Figure 2.7: **An example of maximally stable regions of Matas [91].** (a),(b) Two images of the same scene taken from different viewpoints. Note the scale change and perspective distortion. (c),(d) Close-ups of two maximally stable regions shown in white detected independently in images (a) and (b). Note that both regions cover the same area on the surface of the building. ((c) and (d) are taken from [121]).

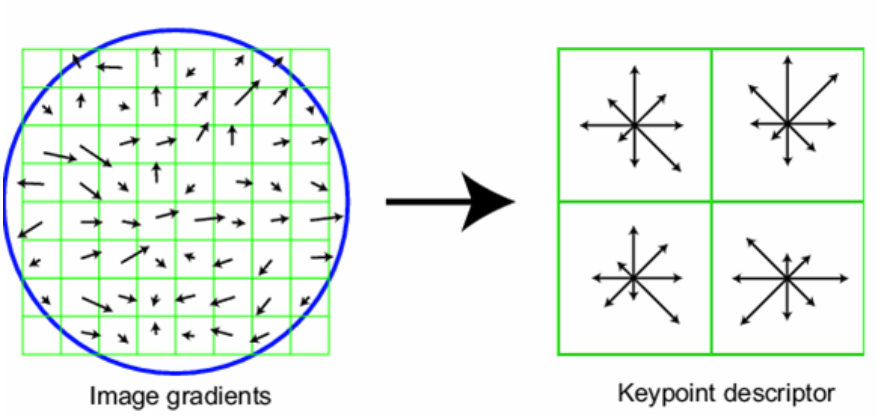


Figure 2.8: Illustration of the SIFT descriptor of Lowe [85]. Image gradients within a patch (left) are accumulated into a coarse 4×4 spatial grid (right, only a 2×2 grid is shown). A histogram of gradient orientations is formed in each grid cell. 8 orientation bins are used in each grid cell giving a descriptor of dimension $128 = (4 \times 4 \times 8)$.

Local distribution of pixel values: Another way to describe the normalized patch is to capture the distribution of pixel values in the region (for example by a histogram). Both grey values or colour bands could be used. Colour histograms are particularly popular in image retrieval [141]. The discriminative power of the pixel value distribution is lower than the previously mentioned methods, since all the spatial relationships between pixels are discarded.

SIFT descriptor: Lowe [85] proposed the gradient orientation histogram, which retains coarse spatial information. The SIFT (Scale Invariant Feature Transform) descriptor is illustrated in figure 2.8. To achieve rotation invariance, all gradients within the patch are computed relative to a dominant gradient orientation, which is obtained as the highest peak in a histogram of all gradient orientations within the patch.

The SIFT descriptor has been experimentally shown [94] to outperform other descriptors like steerable filters [92], complex filters [121], and cross-correlation on raw pixel intensities, in the context of matching affine covariant regions. This is partially because of its high dimensionality (compared to e.g. filter responses) and partially because it is tolerant (due to the coarse spatial grid) to small localization errors, which often occur.

2.1.6 Representing 3D objects using local patches

The work of Schmid and Mohr [126] and Lowe [85], reviewed in section 2.1.3, deals mostly with representing and matching close-to-planar objects or single views of a 3D object. Approaches for representing 3D objects using local patches include those of Lowe (2001) [86], Rothganger *et al.* (2003) [112] and Ferrari *et al.* (2004) [48]. Lowe and Ferrari *et al.* represent 3D objects by a collection of images with known multiple view region correspondences. An example is shown in figure 2.9. Lowe first clusters similar model views of an object and establishes region matches (links) between adjacent view-clusters. In recognition, region matches cast votes in all linked views. The result is increased robustness, as votes are not accidentally spread across close-by views. Ferrari *et al.* take the integration between model views even further. In recognition, region matches to a test image are not only propagated to adjacent model views but their mutual consistency is also verified. A different approach is taken by Rothganger *et al.* [112]. An explicit 3D model is built from a collection of still images reconstructing the structure of the object. An example is shown in figure 2.10. During recognition, geometric constraints are applied. Note that in all of the above methods, model images are assumed to have a clean background.

In chapter 4, we present a method for automatically associating different views of a 3D object from video. In our case, we do not enforce global 3D consistency – the 3D object is represented implicitly by a set of exemplar images, and this loose coupling allows a degree of deformation (e.g. for facial expressions). Also, we build this object model automatically from video shots despite background clutter. Recently, a similar idea of object model building from video has been explored by Rothganger *et al.* (2004) [113], but the focus is more on model building rather than matching, recognition and retrieval, and only rigid objects are considered.

Other approaches to building appearance models from video include that of Mahindroo *et al.* [90], where optic-flow based motion segmentation is used to extract objects from video, and that of Wallraven and Bulthoff [169], where an object is modelled by selecting keyframes (using point tracking) from sequences of single objects (some of which are artificial).

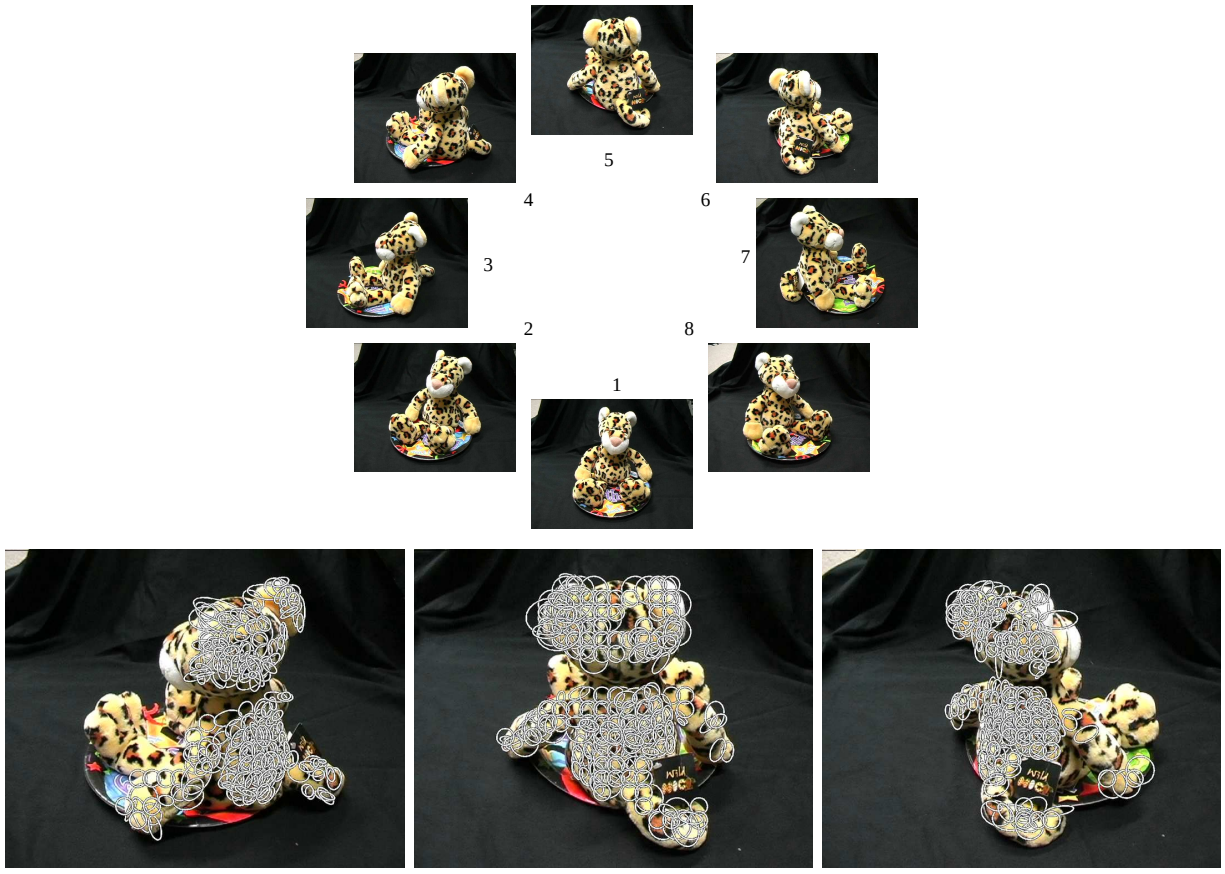


Figure 2.9: Representing a 3D object using multiple views with known region correspondences in Ferrari *et al.* [48]. Top: Model views of the object. Bottom: Dense region correspondences between three views.

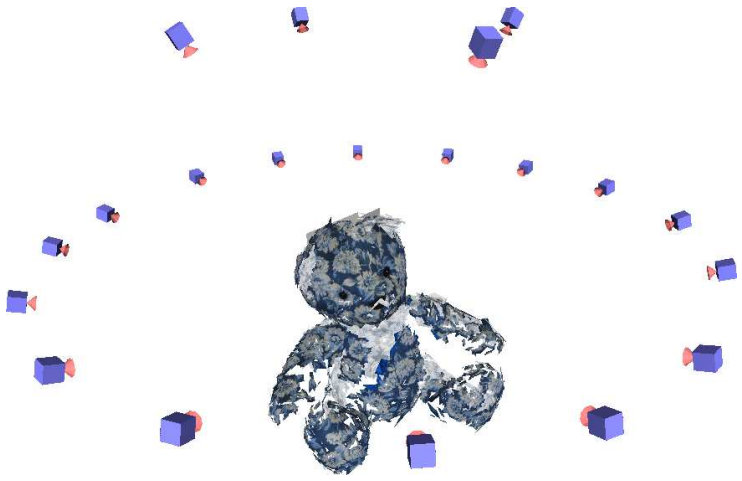


Figure 2.10: Rothganger *et al.* [112] represent 3D objects by local patches with explicit 3D structure.

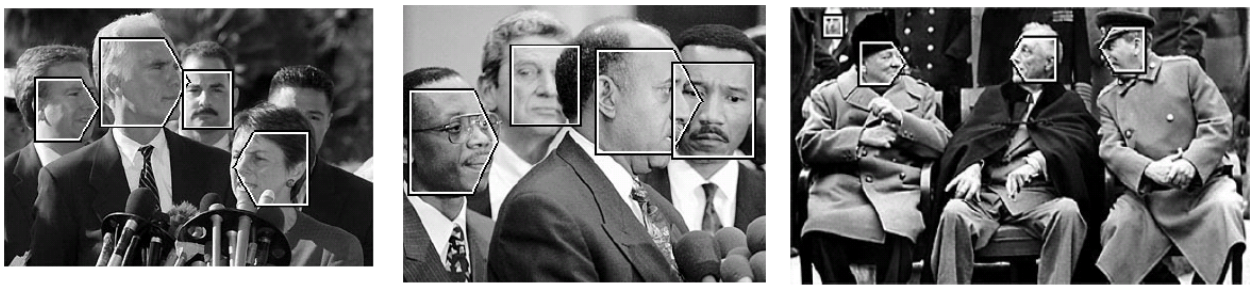


Figure 2.11: Examples of successful face detections by the Schneiderman and Kanade detector [127].

2.2 Recognition of object categories

In this section we review previous work on object class recognition. The challenge is that appearance variations among instances of an object class have to be modelled, in addition to standard problems of viewpoint and lighting changes and partial occlusion. It should be noted though that currently, very few methods are trying to address all of the above issues. In contrast to object instance recognition, which is reaching some maturity, object class recognition is still a very active research area, with many different competing, and sometimes complementary, approaches. Approaches reviewed below are broadly divided into: (i) sliding window classifiers, (ii) geometry-free methods, and (iii) methods and models with strong geometrical structure.

2.2.1 Sliding window classifiers

These methods involve training a classifier, which for a small image patch, say 24×24 pixels [168], decides whether the desired object (e.g. a face) is present. Given a test image, such a classifier is then applied within a 'sliding window', over a range of translations and scales. Extracted image features (image measurements), and the form of the classifier, vary considerably. Training the classifier usually requires many tightly cropped training images (with both object present and absent). The task of the classifier is to capture the intra-class variations present in the training data.

An example of a sliding window approach is the work of Schneiderman and Kanade [127], where extracted image features are wavelet decomposition coefficients. The classifier is a probabilistic decision (likelihood ratio) based on modelling statistics of image features in both foreground (object present) and background (object absent) images. The method is applied to multi-aspect detection of

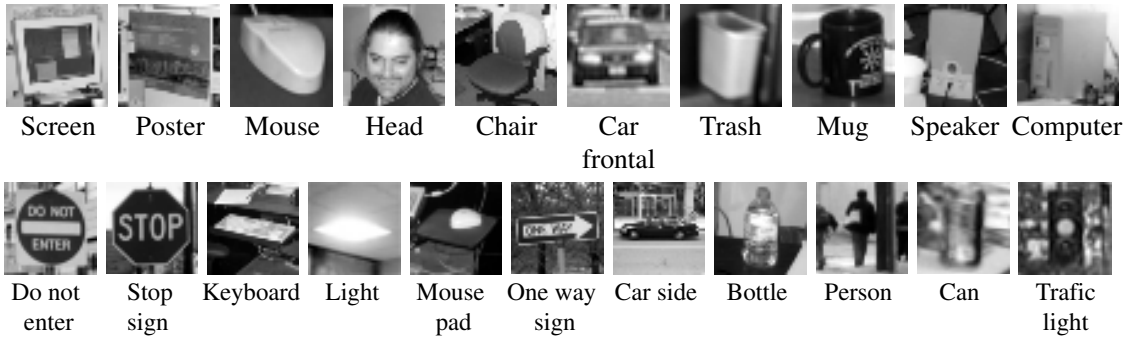


Figure 2.12: 21 object classes learnt by Torralba *et al.* [154].

faces and cars by training a separate detector for each aspect. For example, two separate detectors are trained for faces: one for profile faces and one for frontal faces. Examples of successful face detections are shown in figure 2.11.

Another outstanding sliding window approach was presented by Viola and Jones [168]. Their contribution was a very fast frontal face detector. The key enabling ideas are: (i) features based on sums of pixel values in rectangular image regions, which can be computed very efficiently using an integral image, and (ii) a cascade of detectors with increasing complexity, where only image windows likely to contain faces are passed to more complex classifiers further down the cascade. Each stage of the cascade is a classifier trained using AdaBoost.

Sliding window approaches have also been applied for detecting humans [30, 95] and side-views of cars [4]. Note that the above mentioned approaches were designed and tuned to detect one object category. More recently the focus has shifted to more general methods, which would be capable of learning many object categories without the need for significant modifications and tuning, and perhaps requiring a smaller amount of supervision. Several such approaches are reviewed next. One notable sliding window method, applied to 21 object classes (shown in figure 2.12), is the work of Torralba *et al.* [154], where the authors propose to share features (weak classifiers in the context of boosting) between object classes. The result is reduced computational complexity, and a smaller number of training images required to achieve given performance.

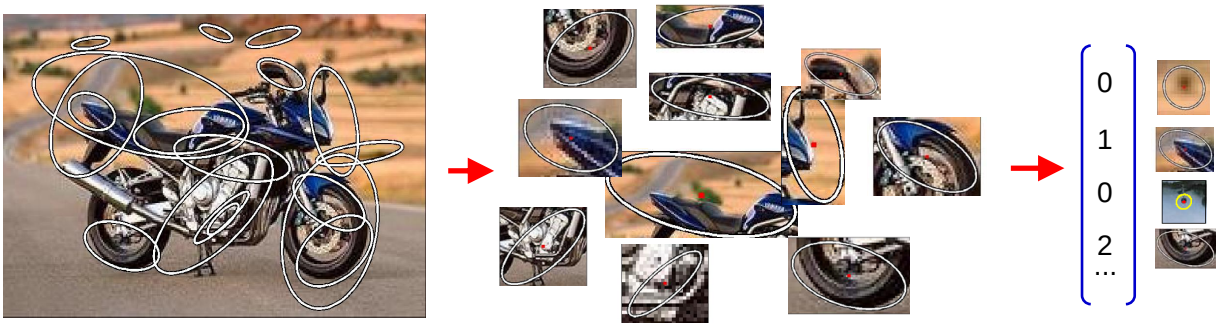


Figure 2.13: Illustration of the 'bag of visual words' model. An image is represented by a histogram of quantized local regions – visual words. Note that all spatial relations between regions are lost.

2.2.2 Geometry-free methods

The approaches of Csurka *et al.* [29], Opelt *et al.* [104] and Zhang *et al.* [178] represent images using histograms of quantized appearances of local patches – the 'bag-of-words' model. In learning, local regions are extracted from all training images and quantized into a codebook (visual vocabulary). Each image is then represented by a vector indicating the number of occurrences of each visual word. The bag-of-words image representation is illustrated in figure 2.13. A classifier is then trained to predict the presence/absence of an object in novel images, which are also described by a vector of visual word occurrences. Local features extracted are usually scale or affine covariant regions, described by SIFT descriptors. Classifiers used are usually Support Vector Machines, variants of boosting or nearest neighbour.

The advantage of 'bag-of-words' methods is their simplicity and the relatively small amount of supervision required. Labelling training data only requires indicating the presence/absence of an object in the image. No manual object segmentation or bounding box specification is needed. On the downside, these methods classify entire images, and are unable to localize objects. Experimentally, 'bag-of-words' models achieved excellent results on several datasets. In recent comparison on very difficult real-world data, consisting of ten object classes [162] (examples are shown in figure 2.14), the 'bag-of-words' methods have demonstrated a very good performance. This is somewhat surprising, given the lack of any spatial structure in the model – a point we return to in chapter 6.

Methods [29, 104] were developed independently of, and after, our object instance retrieval

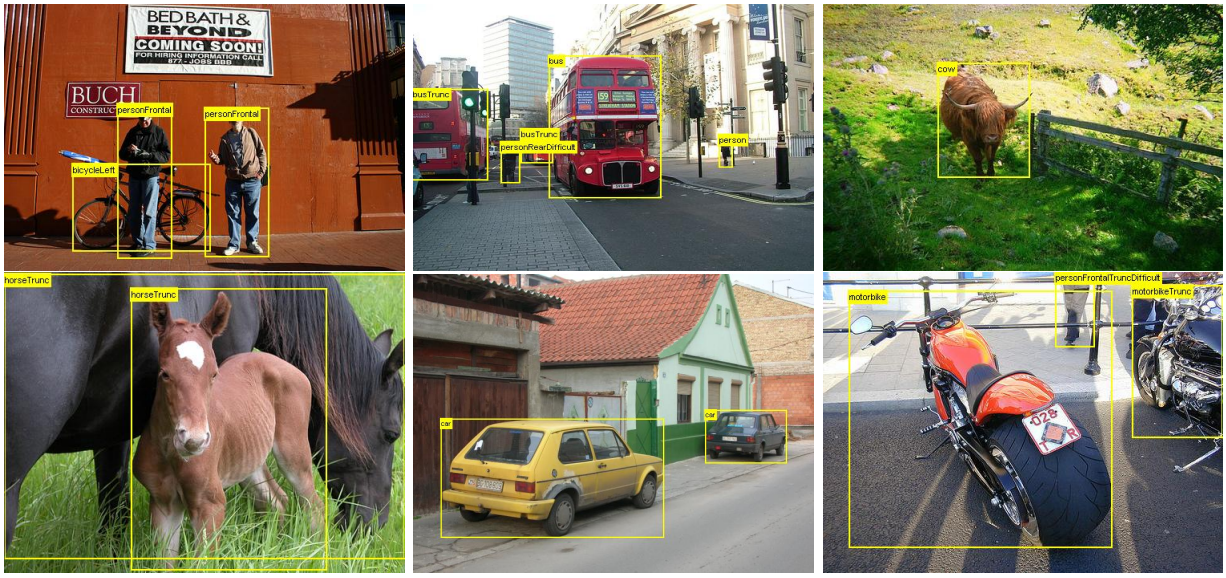


Figure 2.14: Example images with annotations from the Pascal Visual Object Class Challenge 2006. The dataset contains 10 object classes: bicycle, bus, car, cat, cow, dog, horse, motorbike, person, and sheep. In total there are 5,304 images, containing 9,507 annotated objects.

algorithm, also based on the 'bag-of-words' model, presented in chapters 3 and 4. Our object class retrieval algorithm described in chapter 6 also uses 'bag-of-words' image representation, but investigates the unsupervised scenario, where no labelled training data is available.

2.2.3 Geometry based methods

Fergus *et al.* [45] model object classes as probabilistic constellations of parts similar to the pictorial structure model of Fischler and Elschlager [50]. The appearance of each part, as well as pair-wise relations between parts, are modelled using Gaussian distributions. The model can be nicely visualized as a collection parts connected by springs, so that parts can move with respect to each other. Model parameters are automatically estimated by maximizing the likelihood of the training data known to contain instances of the desired object class. Bounding boxes or the manual segmentation of the training data is not required. The learning algorithm automatically looks for a configuration of detected image regions consistent over the training data. Recognition proceeds by first detecting potential part locations in an image, and then comparing hypotheses as to whether observed features are generated by the category model or by the background model. The constellation model was shown to perform well on several diverse object categories with varying scale and background

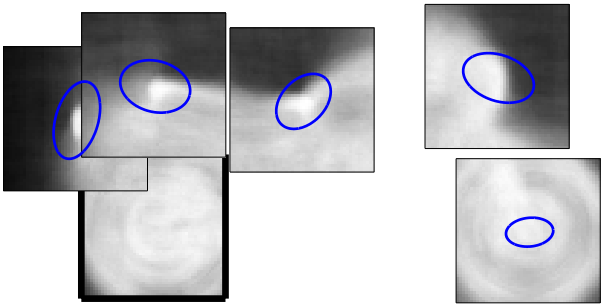


Figure 2.15: A motorbike pictorial structure model from Crandall *et al.* [28]. Square tiles illustrate appearance models for each part. Ellipses illustrate the location variances with respect to the rear-wheel part.

clutter, but only single aspects of objects were considered. In contrast to 'bag-of-words' models, the constellation model can localize objects in images. The downside of the model is the computational complexity. When mutual positions of all parts are modelled, the recognition is intractable. In more detail, evaluating a model with n parts in an image with h detected potential feature locations has complexity $O(h^n)$. This complexity can be reduced for models with simpler spatial structure, where spatial relations are modelled only between some parts. As was experimentally shown in the work of Crandall *et al.* [28], the resulting loss in performance is small. An example pictorial structure model is shown in figure 2.15. Pictorial structure models were also applied for detecting humans and their pose in images and videos by Felzenszwalb and Huttenlocher [43] and Ramanan *et al.* [110]. A Bayesian extension of the constellation model of Fergus *et al.* [45], capable of learning from a small number (3-5) of training images, was presented by Fei-Fei *et al.* [40]. This work shows that knowledge about other object classes, here in the form of a prior, can help in learning new object class models.

Leibe and Schiele [79] model objects as collections of local quantized patches. The spatial model is a non-parametric distribution, capturing the locations of parts relative to the object centroid. Learning an object class model consists of building the category specific codebook and the non-parametric spatial model. In recognition, detected features in a novel image are matched to codebook entries, which in turn cast votes for the object centroid. The method was experimentally shown to perform very well on the object localization task, and can deal with scale variations, background clutter, and some amount of partial occlusion. Currently, only single-aspect object de-

tection results have been shown. Learning requires specifying a bounding box around each training object. The Leibe and Schiele method is capable of object segmentation, provided segmentation is specified for the training data.

Berg and Malik [15] formulate the object category recognition problem as deformable shape matching to object exemplars. The idea is that instances of the same object class should have similar shapes, and that therefore the cost of deforming one instance into another should be low. Recognition is performed in a nearest neighbour framework. A deformation cost is computed between the novel image and all stored training examples, and the novel image is then assigned to the object class with the lowest deformation cost. To align two shapes, correspondences have to be found. The method works on object categories with prominent shape, and can deal with some amount of partial occlusion and background clutter. So far, only single-aspect object detection results have been shown.

Kumar *et al.* [75], Shotton *et al.* [131] and Opelt *et al.* [105] have also addressed the recognition of object categories with prominent shape (e.g. horses, cows, bottles). Objects are modelled as collections of 'object boundary fragments', related by a spatial model. Again, only single-aspect object detection results have been so far shown.

2.3 Codebooks in recognition

'Visual words', as used in this thesis, are quantized local appearance descriptors. In particular, we quantize descriptors of affine covariant regions and facial features to form a vocabulary for object and face recognition. In this section, we review other approaches that apply some form of 'codebook' in object recognition. In this section, 'codebook' is equivalent to 'visual vocabulary', and 'codeword' is equivalent to 'visual word'. The usual purpose of codebook generation in recognition is to establish potential correspondences between local image regions. In particular, after the codebook is generated, all local regions with the same codebook label are deemed matched. The 'granularity' of a codebook can be tuned to the required generalization: 'coarse' quantization can capture intra-class variations for object category recognition; 'fine' quantization is more suitable for object instance recognition. Another advantage is its efficiency; local patches in novel images are assigned codeword labels, and the original patch descriptors, possibly high dimensional, can be discarded.

2.3.1 Codebooks for Texture Classification – Textons

Leung and Malik [82] used quantized appearance descriptors, called 'Textons', for texture classification. Textons are the quantized responses of a filter bank applied densely over an entire image. Textons form the basic elements of texture description. Textures are represented by distributions of textons, which are compared using the χ^2 statistic. The goal of the work is to recognize textures captured from different camera viewpoints, and under varying illumination.

Varma and Zisserman [165] modified this approach by quantizing small image patches rather than filter responses. In both above approaches, viewpoint and lighting variations are addressed indirectly by including examples in the training data. Standard texture datasets contain only segmented images, and so robustness to background clutter and partial occlusion is not necessary.

Lazebnik *et al.* [78] address texture classification using quantized affine covariant regions. Textons are obtained independently for each image, and texton distributions are compared using the Earth Mover Distance [116]. Using affine covariant regions widens the applicable range of viewpoint changes.

In all the above approaches, quantization is performed using k-means clustering, and the number of clusters is chosen empirically to maximize classification performance. Methods are supervised starting from two sets of segmented images, one for training and one for testing.

2.3.2 Codebooks for object instance recognition

Chapter 3 of this thesis describes the application of quantized affine covariant regions for object instance retrieval from videos. A visual vocabulary (codebook) is built in an unsupervised way, without knowing the objects of interest. This is in contrast to texture classification, where usually, separate codebooks are built for each texture class. Retrieval results on entire movies are shown for several diverse objects, not just repeated textures. The proposed retrieval method is also designed to deal with partial occlusion and significant amounts of background clutter.

Recently, Nister and Stewenius [99], building on our work of chapter 3, demonstrated impressive retrieval results on a database of 50,000 CD covers. Their main idea is to apply hierarchical k-means clustering to obtain a vocabulary organized in a tree, rather than the 'flat' vocabulary obtained by simple k-means clustering.

2.3.3 Codebooks for recognition of object classes

Early approaches quantized square image patches around interest points. No scale or viewpoint invariance was attempted. Usually, a separate codebook was generated for each object class. Examples include: Weber *et al.* [171], detecting faces and rear-views of cars; Agarwal and Roth [4], detecting side-views of cars; and more recently Leibe and Schiele [79], detecting several object classes including side-views of cars, motorbikes, cows, and pedestrians. All the above methods build spatial models, relating object parts having quantized appearance. The number of codebook entries is relatively low, around 100 for each class. The codebooks are generated using k-means [171], or agglomerative [4, 79] clustering.

The k-means algorithm was also used by Barnard *et al.* [9] and Duygulu *et al.* [34] to vector quantize measurements on image segments, obtained by normalized cuts segmentation [130]. The authors apply machine translation methods to learn the correspondence between textual annotations and a set of quantized image segments. The learned correspondences are then used for automatic image annotation and object recognition. The method suffers from the rather crude set of image features used.

Recent increased interest in using codebooks for object category recognition has been partly motivated by the success of 'bag-of-words' models. Initially, 'bag-of-words' models [29, 104], including the work in this thesis, created codebooks by quantizing local appearances of scale or affine covariant regions. Currently, 'dense' image representations are gaining momentum. Winn *et al.* [173] use texton-like object representation by quantizing filter responses computed at every pixel. Agarwal *et al.* [3], Jurie and Triggs [69] and Bosch *et al.* [20] quantize densely sampled overlapping square or circular patches at multiple scales. Dense representations can represent regions with a lack of local texture, like sky or water, and might be more suitable for subsequent object segmentation. 'Soft' quantization, where an image patch is assigned to multiple codebook entries with different weights, has also been recently investigated [3].

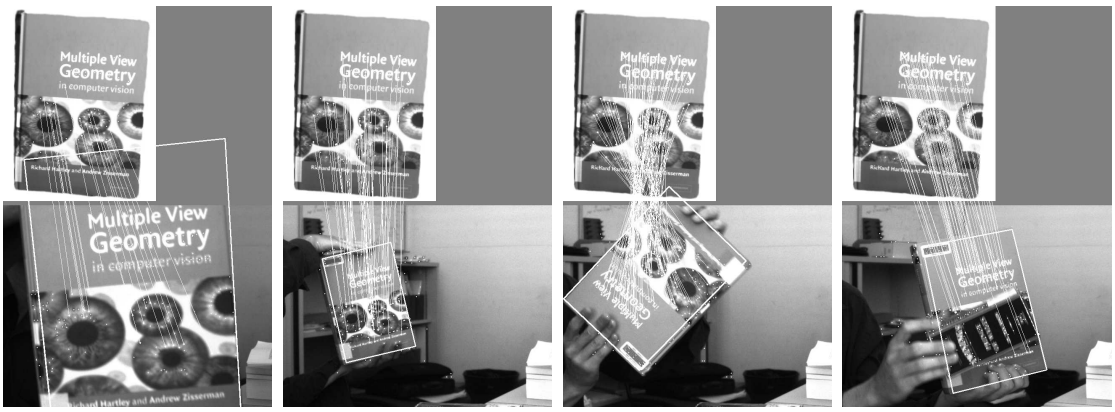


Figure 2.16: Real-time object recognition in a video sequence by Lepetit *et al.* [80].

2.4 Efficient indexing for object recognition

In this section, we review recent work on efficient indexing applied to object recognition. In particular, we focus on object instance recognition using local regions.

Lowe [85, 87] uses an approximate nearest neighbour search to find matching 128-dimensional SIFT descriptors in a database of model objects. The search is implemented using a k-d tree, with a best-bin-first backtracking strategy. Finding the exact nearest neighbour is not guaranteed, but experimentally the method achieves significant (up to two orders of magnitude) speed-up, with a small loss in performance. More recently, decision tree based classifiers have been applied for indexing local patches, and these methods are described next.

Lepetit *et al.* (2005) [80] build a classifier to index local regions. In particular, several randomized decision tree classifiers are trained using local regions extracted from synthesized object images depicted under varying pose and lighting conditions. The image measurements taken at each node of the tree are differences of intensities of two pixels within the patch. The method achieves real-time (25Hz) recognition of a small number of objects (2-3) in a 640×480 pixel video. The off-line training takes about 15 minutes, and each new object requires re-training. The results are illustrated in figure 2.16.

Obdrzalek and Matas (2005) [101] also use a binary decision tree to index local regions. Their tree is built in an unsupervised way, without knowledge of training object identities or region correspondences. After an off-line training stage, the method supports near real-time recognition

of hundreds of objects.

2.5 Image retrieval

Early image retrieval (see for example the review by Smeulders *et al.* [141]) focused on the retrieval of entire images. Given a query image, the goal was to retrieve entire scenes (e.g. sunset over sea) or it was assumed that images contain a single object occupying most of the image. Background clutter or partial occlusions were not explicitly handled. Image measurements were usually quite basic, for example global colour and texture statistics. Intra-class variations, camera viewpoint or illumination variations were usually not explicitly modelled. Later, some approaches tried to extract 'objects' from images by segmenting them into regions with coherent image properties like colour or texture. An example is 'Blobworld' by Belongie *et al.* [13]. Such systems enjoyed only limited success, because: (i) bottom-up segmentation rarely produces segments which would correspond to objects; (ii) segments were described by a rather crude set of measurements, e.g. global colour, area, centroid, or eccentricity. Recently, affine covariant regions have been applied to object instance retrieval in a database of buildings by Van Gool *et al.* [163], or a database of 360 mainly planar objects by Obdrzalek and Matas [100]. Our work described in chapter 3 also uses affine covariant regions for object retrieval. Our contribution lies in quantizing region descriptors into 'visual words', and applying indexing methods from text retrieval.

Currently, the focus of the image retrieval community seems to be shifting to extracting 'concepts' (like people, objects, places, or events) using machine learning and statistical analysis methods. Such 'concepts' (similar to 'object categories' in computer vision) could then be used for indexing visual information, and could replace basic image measurements like colour. Another direction being explored is exploiting additional modalities available with visual information, e.g. text surrounding images on the Internet, or audio and close captions available with video. Example works in this direction are by Smith *et al.* [142] and Snoek *et al.* [143].

TRECVid: Significant efforts in the image and video retrieval community focus on evaluation. An example is the annual TRECVid [159] conference, which is essentially a large scale benchmarking competition of video retrieval systems. The data consists of about 200 hours of broadcast news

footage, and has recently been expanded to include BBC production footage. The goal is first to detect given 'concepts' in the video, and then to use these for a visual search for a given 'search topic'. Examples of concepts are: *Boats/Ships, Madeleine Albright, Bill Clinton, Trains, Beach, Basket score – ball passing down the hoop and the net, Airplane taking off, People walking or running, Physical violence between people and/or objects, Road – any size, paved or not.* The range of search topics also varies from object instance recognition, e.g. "Find shots zooming in on the US Capitol dome", to more general queries like "Find shots of one or more people and one or more dogs walking together".

Overall, the focus seems to be on establishing whether a concept is present/absent in the video, not necessary localization. Methods usually involve a limited amount of visual processing, as the emphasis is on combining visual information, with audio and text obtained from automated speech recognition. In some cases, object detectors developed in the computer vision community are applied, for example face or car detectors. The emphasis of the visual search task is on browsing. The user is given a limited amount of time to manually browse the video collection and label relevant shots. The challenge seems to be in building suitable video browsing interfaces, rather than in object recognition.

2.6 Text retrieval and statistical text analysis

Here we review ideas and methods used in text retrieval and statistical text analysis. First we review a standard text retrieval system, and then focus on probabilistic topic discovery models.

2.6.1 A standard text retrieval system

Text retrieval (or information retrieval) [8, 175] systems deal with the organization, storage, indexing, and accessing of textual documents. Generally, text retrieval systems, Google being an example of one, employ a number of standard steps [8]:

Text preprocessing: The input for a text retrieval system is usually text, including tags and formatting strings, e.g. tex, html or postscript files. Therefore, the input file has first to be parsed into words. The text is also stripped of special characters like digits, hyphens or punctuation marks.

occurrences	word
692	and
660	the
597	i
548	to
471	a
382	of
374	my
352	that
340	is
316	in
289	you
278	thou
264	me
258	not
246	with
230	it
222	for
217	this
211	be
178	thy

Table 2.1: The top 20 most frequent words from the play 'Romeo and Juliet' by William Shakespeare.

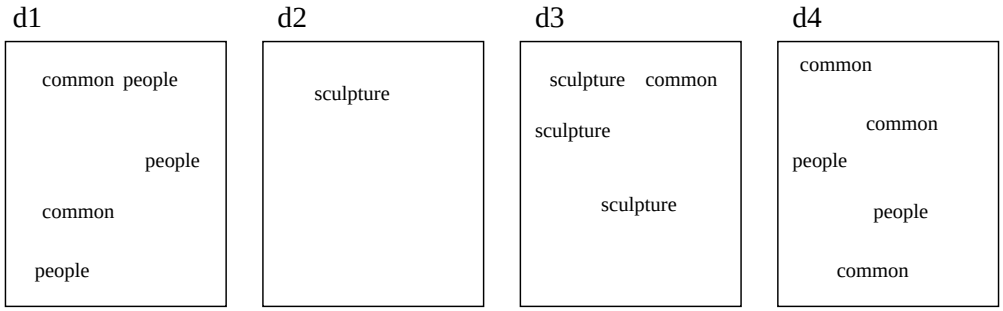
Removing stop-words: 'Stop-words' are words that are present in almost all documents in the database and thus provide almost no discrimination power. Stop-words are usually removed from the text. A stop-list also has the benefit of reducing the size of indexing structures.

The top 20 most frequent words from the play 'Romeo and Juliet' by William Shakespeare are shown in table 2.1. The play contains 25,591 words and is formed by 3,380 distinct terms. More interestingly, the top 20 most frequent words constitute 29% of the total number of words in the play.

Vector model: After stop-word removal, all remaining V words (terms) are assigned a unique identifier, i, where $i = 1, \ldots, V$, and each document is represented by a V-vector,

$$\mathbf{v}_d = (t_1, ..., t_i, ..., t_V)^\top \tag{2.4}$$

where in the simplest case t_i is either 0 or 1, depending whether the term i is present or absent in the document. Similarity between documents is measured by the normalized scalar product (cosine of angle) of their representing document vectors. During retrieval, documents are ranked according



(a)

Term	**List of hits** (occurrences in documents)
People	[d1:hit hit hit], [d4:hit hit] …
Common	[d1:hit hit], [d3: hit], [d4: hit hit hit] …
Sculpture	[d2:hit], [d3: hit hit hit] …

(b)

Figure 2.17: Illustration of the inverted file structure. (a) A set of four documents with highlighted occurrences of three words. (b) The inverted file for the three words.

to their similarity. A more elaborate weighting scheme called *tf-idf* is based on examining word frequencies (number of word occurrences) in the documents. The *tf-idf* weighting scheme is further discussed in chapter 3.

Indexing with inverted files: An inverted file [175] is a commonly used indexing structure in text retrieval. It is structured like a complete book index. It has an entry for each word in the corpus, followed by a list of all hits (occurrences) in all documents. A hit contains the position in the document, and possibly some other information like word type (upper case, lower case, title), font size, etc. Figure 2.17 illustrates the basic idea of the inverted file structure.

Proximity, type and PageRank weights: The web search engine Google [160, 21] ranks retrieved pages based on a combination of three weighting schemes: proximity, type and PageRank weights.

The idea of proximity weights is to increase the weighting of words appearing close together in the query and retrieved documents. Refer again to figure 2.17. If the query contained the words 'common people', then occurrences of these two words appearing close together would be weighted

more, and consequently document d1 would be ranked higher than document d4. Since text is a linear structure with a well defined ordering, the proximity of two words is easy to implement. Google [21] uses ten bins to classify word proximity, ranging from 'a phrase match' to 'not even close'. Type weights [21] reflect the importance of a particular word in the document, e.g. whether the word is part of a title, URL, anchor, or based on its relative font size in the document. The PageRank is a score assigned to each web page based on its importance, where importance is computed from the link structure of the World Wide Web. Web pages which have many links pointing to them are assigned more weight than those with fewer links. In addition, not all links count the same. Links from higher ranked pages 'weigh' more.

2.6.2 Topic discovery models

The motivation for fitting a topic discovery model to a collection of text documents is twofold. First, we might be interested in the topics themselves, as we might want to discover the common, important or relevant topics present in the collection. Second, the discovered topics might be useful for retrieval. For example, imagine querying with the word 'movie'. If only this word is used as a query, many relevant documents containing the word 'film' (and not 'movie') will be missed. However, if the query can be mapped onto a topic about movies, which contains both 'movie' and 'film', and possibly other words like 'Hollywood', 'actor' and 'star', many more relevant documents will be retrieved.

Typically, each document in a collection is represented using a vector of word occurrences, i.e. a 'bag-of-words' model. The entire collection of documents can then be represented by stacking all document vectors as columns side by side into a 'term-document matrix'. The goal is to discover some structure in the data (a set of topics) by analyzing co-occurrences of words in documents. The intuition is that if two words co-occur in many documents, they might be describing the same topic.

The initial work of Deerwester *et al.* [33] (1990) applied singular value decomposition (SVD) to the term document matrix, obtaining a set of basis document vectors (topics) and projection coefficients for each document. Hofmann (1999) [64, 65] devised a probabilistic version, called probabilistic latent semantic analysis (pLSA). The appeal of this method lies in its underlying

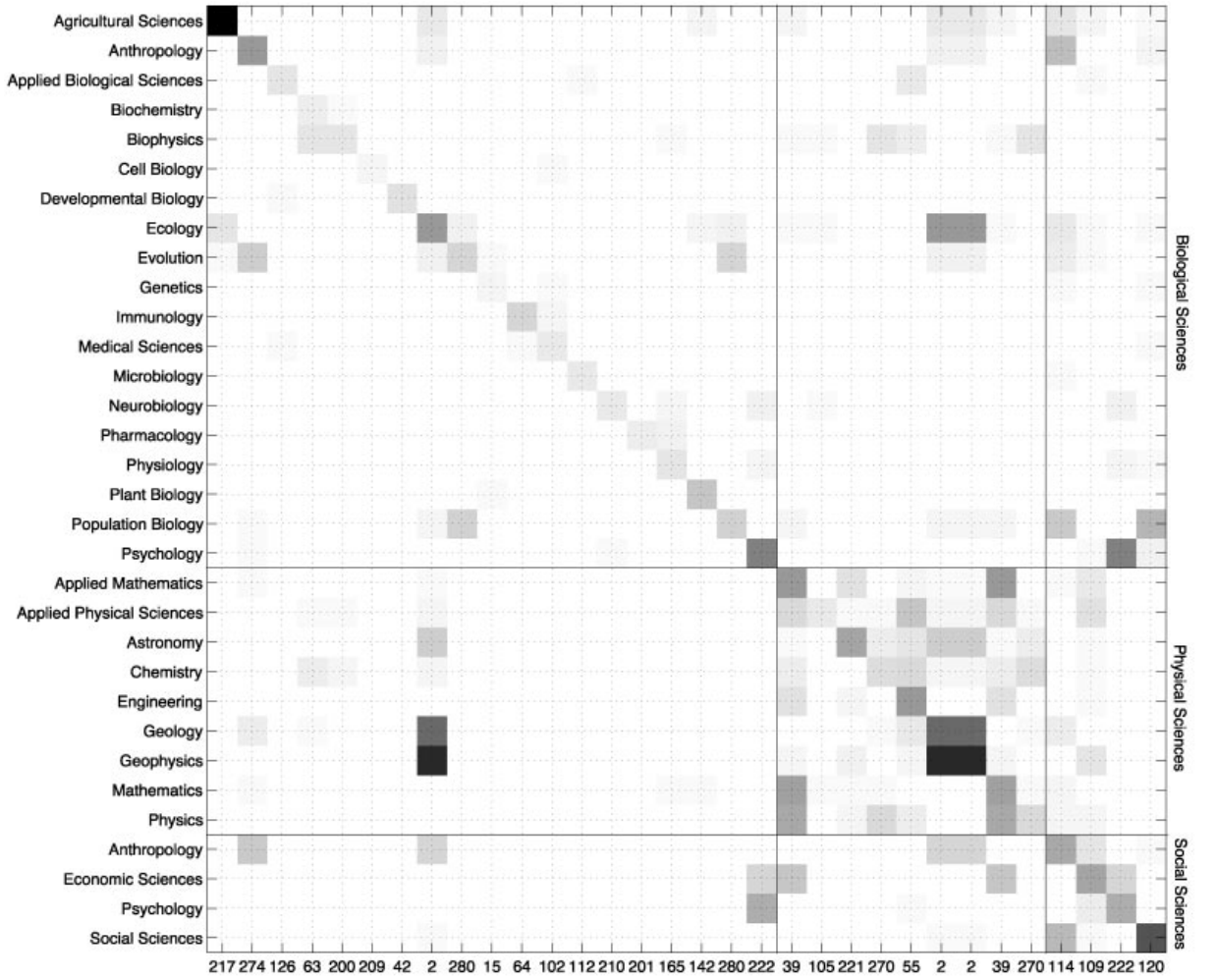
Aspect 1	Aspect 2	Aspect 3	Aspect 4
plane	space	home	film
airport	shuttle	family	movie
crash	mission	like	music
flight	astronauts	love	new
safety	launch	kids	best
aircraft	station	mother	hollywood
air	crew	life	love
passenger	nasa	happy	actor
board	satellite	friends	entertainment
airline	earth	cnn	star

Figure 2.18: Example of four discovered topics (out of 128) by probabilistic latent semantic analysis (pLSA, Hofmann [65]) in a collection of 16,000 text documents. Each topic is shown by the 10 most probable words (from top).

probabilistic generative model, which describes how the entire document collection is sampled from an underlying set of topics. Topics are distributions of words, and each document is a mixture of topics with mixing weights particular to that document, so that different documents can have different proportions of each topic. Given a particular collection of documents, the goal is to 'reverse' the generative process and estimate the model parameters: the set of topics and the document specific mixing weights. Fitting the pLSA model is discussed in more detail in chapter 6.

Figure 2.18 shows four topics discovered using pLSA in a collection of about 16,000 text documents. Note for example, that topics (aspects) 3 and 4 both contain the word 'love', but in different contexts. In this case, 'love' appears in the context of 'home' and 'family' (topic 3) and in the context of 'film' and 'music' (topic 4). Note also that topic 4 contains both 'movie' and 'film', which are the two synonyms mentioned in the retrieval example at the beginning of this section.

A Bayesian extension of the pLSA model, called Latent Dirichlet Allocation (LDA), was described by Blei *et al.* [18]. Griffiths and Steyvers [57] applied the Latent Dirichlet Allocation model to discovering topics in scientific abstracts. An example is shown in figure 2.19. Note how the statistical model can capture some amount of the semantic content of documents. For example, documents (abstracts) of related scientific disciplines are described by similar topics (e.g. 'Applied Mathematics', 'Mathematics' and 'Physics' have all high correlation with topic 39).



217	274	126	63	200	209
INSECT	SPECIES	GENE	STRUCTURE	FOLDING	NUCLEAR
MYB	PHYLOGENETIC	VECTOR	ANGSTROM	NATIVE	NUCLEUS
PHEROMONE	EVOLUTION	VECTORS	CRYSTAL	PROTEIN	LOCALIZATION
LENS	EVOLUTIONARY	EXPRESSION	RESIDUES	STATE	CYTOPLASM
LARVAE	SEQUENCES	TRANSFER	STRUCTURES	ENERGY	EXPORT

42	2	280	15	64	102
NEURAL	SPECIES	SPECIES	CHROMOSOME	CELLS	TUMOR
DEVELOPMENT	GLOBAL	SELECTION	REGION	CELL	CANCER
DORSAL	CLIMATE	EVOLUTION	CHROMOSOMES	ANTIGEN	TUMORS
EMBRYOS	CO2	GENETIC	KB	LYMPHOCYTES	HUMAN
VENTRAL	WATER	POPULATIONS	MAP	CD4	CELLS

112	210	201	165	142	222
HOST	SYNAPTIC	RESISTANCE	CHANNEL	PLANTS	CORTEX
BACTERIAL	NEURONS	RESISTANT	CHANNELS	PLANT	BRAIN
BACTERIA	POSTSYNAPTIC	DRUG	VOLTAGE	ARABIDOPSIS	SUBJECTS
STRAINS	HIPPOCAMPAL	DRUGS	CURRENT	TOBACCO	TASK
SALMONELLA	SYNAPSES	SENSITIVE	CURRENTS	LEAVES	AREAS

Figure 2.19: Some topics discovered by Griffiths and Steyvers [57] in a collection of about 30,000 scientific abstracts. The matrix (top) shows correlation between the true topics (rows) and discovered topics (columns). Dark elements indicate high correlation. Some topics are illustrated (bottom) by the five most probable words. For example, discovered topic 217 (the first column) has high correlation with the true topic 'Agricultural Sciences' (the first row) and the highly probable words for topic 217 are, for example, 'insect' and 'larvae'.

Chapter 3

Efficient visual search for particular objects

3.1 Introduction

This chapter is concerned with the efficient retrieval of particular objects in video given a single image of the object as a query. The goal is to return a ranked list of keyframes or shots containing occurrences of the query object, in a similar manner to the way in which Google returns text documents. Objects are represented as collections of local affine covariant regions, reviewed in section 2.1.4, so that retrieval can proceed successfully despite significant changes in viewpoint, illumination and partial occlusion.

We investigate whether text retrieval methods like the 'bag-of-words' model, document ranking, and inverted file indexing, reviewed in chapter 2, can be successfully employed for visual search in videos. For this we need a visual analogue of a textual word and we provide it by vector quantizing the local region descriptors. Images are then represented as 'bags-of-visual-words'. In addition to the text retrieval ranking techniques we also employ a ranking measure taking into account the spatial layout of visual words.

Object retrieval results are shown on feature length movies including 'Groundhog Day' [Ramis, 1993] and 'Casablanca' [Curtiz, 1942].



(a) (b)

Figure 3.1: **Object query example I.** (a) Top row: (left) a frame from the movie 'Groundhog Day' with an outlined query region and (right) a close-up of the query region delineating the object of interest. Bottom row: (left) all 1039 detected affine covariant regions superimposed and (right) close-up of the query region. (b) (left) two retrieved frames with detected regions of interest and (right) a close-up of the images with affine covariant regions superimposed. These regions match to a subset of the regions shown in (a). Note the significant change in foreshortening and scale between the query image of the object, and the object in the retrieved frames. For this query there are four correctly retrieved shots ranked 1, 2, 3 and 12. Querying all the 5,640 keyframes of the entire movie took 0.36 seconds on a 2GHz Pentium.

Chapter outline: In section 3.2 we describe the visual descriptors employed. Section 3.3 then describes how they are quantized into 'visual words' and sections 3.4 and 3.5 show how the text retrieval techniques are applied in the visual domain. Finally, in section 3.6, we evaluate the proposed approach on a ground truth set of six object queries. We investigate retrieval performance with respect to various visual vocabularies and compare the performance of several ranking measures. Performance is compared to a baseline method implementing standard frame to frame matching.

3.2 Viewpoint invariant object description

To achieve viewpoint invariant object retrieval we build on the work of viewpoint invariant image descriptors, as reviewed in section 2.1.4. Specifically, two types of viewpoint covariant regions are computed for each frame of the video. The first is constructed by elliptical shape adaptation [93, 121] about a Harris [59] interest point. This region type is referred to as Shape Adapted (SA). The second type of region is constructed by selecting areas from an intensity watershed image segmentation. The regions are those for which the area is approximately stationary as the intensity threshold is

varied. This region type is referred to as Maximally Stable [91] (MS). A more detailed description of both methods is given is section 2.1.4.

Two types of regions are employed because they detect different image areas and thus provide complementary representations of a frame. The SA regions tend to be centred on corner like features, and the MS regions correspond to blobs of high contrast with respect to their surroundings, such as a dark window on a grey wall. Both types of regions are represented by ellipses. These are computed at twice the originally detected region size in order for the image appearance to be more discriminating. For a 720×576 pixel video frame the number of regions computed is typically in the order of 1,200. An example is shown in Figure 3.1.

Each elliptical affine invariant region is represented by a 128-dimensional vector using the SIFT descriptor developed by Lowe [85]. In [94] this descriptor was shown to be superior to others used in the literature, such as the response of a set of steerable filters [93] or orthogonal filters [121], and we have also found SIFT to be superior (by comparing scene retrieval results against ground a truth [138]). One reason for this superior performance is that SIFT, unlike the other descriptors, is designed to be invariant to a shift of a few pixels in the region position, and this localization error is one that often occurs. Combining the SIFT descriptor with affine covariant regions gives region description vectors which are invariant to affine transformations of the image. Note, both the region detection and the description is computed on monochrome versions of the frames; colour information is not currently used in this work.

To reduce noise and reject unstable regions, information is aggregated over a sequence of frames. The regions detected in each frame of the video are tracked using a simple constant velocity dynamical model and correlation. The region tracker is described in more detail in the next chapter, section 4.2.1. Any region which does not survive for more than three frames is rejected. This 'stability check' significantly reduces the number of regions to about 600 per frame.

3.3 Building a visual vocabulary

The objective here is to vector quantize the descriptors into clusters which will be the visual 'words' for text retrieval. The vocabulary is constructed from a subpart of the movie, and its matching accuracy and expressive power are evaluated on the entire movie, as described in the following

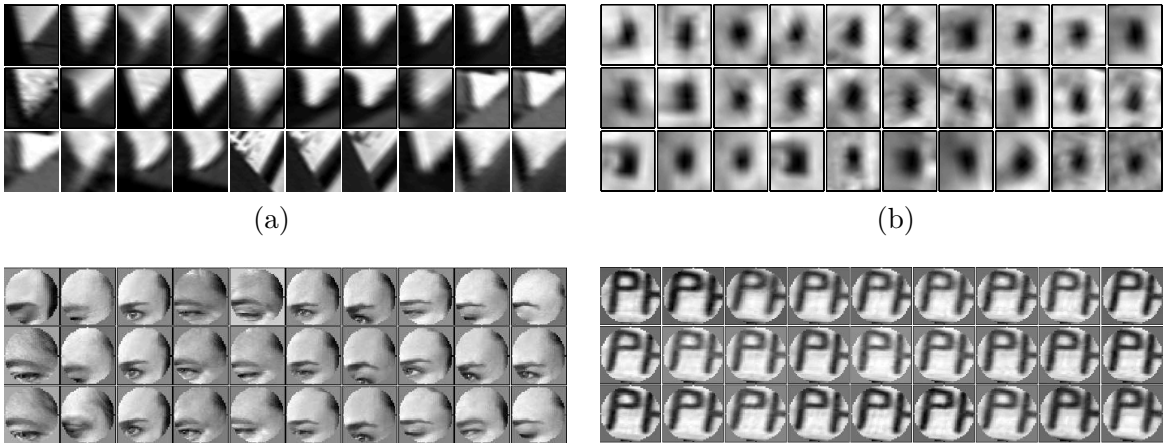


Figure 3.2: Samples of normalized affine covariant regions from clusters corresponding to a single visual word: (a,c,d) Shape Adapted regions; (b) Maximally Stable regions. Note that some visual words represent generic image structures, e.g. corners (a) or blobs (b), and some visual words are rather specific, e.g. an eye (c) or a letter (d).

sections. The running example is for the movie 'Groundhog Day'.

The vector quantization is carried out by K-means clustering, though other methods (K-medoids, histogram binning, mean shift, etc) are certainly possible.

3.3.1 Implementation

Each descriptor is a 128-vector, and to simultaneously cluster all the descriptors of the movie would be a gargantuan task. Instead, a random subset of 474 frames is selected. Even with this reduction there still remains around 300K descriptors that must be clustered.

Mahalanobis distance is used as the distance function for K-means clustering. The distance between two descriptors $\mathbf{x}_1$, $\mathbf{x}_2$, is then given by

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^\top \Sigma^{-1} (\mathbf{x}_1 - \mathbf{x}_2)}. \tag{3.1}$$

The covariance matrix Σ is determined by (i) computing covariances for descriptors throughout tracks within several shots, and (ii) assuming Σ is the same for all tracks (i.e. independent of the region) so that covariances for tracks can be aggregated. In this manner sufficient measurements are available to estimate all elements of Σ. The Mahalanobis distance enables the more noisy components of the 128–vector to be weighted down, and also decorrelates the components. Empirically

there is a small degree of correlation. As is standard, the descriptor space is affine transformed by the square root of Σ so that Euclidean distance may be used.

6,000 clusters are used for Shape Adapted regions, and 10,000 clusters for Maximally Stable regions. The ratio of the number of clusters for each type is chosen to be approximately the same as the ratio of detected descriptors of each type. The K-means algorithm is run several times with random initial assignments of points as cluster centres, and the lowest cost result used. The number of clusters was chosen empirically to maximize matching performance on a ground truth set for scene retrieval [138]. The object retrieval performance with respect to the number of clusters is tested on a new ground truth set for object retrieval in section 3.6.

Figure 3.2 shows examples of the regions which belong to particular clusters, i.e. which will be treated as the same visual word. The clustered regions reflect the properties of the SIFT descriptors which penalize intensity variations amongst regions less than cross-correlation. This is because SIFT emphasizes orientation of gradients, rather than the position of a particular intensity within the region.

The reason that SA and MS regions are clustered separately is that they cover different and largely independent regions of the scene. Consequently, they may be thought of as different vocabularies for describing the same scene, and thus should have their own word sets. In the same way as one vocabulary might describe architectural features and another the material quality (e.g. defects, weathering) of a building.

3.4 Visual indexing using text retrieval methods

Recall from section 2.6 that in text retrieval each document is represented by a vector of (weighted) word frequencies. In this section we briefly review the standard weighting that is employed, and the visual analogy of document retrieval to frame retrieval. The following sections describe the visual analogue of a stop list, and the method used to rank images based on the spatial layout of their visual words.

3.4.1 Term frequency–inverse document frequency weighting

The standard weighting [8] is known as 'term frequency–inverse document frequency' (*tf–idf*) and is computed as follows. Suppose there is a vocabulary of V words, then each document is represented by a vector

$$\mathbf{v}_d = (t_1, ..., t_i, ..., t_V)^\top \tag{3.2}$$

of weighted word frequencies with components

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{N_i}, \tag{3.3}$$

where n_{id} is the number of occurrences of word i in document d, n_d is the total number of words in the document d, N_i is the number of documents containing term i, and N is the number of documents in the whole database. The weighting is a product of two terms: the *word frequency*, n_{id}/n_d, and the *inverse document frequency*, $\log N/N_i$. The intuition is that the word frequency weights words occurring more often in a particular document higher (compared to word present/absent), and thus describes it well, whilst the inverse document frequency downweights words that appear often in the database, and therefore do not help to discriminate between different documents.

At the retrieval stage documents are ranked by the normalized scalar product (cosine of angle)

$$f_d = \frac{\mathbf{v}_q{}^\top \mathbf{v}_d}{\|\mathbf{v}_q\|_2 \ \|\mathbf{v}_d\|_2} \tag{3.4}$$

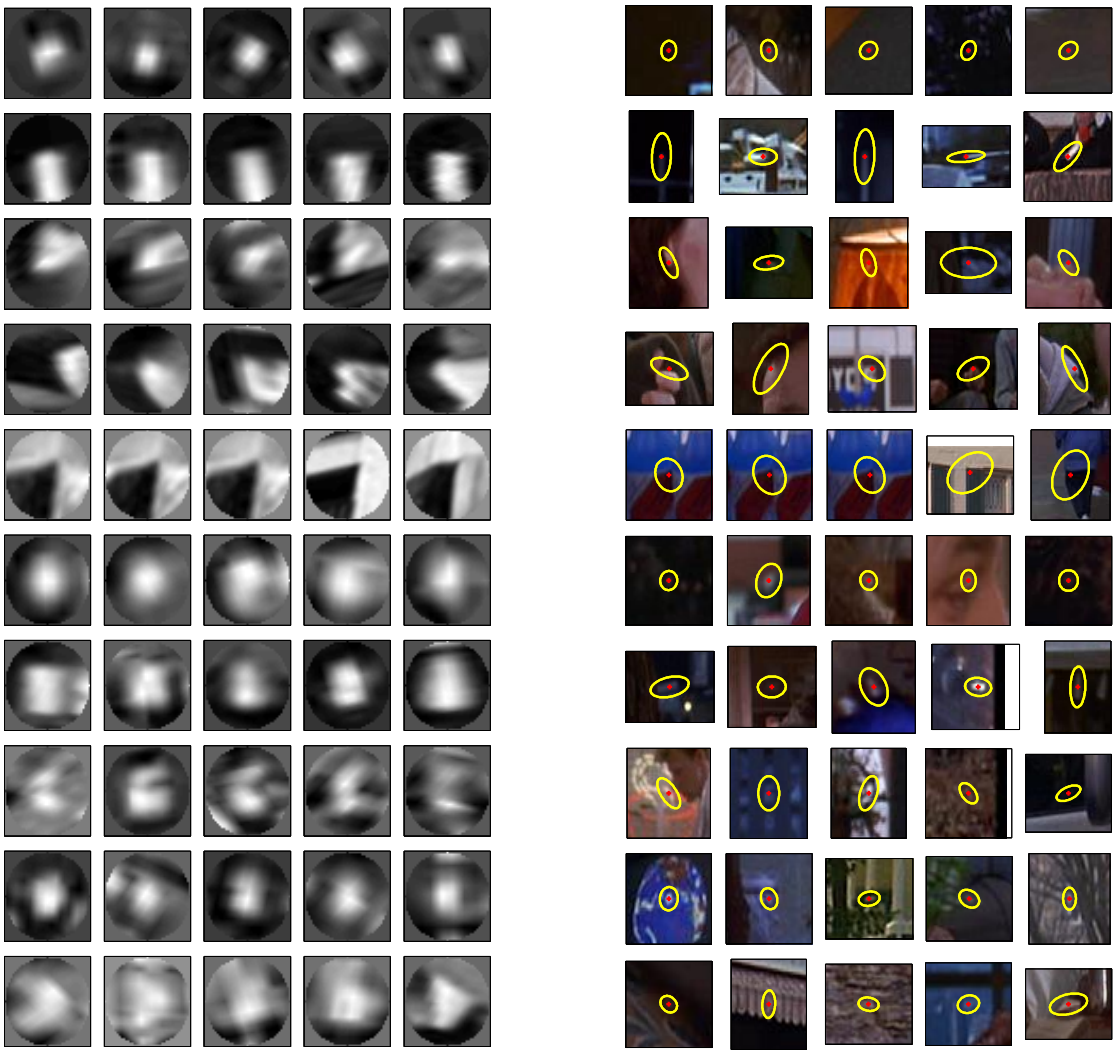
between the query vector $\mathbf{v}_q$ and all document vectors $\mathbf{v}_d$ in the database, where $\|\mathbf{v}\|_2 = \sqrt{\mathbf{v}^\top \mathbf{v}}$ is the L_2 norm of $\mathbf{v}$.

Note that if document and query vectors are pre-normalized to have unit L_2 norm, then (3.4) can be rewritten as

$$f_d = \mathbf{v}_q{}^\top \mathbf{v}_d = 1 - \frac{1}{2}\|\mathbf{v}_q - \mathbf{v}_d\|_2^2. \tag{3.5}$$

As a consequence of (3.5), sorting documents according to their ascending (squared) L_2 distance to the query vector produces the same ranking as sorting using the (descending) angle score (3.4).

In our case the query vector is given by the frequencies of visual words contained in a user specified sub-part of an image, weighted by the inverse document frequencies computed on the



(a) (b)

Figure 3.3: Examples of stop-listed visual words. Each row shows five random examples from one visual word. Normalized regions are shown in (a). Scale normalized close-ups of the corresponding elliptical regions overlaid over the original image are shown in (b).

entire database of frames. Retrieved frames are ranked according to the similarity of their weighted vectors to this query vector.

3.4.2 Stop list

Using a stop list analogy the most frequent visual words that occur in almost all images are suppressed. In our case the very common words are large clusters of over 2K points. These might correspond to small specularities (highlights), for example, which occur in many frames. The effect

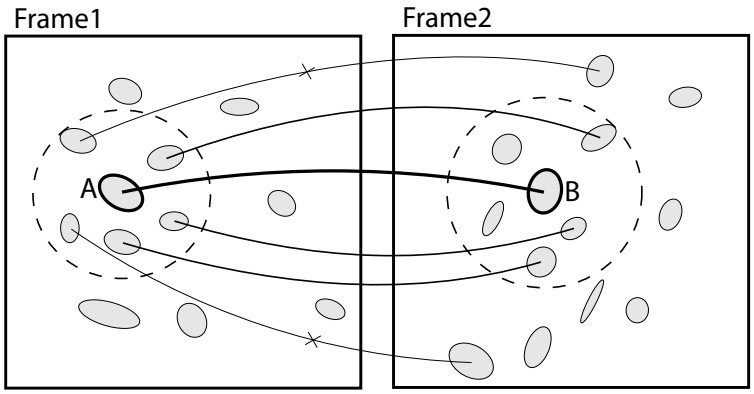


Figure 3.4: Illustration of spatial consistency voting. To verify a pair of matching regions (A,B) a circular search area is defined by the k (=5 in this example) spatial nearest neighbours in both frames. Each match which lies within the search areas in both frames casts a vote in support of match (A,B). In this example three supporting matches are found. Matches with no support are rejected.

of applying a stop list is evaluated on a set of ground truth queries in section 3.6.

Examples of stop-listed visual words are shown in figure 3.3. Figure 3.5 shows the benefit of imposing a stop list – very common visual words occur in many places in an image and can be responsible for mis-matches. Most of these are removed once the stop list is applied. The removal of the remaining mis-matches is described next.

3.4.3 Spatial consistency

Google increases the ranking for documents where the searched for words appear close together in the retrieved texts (measured by word order). This analogy is especially relevant for querying objects by an image, where matched covariant regions in the retrieved frames should have a similar spatial arrangement [120, 125] to those of the outlined region in the query image. The idea is implemented here by first retrieving frames using the weighted frequency vector alone, and then re-ranking them based on a measure of spatial consistency.

Spatial consistency can be measured quite loosely by requiring that neighbouring matches in the query region lie in a surrounding area in the retrieved frame. It can also be measured very strictly by requiring that neighbouring matches have the same spatial layout in the query region and retrieved frame. In our case the matched regions provide the affine transformation between the query and retrieved image so a point to point map is available for this strict measure.

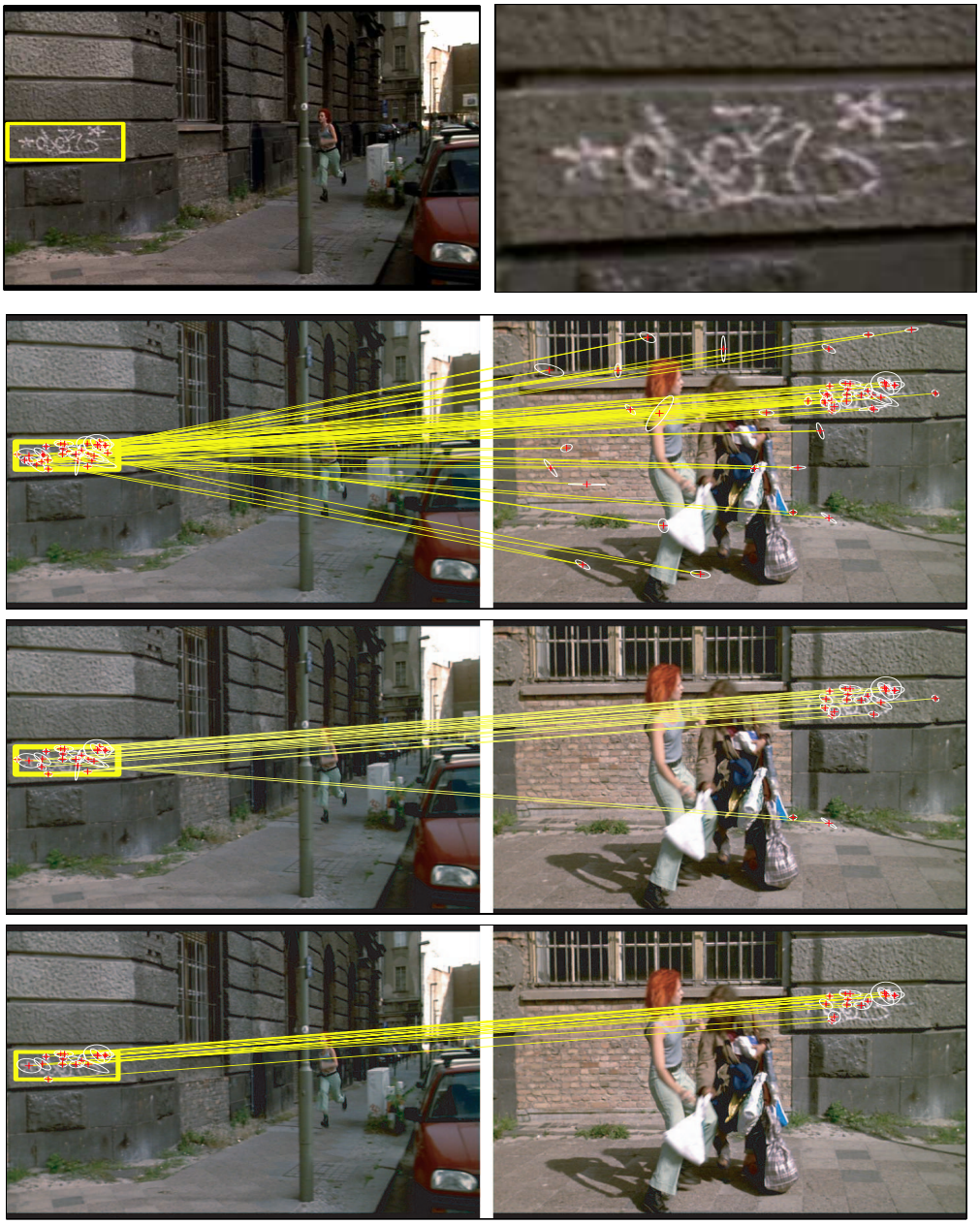


Figure 3.5: **Matching stages.** Top row: (left) Query region and (right) its close-up. Second row: Original matches based on visual words. Third row: matches after using the stop-list. Last row: Final set of matches after filtering on spatial consistency. The object retrieval algorithm is summarized in figure 3.8.

We have found that a good performance is obtained at the less constrained end of this possible range of measures. A search area is defined by the 15 nearest spatial neighbours of each match, and each region which also matches within this area casts a vote for that frame. Matches with no support are rejected. To discount repeated structures, which we found are responsible for many highly ranked false positives, matches with the same visual word label are not allowed to vote for each other, each match can accumulate at most one vote from one distinct visual word and one visual word is allowed to vote only once per image. The final score of the frame is determined by summing the spatial consistency votes, and adding the frequency score f_d given by (3.4). Including the frequency score (which ranges between 0 and 1) disambiguates ranking amongst frames which receive the same number of spatial consistency votes. The object bounding box in the retrieved frame is determined as the rectangular bounding box of the matched regions after the spatial consistency test. The spatial consistency voting is illustrated in figure 3.4. This works well as is demonstrated in the last row of figure 3.5, which shows the spatial consistency rejection of incorrect matches. The object retrieval examples presented in this chapter employ this ranking measure and amply demonstrate its usefulness.

Other measures which take account of the affine mapping between images may be required in some situations, but this involves a greater computational expense.

Suppressing repeated structures: Figure 3.6 demonstrates how the spatial consistency algorithm discounts repeated structures. Figure 3.7 shows the same algorithm applied on a pair of frames containing the same object, which is correctly matched. The two figures illustrate how the spatial consistency algorithm successfully discounts repeated structures while not discarding correct matches. Empirically we found that on our ground truth set of queries, introduced in section 3.6, this heuristic suppressing repeated structures improves retrieval performance. However, other strategies explicitly taking into account repeated structures might be required in some situations.

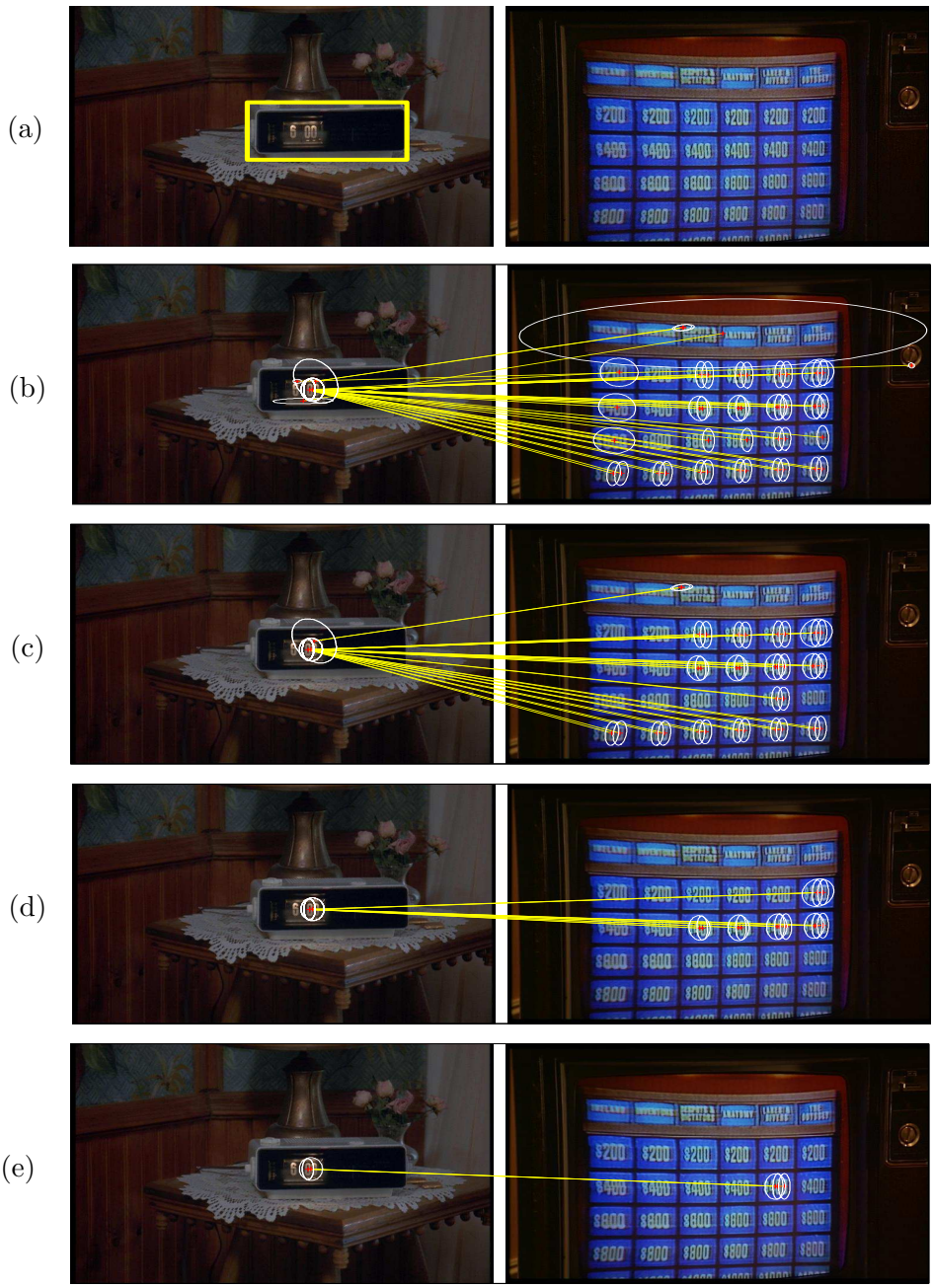


Figure 3.6: **Discounting repeated structures.** (a) Left: Frame with user specified query region. Right: Another frame which does not contain the user specified object ('Digital clock') but generates many false matches, due to similarity of the '00' on the clock to several repeated '00's on the TV screen. (b) Matches generated by visual words (after applying stop-list). (c)-(e) Stages of the spatial consistency check. In (c) all regions matches which do not have a supporting match within its 15th nearest neighbours in the image are removed. In (d) matches with the same visual word are not allowed to vote for each other. Note that all 'double' matches, generated by the same visual word representing the single '0', are removed. In (e) each visual word is allowed to vote only once per image. As a result of these checks the frame on the right hand side receives low number of spatial consistency votes and is correctly low ranked. Figure 3.7 shows the same spatial consistency algorithm applied to a pair of frames containing the same object.

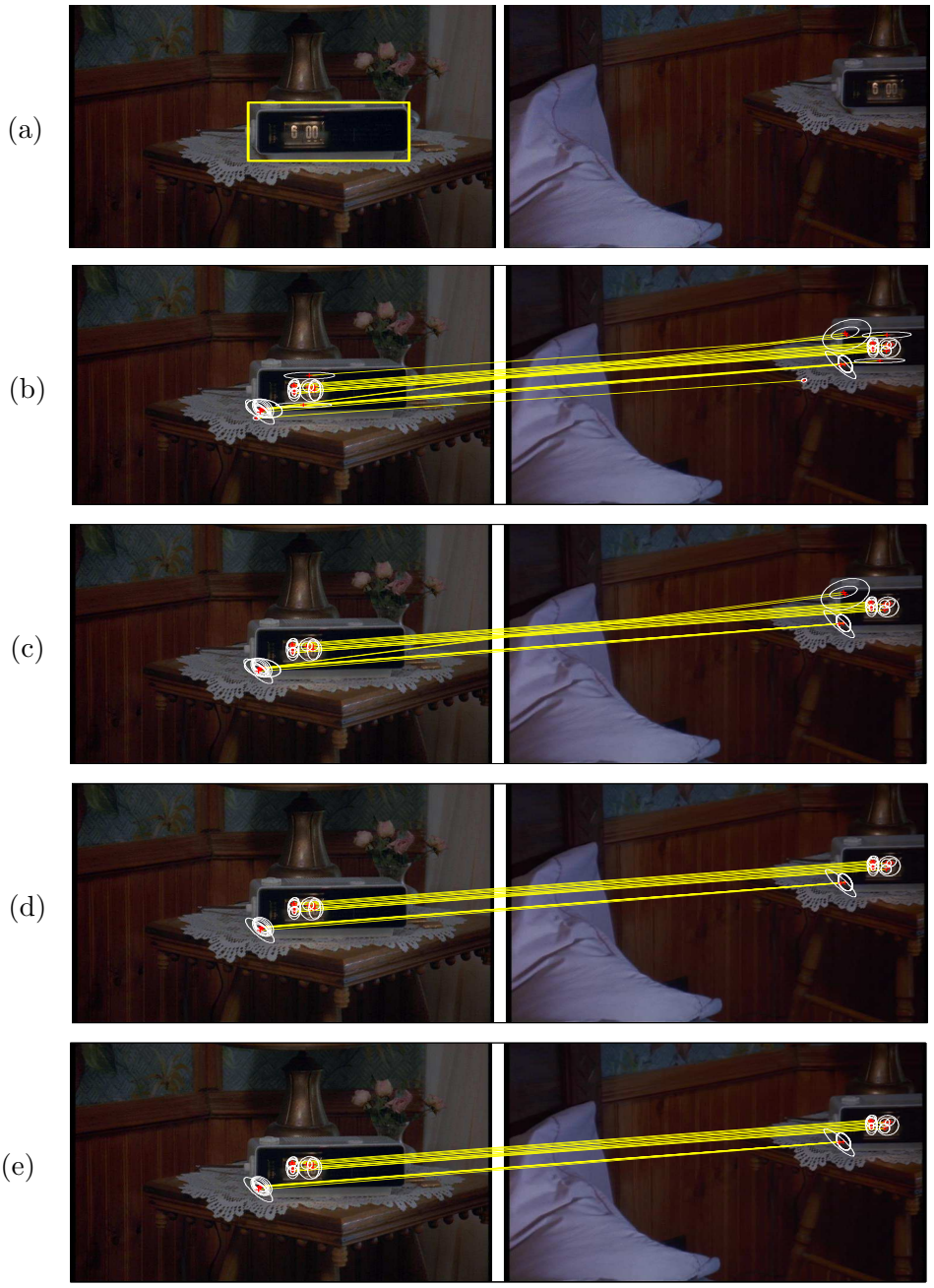


Figure 3.7: Stages of the spatial consistency check (compare with figure 3.6). (a) Left: The query frame from figure 3.6. Right: A test frame containing the same object. (b) Matches generated by visual words (after applying stop-list). (c)-(e) Stages of the spatial consistency check, as described in figure 3.6. The clock is correctly matched between the query and the test frame.

1. **Pre-processing (off-line)**

   - Detect affine covariant regions in each keyframe of the video. Represent each region by a SIFT descriptor (section 3.2).
   - Track the regions through the video and reject unstable regions (section 3.2).
   - Build a visual vocabulary by clustering stable regions from a subset of the video. Assign each region descriptor in each keyframe to the nearest cluster centre (section 3.3).
   - Remove stop-listed visual words (section 3.4.2).
   - Compute tf–idf weighted document frequency vectors (section 3.4.1).
   - Build the inverted file indexing structure (section 3.5).

2. **At run-time (given a user selected query region)**

   - Determine the set of visual words within the query region.
   - Retrieve keyframes based on visual word frequencies (section 3.4.1).
   - Re-rank the top $N_s(= 500)$ retrieved keyframes using spatial consistency (section 3.4.3).

Figure 3.8: The Video Google object retrieval algorithm. Example retrieval results are shown in figures 3.9 and 3.10.

3.5 Object retrieval using visual words

We first describe the off-line processing. A feature length film typically has 100K-150K frames. To reduce complexity, roughly one keyframe is used per second of video, which results in 4K-6K keyframes. Descriptors are computed for stable regions in each keyframe (stability is determined by tracking as described in section 3.2). The descriptors are vector quantized using the centres clustered from the training set, i.e. each descriptor is assigned to a visual word. The visual words over all frames are assembled into an inverted file structure where for each word, all occurrences and the position of the word in all frames are stored.

At run-time a user selects a query region, which specifies a set of visual words and their spatial layout. Retrieval then proceeds in two steps: Firstly, frames are retrieved based on their tf–idf weighted frequency vectors (the bag of words model), and are then re-ranked using spatial

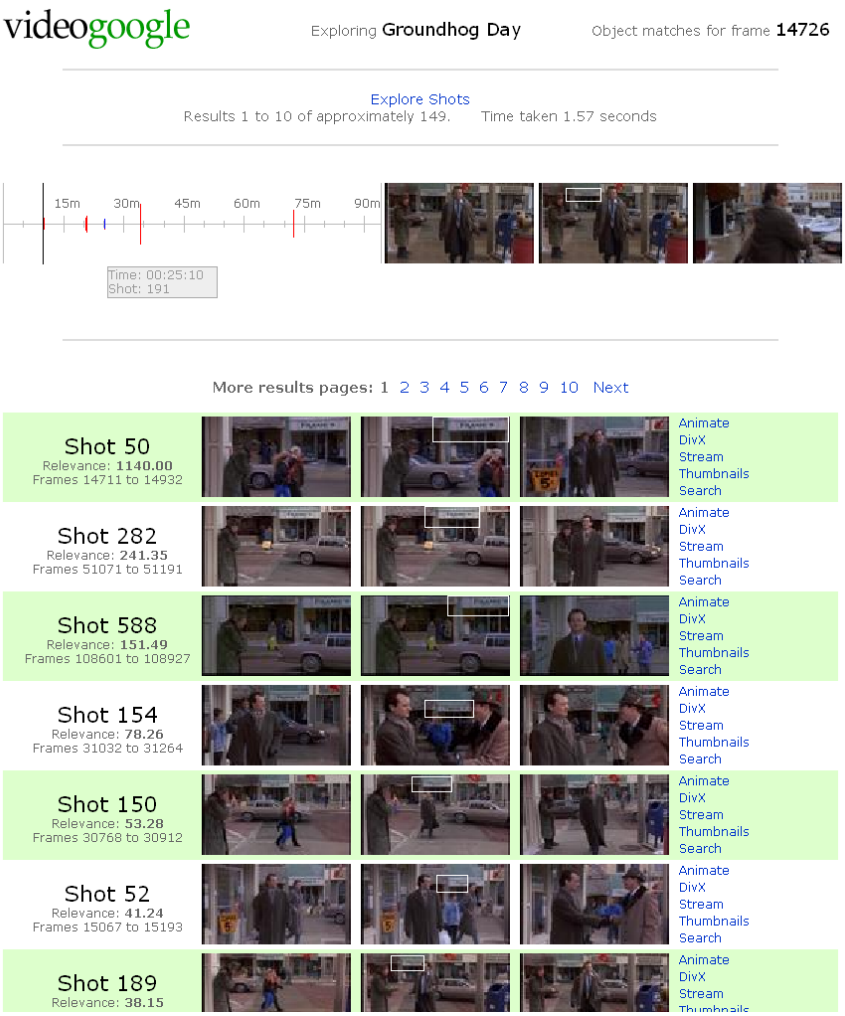


Figure 3.9: **Object query example II: Groundhog Day.** A screenshot of the running object retrieval system showing results of object query 3 from the query set of figure 3.11. The top part of the screenshot shows an interactive timeline, which allows the user to browse through the retrieved results in a chronological order. The bottom part of the screenshot shows the first seven ranked shots from the first page of retrieved shots. Each shot is displayed by three thumbnails showing (from left to right) the first frame, the matched keyframe with the identified region of interest shown in white, and the last frame of the shot. The precision-recall curve for this query is shown in figure 3.12.

consistency voting. The frequency based ranking is implemented using Matlab's sparse matrix engine and the spatial consistency re-ranking is implemented using the inverted file structure. The entire process is summarized in figure 3.8 and examples are shown in figures 3.9 and 3.10.

It is worth examining the time complexity of this retrieval architecture and comparing it to that of a method that does not vector quantize the descriptors. The huge advantage of the quantization

(a)



(b)



Figure 3.10: **Object query example III: Casablanca.** (a) Keyframe with user specified query region (lamp). (b) Screenshot showing the first eight ranked shots. Each shot is displayed by three thumbnails showing (from left to right) the first frame, the matched keyframe with the identified region of interest shown in white, and the last frame of the shot.

is that all descriptors assigned to the same visual word are considered matched. This means that the burden on the run-time matching is substantially reduced as descriptors have effectively been pre-matched off-line.

In detail, suppose there are N frames, a vocabulary of V visual words, and each frame contains R regions and M distinct visual words. $M < R$ if some regions are represented by the same visual word. Each frame is equivalent to a vector in $\mathbb{R}^V$ with M non-zero entries. Typical values are $N = 10,000$, $V = 20,000$ and $M = 500$. At run-time, the task is to compute the score (3.4) between the query frame vector $\mathbf{v}_q$ and each frame vector $\mathbf{v}_d$ in the database (another situation might be to only return the n closest frame vectors). The current implementation exploits sparse coding for efficient search as follows. The vectors are pre-normalized (so that the denominator of (3.4) is unity), and the computation reduces to one scalar product for each of the N frames. Moreover, only the $m \leq M$ entries which are non-zero in both $\mathbf{v}_q$ and $\mathbf{v}_d$ need to be examined during each scalar product computation (and typically there are far less than R regions in $\mathbf{v}_q$ as only a subpart of a frame specifies the object search). In the worst case if $m = M$ for all documents the time complexity is $O(MN)$.

If vector quantization is *not* used, then two alternative architectures are possible. In the first, the query frame is matched to each frame in turn. In the second, descriptors over all frames are combined into a single search space. As SIFT is used, the dimension, D, of the search space will be 128. In the first case the object search requires finding matches for *each* of the R descriptors of the query frame, and there are R regions in each frame, so there are R searches through R points of dimension D for N frames, a worst case cost of $O(NR^2D)$. In the second case, over all frames there are NR descriptors. Again, to search for the object requires finding matches for *each* of the R descriptors in the query image, i.e. R searches through NR points, again resulting in time complexity $O(NR^2D)$.

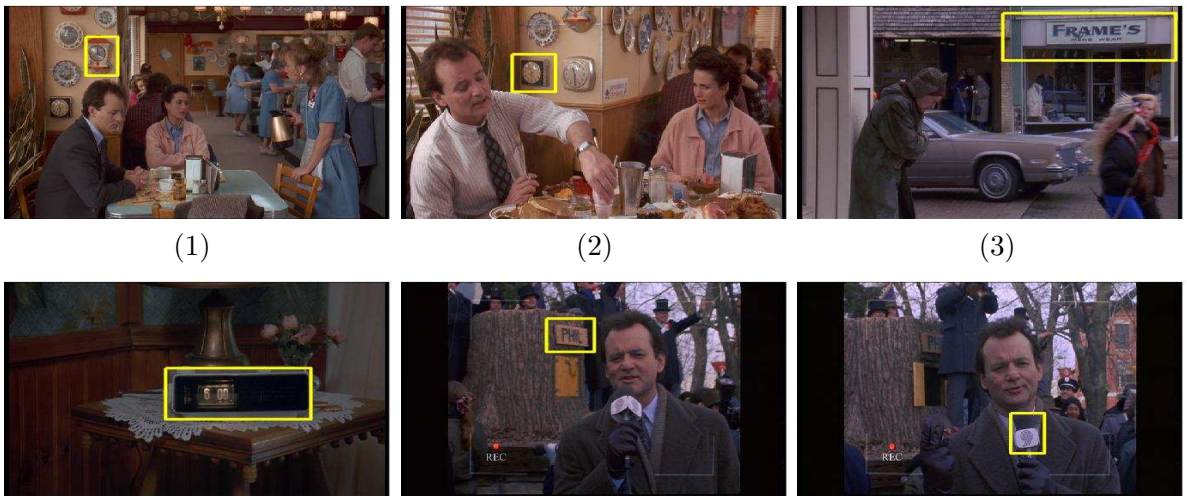
Consequently, even in the worst case, the vector quantizing architecture is a factor of RD times faster than not quantizing. These worst case complexity results can be improved by using efficient nearest neighbour or approximate nearest neighbour search [87].

3.6 Experiments

In this section, we evaluate the object retrieval performance over the entire movie on a ground truth test set of six object queries. In part this retrieval performance assesses the expressiveness of the visual vocabulary, since only about 12% of ground truth keyframes (and the invariant descriptors they contain) were included when clustering to form the vocabulary.

The performance is compared to a baseline method implementing standard frame to frame matching. The goal is to evaluate the potential loss of performance due to the descriptor quantization. In the baseline method, the same detected regions and descriptors (after the stability check) in each keyframe are used. The detected affine covariant regions within the query area in the query keyframe are sequentially matched to all 5,640 keyframes in the movie. For each keyframe, matches are obtained based on the descriptor values using nearest neighbour matching with a threshold ϵ on the distance. This results in a single or no match between each query descriptor and each keyframe. Euclidean distance is used here. Keyframes are ranked by the number of matches, and shots are ranked by their best scoring keyframes. Note that the baseline method is essentially equivalent to pooling all descriptors from all 5,640 keyframes into a single database and performing an 'ϵ-nearest neighbour search' for each query descriptor. In more detail, the ϵ-nearest neighbour search amounts to finding all points in the database within (Euclidean) distance ϵ of the query descriptor with an additional uniqueness constraint that only the best matching descriptor from each keyframe is retained. This is a type of descriptor matching method used by Schmid and Mohr [126] and later by Lowe [85].

The performance of the proposed method is evaluated on six object queries in the movie Groundhog Day. Figure 3.11 shows the query frames and corresponding query regions. Ground truth occurrences were manually labelled in all the 5,640 keyframes (752 shots). Retrieval is performed on keyframes as outlined in section 3.4 and each shot of the video is scored by its best scoring keyframe. Performance is measured using a precision-recall plot for each query. Precision is the number of retrieved ground truth shots relative to the total number of shots retrieved. Recall is the number of retrieved ground truth shots relative to the total number of ground truth shots in the movie. Precision-recall plots are shown in figure 3.12. The results are summarized using the Average Precision (AP) in figure 3.12. Average Precision is a scalar valued measure computed as



(1) (2) (3)

(4) (5) (6)

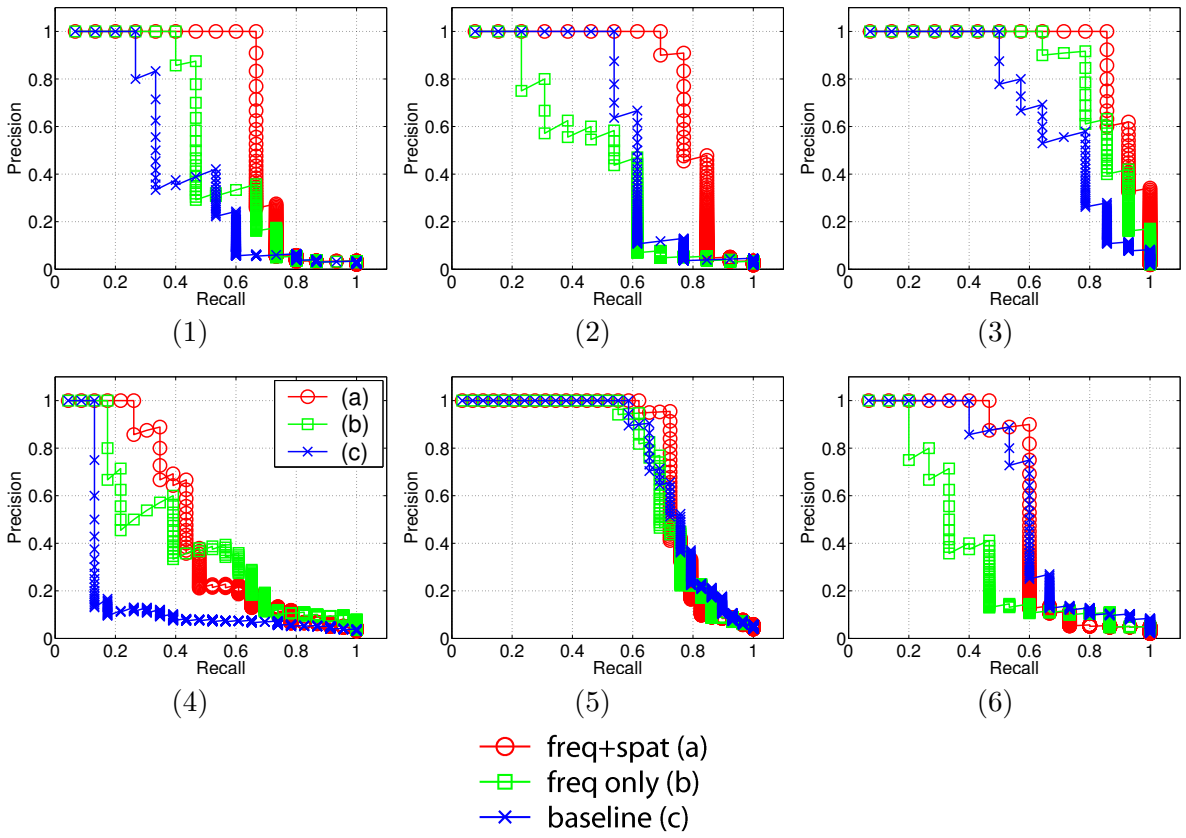
Object	# of keyframes	# of shots	# of query regions
1 Red Clock	138	15	31
2 Black Clock	120	13	29
3 Frames sign	92	14	123
4 Digital clock	208	23	97
5 Phil sign	153	29	26
6 Microphone	118	15	19

Figure 3.11: Query frames with outlined query regions for the six test queries with manually obtained ground truth occurrences in the movie Groundhog Day. The table shows the number of ground truth occurrences (keyframes and shots) and the number of affine covariant regions lying within the query rectangle for each query.

the area under the precision-recall graph and reflects performance over all recall levels. An ideal precision-recall curve has precision 1 over all recall levels, which corresponds to Average Precision of 1. Note that a precision-recall curve does not have to be monotonically decreasing. To illustrate this, say there are 3 correct shots out of the first 4 retrieved, which corresponds to precision $3/4 = 0.75$. Then, if the next retrieved shot is correct the precision increases to $4/5 = 0.8$.

It is evident that for all queries the average precision of the proposed method exceeds that of using frequency vectors alone – showing the benefits of using the spatial consistency to improve the ranking. On average (across all queries), the tf–idf frequency ranking method performs comparably to the baseline method. This demonstrates that using visual word matching does not result in a significant loss in performance against the standard frame to frame matching.

Further examining the precision-recall curves in figure 3.12 we note that the performance is biased towards high precision at lower recall levels. In practice, this might be acceptable for some



	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Average
AP freq+spat (a)	0.70	0.81	0.93	0.48	0.77	0.62	**0.72**
AP freq only (b)	0.55	0.49	0.86	0.43	0.73	0.41	**0.58**
AP baseline (c)	0.44	0.62	0.72	0.20	0.76	0.62	**0.56**

Average precision (AP) for the six object queries.

Figure 3.12: Precision-recall graphs (at the shot level) for the six ground truth queries on the movie Groundhog Day. Each graph shows three curves corresponding to (a) frequency ranking (tf–idf) followed by spatial consistency re-ranking (circles), (b) frequency ranking (tf–idf) only (squares), and (c) the baseline method implementing standard frame to frame matching (stars). Note the significantly improved precision at lower recalls after spatial consistency re-ranking (a) is applied to the frequency based ranking (b). The table shows average precision (AP) for each ground truth object query for the three different methods. The last column shows mean average precision over all six queries.

applications: for example a visual search of videos/images on the Internet, where the first few correctly retrieved videos/images (and their corresponding web-pages) might contain the relevant information. We note, however, that for some other applications, where finding all instances of an object is important (e.g. surveillance), higher precision at higher recall levels might be preferable.

Figures 3.1, 3.13 and 3.14 show example retrieval results for three object queries for the movie 'Groundhog Day', and figure 3.15 shows example retrieval results for the black and white film 'Casablanca'. For the 'Casablanca' retrievals, the film is represented by 5,749 keyframes, and a new visual vocabulary was built as described in section 3.3.

Processing requirements: The region detection, description and visual word assignment takes about 20 seconds per frame (720×576 pixels) in this implementation but this is done off-line. Optimized implementations currently run at 5Hz [99]. The average query time for the six ground truth queries on the database of 5,640 keyframes is 0.82 seconds with a Matlab implementation on a 2GHz Pentium. This includes the frequency ranking and spatial consistency re-ranking. The spatial consistency re-ranking is applied only to the top $N_s = 500$ keyframes ranked by the frequency based score. This restriction results in no loss of performance (measured on the set of ground truth queries).

The query time of the baseline matching method on the same database of 5,640 keyframes is about 500 seconds. This timing includes only the nearest neighbour matching performed using linear search. The region detection and description is also done off-line. Note that on this set of queries our proposed method has achieved about 600-fold speed-up compared to the baseline linear search.

In terms of memory requirements, the inverted file for the movie 'Groundhog Day' takes about 66MB and stores about 2 million visual word occurrences (this is with the 10% most frequent words removed). For each visual word occurrence we store: (i) the frame number, (ii) the x and y position in the frame and (iii) the distance to the 15th nearest neighbour in the image. For comparison, storing 128-dimensional descriptors in double precision (8 bytes) for two million regions would take about 2GB.



Figure 3.13: **Object query example IV: Groundhog Day.** (a) Keyframe with user specified query region in yellow (Phil sign), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 1st, 4th, 10th, and 19th retrieved shots with the identified region of interest shown in yellow, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 21st. The precision-recall graph for this query is shown in figure 3.12 (object 5). Querying 5,640 keyframes took 0.64 seconds on a 2GHz Pentium.



Figure 3.14: **Object query example V: Groundhog Day.** (a) Keyframe with user specified query region in yellow (tie), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 1st, 2nd, 4th, and 19th retrieved shots with the identified region of interest shown in yellow, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 25th. Querying 5,640 keyframes took 0.38 seconds on a 2GHz Pentium.

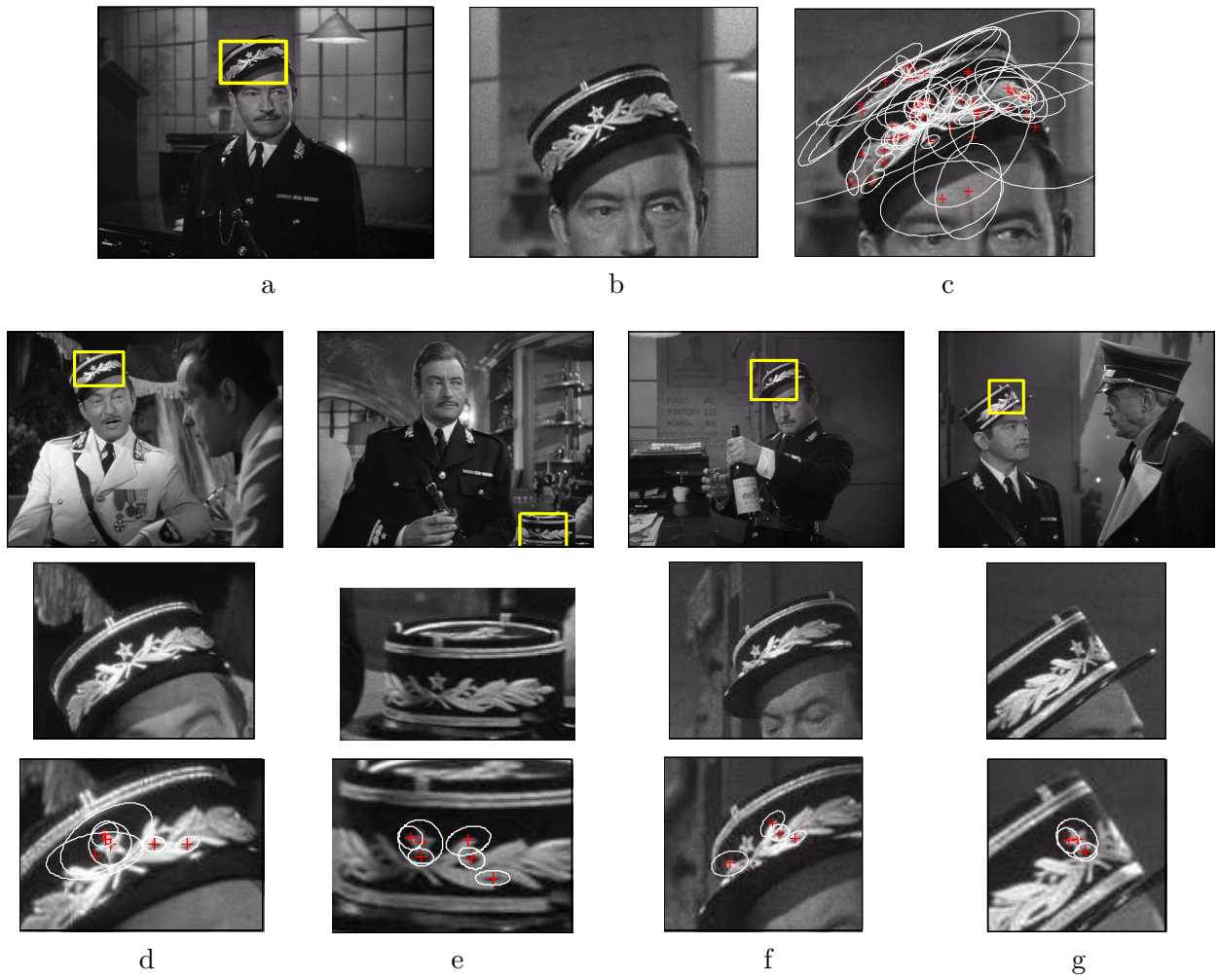


Figure 3.15: **Object query example VI: Casablanca.** (a) Keyframe with user specified query region in yellow (hat), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 4th, 5th, 11th, and 19th retrieved shots with the identified region of interest shown in yellow, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 25th. Querying 5,749 keyframes took 0.83 seconds on a 2GHz Pentium.

Missed detections for the current method

Examples of frames from low ranked shots are shown in figure 3.16. Appearance changes due to extreme viewing angles, large scale changes and significant motion blur affect the process of extracting and matching affine covariant regions. The examples shown represent a significant challenge to the current object matching method.

Searching for objects from outside the movie

Figure 3.17 shows an example of a search for an object outside the 'closed world' of the film. The object (a Sony logo) is specified by a query image downloaded from the Internet. The image was preprocessed as outlined in section 3.2. Figure 3.18 shows three more examples of external searches on the feature length movies 'Run Lola Run' and 'Pretty Woman'. Searching for images from other sources opens up the possibility for product placement queries, or searching movies for company logos, or particular buildings or types of vehicles.

In the following two subsections, we first study the object retrieval performance with respect to different visual vocabularies (subsection 3.6.1), and then we investigate in depth various frequency ranking and weighting methods (subsection 3.6.2).

3.6.1 Vocabulary investigation

In the following experiments, we vary the parameters of the object retrieval system such as the number of words in the visual vocabulary, the size of the stop-list and the size of the retrieval database. Finally, we examine the quality of individual visual words.

Varying the number of words of the visual vocabulary

The goal here is to evaluate the performance of the proposed object retrieval system for different cardinalities of the visual vocabulary. The visual vocabulary is built as described in section 3.3, and retrieval is performed as outlined in section 3.4, using both the frequency ranking and spatial consistency re-ranking steps. The top 10% most frequent visual words were stopped. The proportion of SA to MS regions is kept constant (=3/5) throughout the experiments. The results are summarized in figure 3.19. The best performance is obtained for a visual vocabulary size of 16,000.



(1,2) (4)

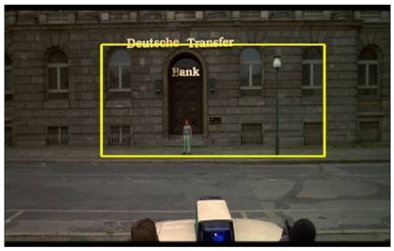
Figure 3.16: Examples of missed (low ranked) detections for objects 1, 2 and 4 from figure 3.11. In the left image the two clocks (objects 1 and 2) are imaged from an extreme viewing angle and are barely visible – the red clock (object 2) is partially out of view. In the right image the digital clock (object 4) is imaged at a small scale and significantly motion blurred. Examples shown here were also low ranked by the baseline method.



Figure 3.17: **Searching for a Sony logo.** First column: (top) Sony Discman image (640 × 422 pixels) with the query region outlined in yellow and (bottom) close-up with detected elliptical regions superimposed. Second and third column: (top) retrieved frames from two different shots in 'Groundhog Day' with detected Sony logo outlined in yellow and (bottom) close-up of the image. The retrieved shots were ranked 1 and 4.



(a) Query



(b)



(c)





(d) Query





(e)





(f)

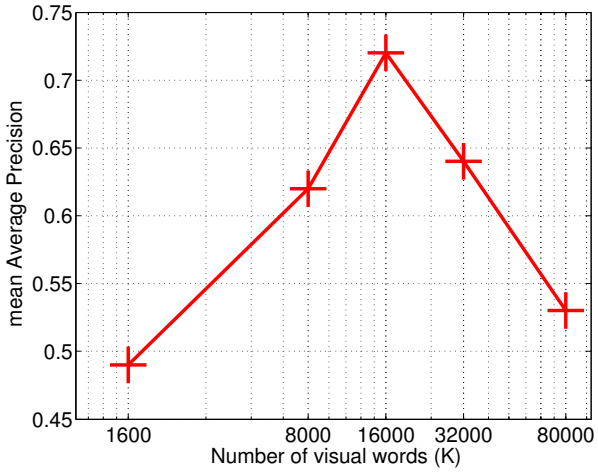


(g) Query



(h)

Figure 3.18: **Searching for objects and locations from outside the movie.** (a) A query frame. (b),(c) Frames from two shots, ranked 3 and 7, correctly retrieved from the movie 'Run Lola Run'. All three images show the same building – a museum in Berlin, which was redesigned to look like a bank in the movie. (d) Top: A query frame. Bottom: close-up of the query region. (e),(f) Top: Frames from two shots, ranked 1 and 2, retrieved from the movie 'Run Lola Run'. Bottom: Close-ups of detected regions of interest. (g) Query frame. (h) A frame from the first shot retrieved from the movie 'Pretty Woman'. All query images, (a),(d) and (g), are downloaded from the Internet.



# of visual words (K)	mean Average Precision
1,600	0.49
8,000	0.62
16,000	**0.72**
32,000	0.64
80,000	0.53

Figure 3.19: The graph and table show mean Average Precision of the proposed object retrieval method with changing number of visual words. The mean Average Precision is computed over the six ground truth object queries from figure 3.11.

The size of the visual vocabulary is clearly an important parameter which affects the retrieval performance. Intuitively, when the number of clusters is too small, the resulting visual words are non-discriminative generating many false positive matches. On the other hand, when the number of clusters is too large, descriptors from the same object/scene region in different images can be assigned (due to e.g. noise) to different clusters generating false negative (missed) matches.

Recently, Nister and Stewenius [99] proposed a visual vocabulary organized in a tree together with a hierarchical scoring scheme, which seems to overcome the difficulty of choosing a particular number of cluster centers.

Effect of the stop list

Table 3.1 evaluates the effect of varying the size of the stop list on the performance of the proposed object retrieval system (after the spatial consistency re-ranking). The best performance (mean Average Precision 0.72) is obtained when 10% of the most frequent visual words are stopped. This amounts to stopping 1,600 most frequent visual words out of the vocabulary of 16,000. Note that

Size of stop list (%)	mean Average Precision
0	0.66
1	0.66
5	0.71
10	**0.72**
20	0.65

Table 3.1: **Effect of the stop list.** Mean Average Precision of the proposed object retrieval method with the varying size of stop list. The mean Average Precision is computed over the six ground truth object queries from figure 3.11. The vocabulary size is 16,000 visual words.

Visual Vocab.	kfrms	K	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Average
(a) Ghd.	474	16,000	0.70	0.78	0.94	0.46	0.78	0.62	**0.71**
(b) Casa.	483	16,000	0.25	0.31	0.47	0.20	0.48	0.15	**0.31**
(c) Casa.+Ghd.	474, 483	32,000	0.58	0.92	0.93	0.51	0.69	0.53	**0.69**

Table 3.2: **Generalization performance.** Performance of the object retrieval method with respect to different visual vocabularies on 5,640 keyframes of 'Groundhog Day'. (a) Visual vocabulary of 16,000 visual words built from 474 keyframes of 'Groundhog Day'. (b) Visual vocabulary of 16,000 visual words built from 483 keyframes of 'Casablanca'. (c) Visual vocabulary of 32,000 visual words obtained by concatenating visual vocabularies (a) and (b). Performance is measured by average precision on the six ground truth queries from 'Groundhog Day' shown in figure 3.11.

stopping the 1,600 most frequent visual words removes about 1.25 million visual word occurrences (out of the total of about 3.2 million) appearing in the 5,640 keyframes of the movie 'Groundhog Day'.

Evaluating generalization performance of visual vocabulary

To test the generalization performance of the visual vocabulary we evaluate the object retrieval performance on the 5,640 keyframes of 'Groundhog Day' with different visual vocabularies. The results are shown in table 3.2. Visual vocabularies (a) from 'Groundhog Day', and (b) from 'Casablanca', were built as outlined in section 3.3, i.e. vector quantization was performed only within frames of one movie. Visual vocabulary (c) was obtained by concatenating visual vocabularies (a) and (b). Using the visual vocabulary built from 'Casablanca' (b) for retrieval in 'Groundhog Day' results in a performance drop in comparison to the performance of the 'Groundhog Day' vocabulary (a). On the other hand, case (c), simple concatenation of vocabularies (a) and (b), brings the performance almost to the original level (a). Note that in all three cases, (a)-(c), the top 5% most frequent visual words are stopped. Using the 10% stop-list lowers the performance (measured by the mean average precision) of vocabulary (b) and (c). This might be attributed to higher importance of

Visual Vocab.	kfrms	K	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Average
(a) Ghd.	474+0	16,000	0.62	0.61	0.92	0.35	0.71	0.54	0.63
(b) Ghd.+Casa.	474+483	16,000	0.49	0.56	0.85	0.37	0.68	0.48	0.56
(c) Ghd.+Casa.	474+483	24,000	0.63	0.50	0.76	0.39	0.73	0.51	0.59
(d) Ghd.+Casa.	474+483	32,000	0.61	0.65	0.84	0.52	0.80	0.54	**0.66**
(e) Ghd.+Casa.	474+483	40,000	0.68	0.59	0.78	0.51	0.79	0.49	0.64

Table 3.3: **Increasing the database size.** Performance of the object retrieval method on a database of 11,389 keyframes from two movies ('Groundhog Day' and 'Casablanca') with respect to different visual vocabularies. See text. Performance is measured by average precision on the six ground truth queries from 'Groundhog Day' shown in figure 3.11.

more general (and more common) visual words in this case.

Increasing the database size

Here we test the retrieval performance on a larger database, composed of 11,389 keyframes from the two movies 'Groundhog Day' (5,640 keyframes) and 'Casablanca' (5,749 keyframes). The same ground truth set of queries from the movie 'Groundhog Day' (figure 3.11) is used here but the additional keyframes from 'Casablanca' act as distractors potentially lowering the precision/recall of the retrieved results. The test was performed with five different visual vocabularies: (a) the original vocabulary of 16,000 visual words computed from 'Groundhog Day'; (b)–(e) vocabularies clustered from 474 'Groundhog Day' keyframes and 483 'Casablanca' keyframes into 16,000 – 40,000 visual words. Results are summarized in table 3.3. In all cases the top 10% most frequent visual words were stopped. Examining results for the vocabulary (a), we observe that increasing the database size by adding the extra distractor keyframes from 'Casablanca' lowers the mean Average Precision from 0.72 to 0.63 (cf figure 3.12, method (a)). The best performance on the extended database (mean Average Precision 0.66) is achieved for vocabulary (d), where descriptors from 'Groundhog Day' and 'Casablanca' are pooled together and jointly clustered into 32,000 visual words. This suggests that including descriptors from 'Casablanca' in the vocabulary building step is beneficial and reduces confusion between 'Groundhog Day' and 'Casablanca' objects. Vocabularies (b), (c) and (e) are clustered from the same set of keyframes as (d) but into a different number of visual words. Note that the number of visual words is an important parameter, which significantly influences the final performance. Similar 'quantization' effects were observed on the database composed of only 'Groundhog Day' keyframes (figure 3.19) but with the best performance at 16,000 visual words.

Quality of individual visual words: It is also interesting to test the 'quality' of individual query visual words. The goal here is to examine retrieval performance if only a single visual word is used as a query. Visual words with good retrieval performance (i) should occur mostly on the object of interest (high precision), and (ii) should retrieve all the object occurrences in the database (high recall). In particular, for an individual visual word, the retrieved keyframes are all keyframes where the visual word occurs. Note that there is no ranking among the retrieved keyframes as all occurrences of a single visual word are treated with an equal weight. As a result, a single visual word produces a single point on the precision-recall curve. Precision is the number of retrieved ground truth keyframes relative to the total number of keyframes retrieved. Recall is the number of retrieved ground truth keyframes relative to the total number of ground truth keyframes in the movie. Precision/recall graphs, shown in figure 3.20, indicate that individual visual words are 'noisy', i.e. occur on multiple objects or do not cover all occurrences of the object in the database. It should be noted here, that the requirement that each visual word occurs on only one object (high precision) might be unrealistic in a real world situation as it implies that the vocabulary would grow linearly with the number of objects. A more realistic situation is that visual words are shared across objects and that an object is represented by a conjuction of several visual words.

3.6.2 Comparison of term frequency weighting and ranking methods

In this section, we describe alternative term frequency weighting and ranking schemes and compare their performance with the standard tf–idf weighting (described in section 3.4.1). Performance is evaluated on the ground truth set of queries of figure 3.11. Spatial consistency is not applied.

Freq-L_2

In this method document vectors are formed using only absolute term frequencies,

$$t_i = n_{id}, \tag{3.6}$$

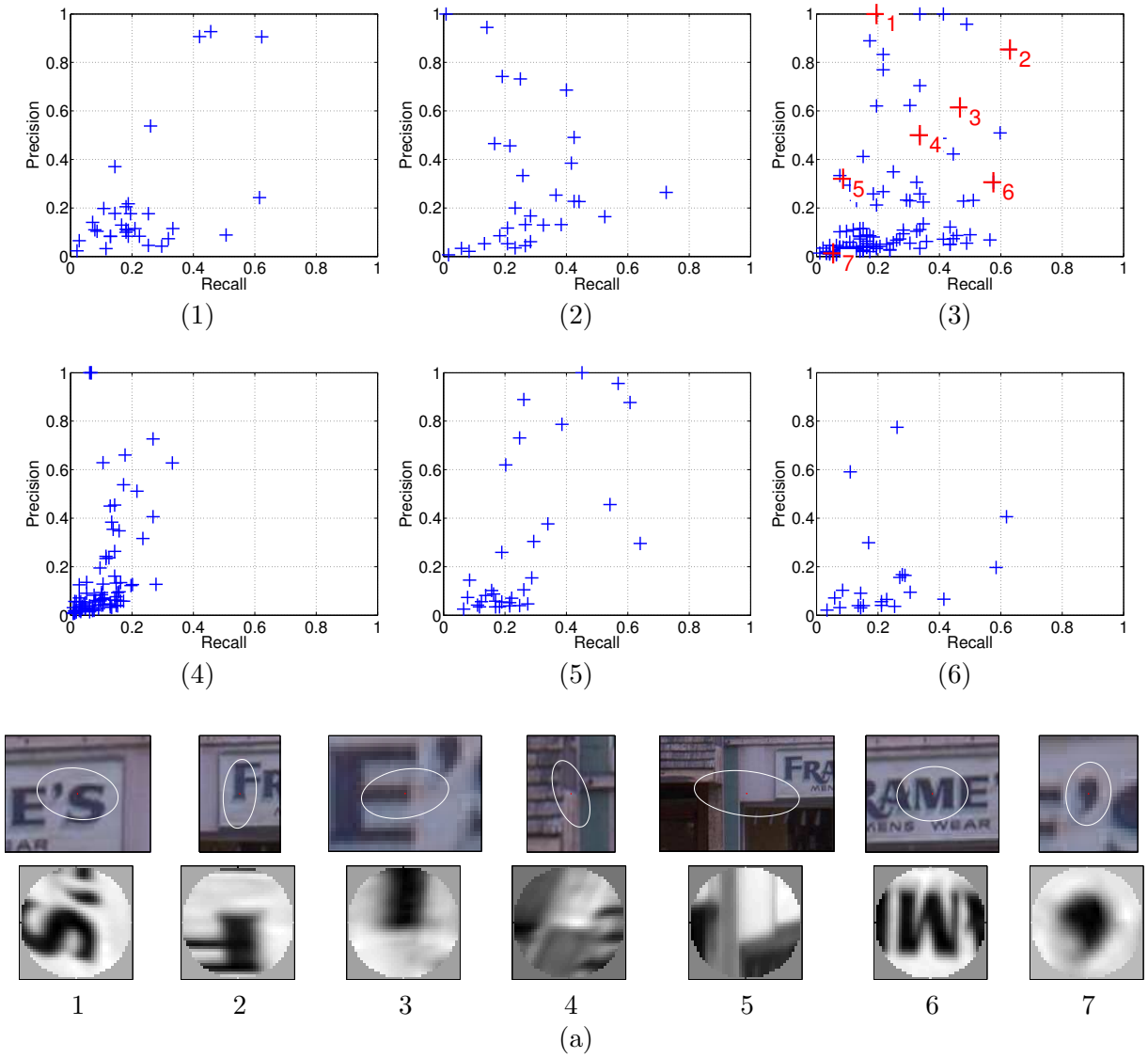


Figure 3.20: **'Quality' of single visual words.** Precision-recall graphs show the 'quality' of each individual query visual word for the six ground truth objects of figure 3.11. Each precision-recall point in a graph represents the quality/performance of a single query visual word. See text. Note that many visual words are quite weak individually with low recall and precision. Some visual words are more indicative for the presence of the object, but none of them achieves perfect results, which would be the top-right corner of the graph. (a) Examples of visual words describing object (3) – 'Frames sign'. Top row: scale normalized close-ups of elliptical regions overlaid over the query image. Bottom row: corresponding normalized regions. Visual words are numbered and their precision and recall values are shown in the precision-recall graph (3).

and query and document vectors are normalized to have unit L_2 norm. Note that starting from relative term frequencies,

$$t_i = \frac{n_{id}}{n_d}, \tag{3.7}$$

gives the same document vector as starting from absolute term frequencies (3.6), as the L_2 normalization cancels the n_d term in the denominator of (3.7). Similarity is computed using the normalized scalar product (3.4). The reason for including this method is to compare the term frequency weighting with the tf–idf weighting and assess the contribution of the inverse document frequency term.

Freq-L_1

In this method document vectors are formed using term frequencies (3.6) but are normalized to have unit L_1 norm (instead of L_2), $\|\mathbf{v}_q\|_1 = 1$, $\|\mathbf{v}_d\|_1 = 1$, where $\|\mathbf{v}\|_1 = \sum_{i=1}^{V} |t_i|$. Using L_1 normalization is equivalent to using relative term frequencies (3.7). The similarity score is computed using L_1 distance as

$$1 - \frac{1}{2}\|\mathbf{v}_q - \mathbf{v}_d\|_1. \tag{3.8}$$

The goal here is to compare the L_1 and L_2 based normalization and similarity score.

Freq-χ^2

Here document vectors are treated as normalized histograms (probability distributions) over terms [76, 83, 102, 166, 167], i.e. relative word frequencies (3.7) are used (vectors are normalized to sum to one). Similarity between two vectors (normalized histograms) is computed using the χ^2 distance [83, 108, 166] as

$$1 - \frac{1}{2}\chi^2(\mathbf{v}_q, \mathbf{v}_d), \tag{3.9}$$

where

$$\chi^2(\mathbf{v}_q, \mathbf{v}_d) = \sum_{i=1}^{V} \frac{(t_{qi} - t_{di})^2}{(t_{qi} + t_{di})}. \tag{3.10}$$

Freq-KL

As in the 'Freq-χ^2' method above, document vectors are treated as probability distributions over terms, but the dissimilarity score between the query vector and document vectors is computed using the *Kullback–Leibler (KL) divergence* [76, 102, 167]

$$D_{KL}(\mathbf{v}_q \| \mathbf{v}_d) = \sum_{i=1}^{V} t_{qi} \log \frac{t_{qi}}{t_{di}}. \tag{3.11}$$

Note that the Kullback–Leibler divergence is not symmetric, $D_{KL}(\mathbf{v}_q \| \mathbf{v}_d) \neq D_{KL}(\mathbf{v}_d \| \mathbf{v}_q)$. In particular, note that document terms which are not present in the query have limited effect on the $D_{KL}(\mathbf{v}_q \| \mathbf{v}_d)$ as the corresponding t_{qi} are zero. This is an important difference from the χ^2 distance based ranking (3.9) as the χ^2 distance is symmetric and penalizes terms which are present in the document vector $\mathbf{v}_d$ and missing in the query vector $\mathbf{v}_q$.

tf-idf-KL

In this method document vectors are formed using the tf-idf weighted visual word frequencies (3.3). Document vectors are then normalized to sum to one and the dissimilarity score between the query vector and document vectors is computed using the KL divergence (3.11). The goal is to compare performance of this method with the 'Freq-KL' method above and evaluate the contribution of the idf weights.

Freq-Bhattacharyya

As above, document vectors are treated as probability distributions over terms, i.e. visual word frequencies (3.6) are used and query and document vectors are normalized to have unit L_1 norm, $\|\mathbf{v}_q\|_1 = 1$, $\|\mathbf{v}_d\|_1 = 1$. The similarity score between the query vector and document vectors is measured using the Bhattacharyya coefficient [5, 26],

$$B(\mathbf{v}_q, \mathbf{v}_d) = \sum_{i=1}^{V} \sqrt{t_{qi}\, t_{di}}. \tag{3.12}$$

The Bhattacharyya coefficient can be geometrically interpreted [5, 26] as a cosine of the angle between vectors $\mathbf{u}_q = (\sqrt{t_{q1}}, \ldots, \sqrt{t_{qV}})^\top$ and $\mathbf{u}_d = (\sqrt{t_{d1}}, \ldots, \sqrt{t_{dV}})^\top$. Note that both $\mathbf{u}_q$ and $\mathbf{u}_d$ have unit L_2 norm since $\mathbf{v}_q$ and $\mathbf{v}_d$ have unit L_1 norm.

tf-idf-Bhattacharyya

Here document vectors are formed using the tf-idf weighted visual word frequencies (3.3). Document vectors are then normalized to sum to one and the similarity score between the query vector and document vectors is computed using the Bhattacharyya coefficient (3.12). The goal is to compare performance of this method with the 'Freq-Bhattacharyya' method above and evaluate the contribution of the idf weights.

Binary

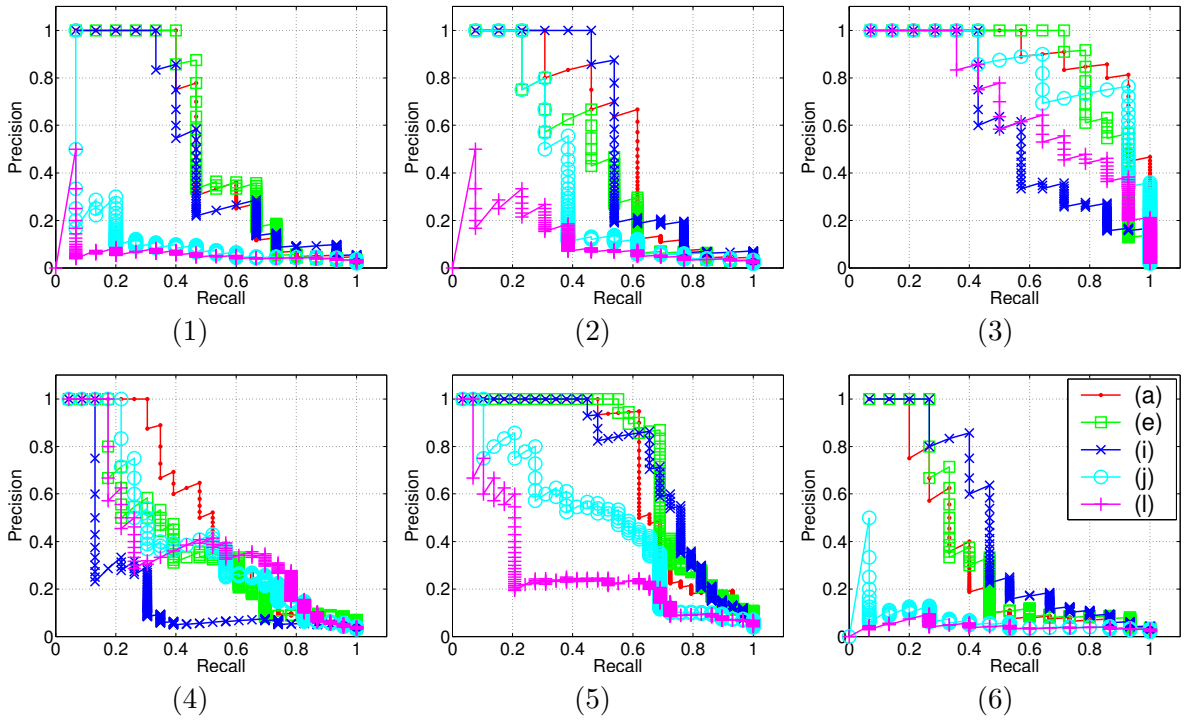
Here document vectors are binary, i.e. $t_i = 1$ if the word i is present in the document and zero otherwise. Similarity is measured using the (unnormalized) scalar product ${\mathbf{v}_q}^\top \mathbf{v}_d$. This similarity score simply counts the number of distinct terms in common between the query and the retrieved document. Note that this method can be also viewed as an interesection of binary (un-normalized) histograms, $\mathbf{v}_q$ and $\mathbf{v}_d$.

In addition to the binary vector method described above we introduce four other binary vector based methods: *Binary-*L_2, *Binary-*L_1, *Binary-*χ^2 and *Binary-*KL. These methods are analogous to methods described above, i.e. the same normalization and similarity score is used. The only difference is that the initial document vectors (before normalization) are binary rather than based on term frequencies (3.6). The reason for including the 'binary' methods is to assess the importance of using term frequencies. Note that the *Binary-Bhattacharyya* method is not included as it produces the same document ranking as the *Binary-*L_2 method.

Performance comparison

Precision-recall plots for the different term frequency ranking methods are shown in figure 3.21. Results are summarized using Average Precision (AP) in the table in figure 3.21.

The best average performance over all queries (mean AP 0.61) is achieved by the 'tf-idf-



	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Average
(a) tf-idf-Bhatta	0.54	0.58	0.91	0.53	0.71	0.37	**0.61**
(b) tf-idf-KL	0.67	0.59	0.72	0.24	0.78	0.57	**0.60**
(c) frq-KL	0.66	0.61	0.67	0.24	0.77	0.56	**0.59**
(d) bin-KL	0.66	0.61	0.67	0.24	0.76	0.57	**0.59**
(e) frq-tf-idf	0.55	0.49	0.86	0.43	0.73	0.41	**0.58**
(f) frq-Bhatta	0.50	0.57	0.86	0.44	0.68	0.33	**0.56**
(g) frq-L_2	0.57	0.49	0.77	0.39	0.70	0.31	**0.54**
(h) bin-L_2	0.45	0.60	0.87	0.38	0.63	0.30	**0.54**
(i) bin	0.51	0.59	0.63	0.23	0.74	0.49	**0.53**
(j) frq-χ^2	0.16	0.38	0.86	0.44	0.49	0.09	**0.40**
(k) bin-χ^2	0.17	0.27	0.82	0.43	0.47	0.09	**0.37**
(l) frq-L_1	0.09	0.16	0.71	0.43	0.30	0.05	**0.29**
(m) bin-L_1	0.11	0.10	0.61	0.39	0.29	0.05	**0.26**

Average precision (AP) for the six object queries.

Figure 3.21: **Comparison of frequency ranking methods.** Precision-recall graphs (at the shot level) for the six ground truth queries on the movie Groundhog Day comparing performance of different term frequency ranking methods. The table shows average precision (AP) for each ground truth object query. The last column shows mean average precision over all six queries. Note that precision–recall graphs are shown only for methods (a), (e), (i), (j) and (l) from the table, so that the curves are visible.

Bhattacharyya' frequency ranking method (a), which combines the 'tf-idf' term weighting with the Bhattacharyya ranking score. Relatively high performance (mean AP 0.58–0.60) is also achieved by Kullback-Leibler divergence methods (b,c,d) and the standard 'tf-idf' method (e), described in section 3.4.1. Considerably worse results (mean AP 0.26–0.40) are obtained using χ^2 (j,k) and L_1 (l,m) distance based methods.

The L_1 (l,m) and χ^2 (j,k) methods perform poorly on queries 1–2 and 5–6. By close inspection of the results we found that this is due to highly ranked false positive images with small total number (10–50) of visual words and only 1–2 visual words common with the query.

Note also the superior performance (measured by the mean Average Precision) of the KL divergence method (c) to the χ^2 method (j). This can be attributed to the asymmetry of the KL divergence as discussed above.

By comparing each frequency method with its corresponding binary method we also note that using term frequencies seems to produce slightly better ((j,k) and (l,m)) or equal ((g,h) and (c,d)) results, measured by the mean Average Precision. The superior performance (measured by the mean AP) of the tf-idf methods (a,b,e) compared with their frequency based counterparts (f,c,g) may be attributed to the positive contribution of the inverse document frequency weighting.

In all the above experiments the top 5% most frequent visual words were stopped. If the 10% stop-list is used, the performance of method (b) goes down slightly to mean average precision 0.59. The performance of methods (a,e) remains the same. Note that methods (a,b,e) use the tf-idf weighting. More interestingly, performance of the other methods (c,d,f–m), which do not use the tf-idf weighting, slightly increases (by on average 0.035). For example, the mean average precision of methods (f) and (g) increases from 0.56 and 0.54 to 0.59 and 0.57, respectively, which makes them comparable to their tf-idf counterparts (a) and (e). This suggests that applying a stop-list has a similar effect to using tf-idf weights. In other words, the inverse document frequency (idf) weighting component might be viewed as a 'soft' stop-list, down-weighting very common visual words. Applying the stop-list, however, has the additional benefit of discarding mis-matches (as was illustrated in figure 3.5), which helps in the spatial consistency re-ranking stage (cf table 3.1), and is also useful for localizing objects in images.

3.7 Conclusions and discussion

We have demonstrated a scalable object retrieval architecture, which utilizes a visual vocabulary based on vector-quantized viewpoint invariant descriptors. The vector quantization does not appear to introduce a significant loss in retrieval performance (precision or recall) compared to nearest neighbour matching.

The spatial consistency re-ranking was shown to be very effective in improving the precision and removing false positive matches. However, the precision could be further improved by a more thorough (and more expensive) verification, based on a stricter measure of spatial similarity (e.g. angular ordering of regions [125], region overlap [49] or common affine geometric transformation [87]).

Currently the search is biased towards lightly textured regions which are detectable by the applied region detectors (corner like features, blob like regions). However, the framework allows other co-variant regions to be added (they will define an extended visual vocabulary), for example those of [156]. We also plan to include shape and contour based descriptors [14, 97], which would allow us to match textureless objects (bottles or mugs) or wiry objects (bicycles or chairs) [22]. The method proposed in this chapter allows retrieval for a particular visual aspect of an object. However, temporal information within a shot may be used to group visual aspects, and enable object level retrieval. This is addressed in the next chapter.

A live demonstration of the 'Video Google' system on a publicly available movie (Dressed to Kill) is available on-line at [1].

Chapter 4

Object Level Grouping for Video Shots

4.1 Introduction

In the video retrieval application described in the previous chapter, a query is specified by an image of the object of interest. Such queries enable retrieval of objects with a limited degree of generalization over viewpoint and deformation – but specifying the front of a car as a query will not retrieve shots of the rear of the car. However, shots in a video do contain examples of objects undergoing viewpoint changes and deformations. In this chapter, the objective is to use such multiple instances of an object in a shot in order to enable object-level retrieval, including: (i) deformable objects, e.g. a face changing expression; and (ii) multiple visual aspects of a 3D object, e.g. a vehicle seen from the front, side, and back. Figure 4.1 shows example shots of a deforming object, and of multiple visual aspects of a 3D object.

The approach we take is to automatically associate regions of frames of the shot into *object-level groupings*. Such groupings, containing multiple exemplars of an object, then implicitly represent different appearances of a deforming object or different visual aspects of a 3D object present in the video shot. Instead of specifying the query object in a single image, the retrieval is performed at an object-level, where the query object is defined over multiple images.

The goal of the object-level grouping is to associate all occurrences of an object in the video



(a)



(b)

Figure 4.1: Two example shots from the film 'Groundhog Day' [Ramis, 1993]. (a) Frames from a shot of an actor turning her head and speaking. Tracks of affine covariant regions are used to associate multiple exemplars of the face (from different viewpoints and with different expressions) for retrieval. (b) Frames from a shot where the camera pans to follow a van passing by. Tracks are used to associate the three visual aspects (back, side and front) of the van which are never visible simultaneously in a single frame.

shot. This is carried out using both motion and appearance consistency throughout the shot. As in chapter 3, the technology we employ is that of affine covariant regions [91, 93, 121, 156], reviewed in section 2.1.4. These regions deform with viewpoint so that their pre-image corresponds to the same surface patch.

Three cues are used to group affine covariant regions on a particular object throughout the shot. The first cue is the *temporal continuity* of the video, and is exploited by frame-to-frame tracking of affine covariant regions. This allows us, for example, to associate exemplars of a deforming object (e.g. a face talking and blinking). The second cue is based on *common motion*, and allows us to associate, for example, the front and back of a moving van, despite the fact that they never appear together in a single frame. Here, the grouping is based on the fact that the front and back of the van move together as one rigid object. Finally, the third cue is based on *common appearance* and allows us to associate occurrences of the same object despite the fact that the object is occluded for part of the shot. This is implemented by matching affine covariant regions on the object.

To achieve object-level grouping, we have developed the state of the art in two areas: first, the affine covariant regions are used to repair short gaps in tracks (section 4.2.2), and also to associate a set of tracks when the object is partially or totally occluded for a period (section 4.5). The result is

that regions are matched throughout the shot whenever they appear. Second, we develop a method of independent motion segmentation using robust affine factorization (section 4.4), which is able to handle degenerate motions [152] in addition to the usual problems of missing and mis-matched points [2, 32, 68, 132]. The object level grouping task we carry out differs from that of layer extraction [151], or dominant motion detection, where generally 2D planes are extracted, though we build on these approaches. Here the object may be 3D, and we pay attention to this, and also it may not always be the foreground layer, as it can be partially or totally occluded for part of the sequence.

Chapter outline: In section 4.2 we describe the algorithm for tracking affine covariant regions. Then section 4.3 shows how region tracks alone are used to associate exemplars of a deforming object. A retrieval example is also given. The algorithm for associating exemplars for different aspects of a 3D object, using independent motion segmentation, is detailed in section 4.4. Section 4.5 shows how wide base-line matching is used to associate repeated appearances of an object within a shot. The performance of the object-level retrieval is assessed against ground truth in section 4.6. Finally, in section 4.7, the proposed method and its possible extensions are discussed.

We illustrate the method on objects in the feature film 'Groundhog Day' [Ramis, 1993]. The film has 141K frames and 752 shots.

4.2 Tracking affine covariant regions

In this section, we describe how affine covariant regions are tracked (associated) through a shot. As outlined in section 3.2, two types of affine covariant regions are detected independently in each frame. Detected regions in one frame of video are shown in figure 4.2.

The goal is to develop long tracks throughout the video, which can be used later for object extraction using motion segmentation techniques. Ideally, a patch on the object surface should be tracked whenever it is visible in the video. To achieve this, we devise a two stage tracking algorithm. In the first 'basic tracking' stage, we match existing (detected) regions between pairs of consecutive frames. In the second 'repair' stage, we extend and join the basic tracks by generating new regions not previously detected by region detectors. The result is that gaps in tracks caused



(a)



(b)



(c)

Figure 4.2: **Example of affine covariant region detection.** (a) Frame 20 from the van shot. (b) Ellipses formed from 722 Shape Adapted (SA) regions. (c) Ellipses formed from 1269 Maximally Stable (MS) regions. Note the large number of regions detected in a single frame, and also that the two types of region detectors fire at different and complementary image locations.

by e.g. missing detections can be bridged and the resulting tracks are significantly longer. In the following subsections, we describe each stage in more detail.

4.2.1 Basic tracking

The basic tracking proceeds sequentially, looking at detected regions in only two consecutive frames at a time. The objective is to obtain correct matches between the frames, which can then be extended to multi-frame tracks. Two matching constraints are used here: first, incorrect matches can be removed by requiring consistency with multiple view geometric relations, second, the regions can be matched on their appearance. The first matching constraint is based on the motion of rigid objects, and the robust estimation of these relations for point matches is mature [60]. The constraint is applied here to the region centroids. The second matching constraint is on image appearance within the segmented region.

In more detail, in a pair of consecutive frames, detected regions in the first frame are putatively matched with all detected regions in the second frame, within a disparity threshold of 50 pixels. Two regions are deemed matched if their normalized cross-correlation exceeds 0.90. Finally, epipolar geometry is fitted between the two views using RANSAC [51], with an inlier threshold of 3 pixels. This step is very effective in removing outlying matches whilst not eliminating the independent motions which occur between the two frames.

The results of this tracking on a shot from the movie 'Groundhog Day' are shown in figure 4.3b. This shot is used throughout this chapter to illustrate the stages of the object-level grouping. Note that the tracks have very few outliers.

The implementation of this tracker is due to Schaffalitzky [122].

4.2.2 Short range track repair by region propagation

The basic stage tracker described above can fail for a number of reasons most of which are common to all such feature trackers: (i) no region (feature) is detected in a frame – the region falls below some threshold of detection (e.g. due to motion blur); (ii) a region is detected but not matched due to a slightly different shape; and, (iii) partial or total occlusion.

The causes (i) and (ii) can be overcome by short range track repair using motion and appearance,



(a) (b) (c)

Figure 4.3: Region tracking: (a) six frames from the van shot. The camera is panning right, and the van moves independently. (b) Frames with the basic region tracks superimposed (before repair). Each frame shows affine covariant regions tracked in that frame. For each tracked region shown, the tracked path of its centroid over the whole life time of the track (i.e. backwards and forwards in time) is shown by its (x, y) position. The path of the region centroid indicates the temporal extent of the track. (c) After short range repair. Note the much longer tracks on the van after applying this repair. For presentation purposes, only tracks lasting for more than 10 frames are shown. Note that the background is not tracked in the middle of the shot due to severe motion blur. A detail of a single region track is shown in figure 4.6.

and we discuss this now. Cause (iii) can be overcome by wide baseline matching on motion grouped objects within one shot, and discussion of this is postponed until section 4.5.

The goal of the track repair is to improve tracking performance in cases where region detection or the first stage tracking fails. The method will be explained for the case of a one frame extension, the other short range cases (2-5 frames) are analogous.

The repair algorithm works on pairs of neighbouring frames and attempts to extend already existing tracks that terminate in the current frame. Each region which has been successfully tracked for more than $n(= 3)$ frames and for which the track terminates in the current frame is propagated to the next frame. The propagating transformation is estimated from a set of $k(= 5)$ spatially neighbouring tracks. In the case of successive frames only translational motion is estimated from the neighbouring tracks. In more detail, the t_x and t_y components of the translation are estimated as median values of the k translations t_{xi} and t_{yi} suggested by the k spatially nearest tracks i continuing to the next frame. Figure 4.4 shows an example. In the case of more separated frames the full affine transformation imposed by each tracked region should be employed.

The refinement algorithm of Ferrari *et al.* [47] is used to fit the propagated region locally in the new frame (this searches a hypercube in the 6D space of affine transformations by a sequence of line searches along each dimension). If the refined region correlates sufficiently with the original region in the previous frame the region track should continue to the new frame. It is here that the advantage of regions over interest points is manifest: this verification test takes account of local deformations due to viewpoint change, and is very reliable.

The standard 'book-keeping' cases then follow: (i) no new region is instantiated (e.g. the region may be occluded in the frame); (ii) a new region is instantiated, in which case the current track is extended; (iii) if the new instantiated region matches (correlates with) an existing region in its (5 pixel) neighbourhood then this existing region is added to the track; (iv) if the matched region already belongs to a track starting in the new frame, then the two tracks are joined.

Figure 4.5 gives the 'before and after' histogram of track lengths for the two example shots of figure 4.1. The results of this repair are shown in figures 4.3 and 4.8. Detail of a single region track after the repair stage is shown in figure 4.6. As can be seen, there is a dramatic improvement in the length of the tracks – as was the objective here.

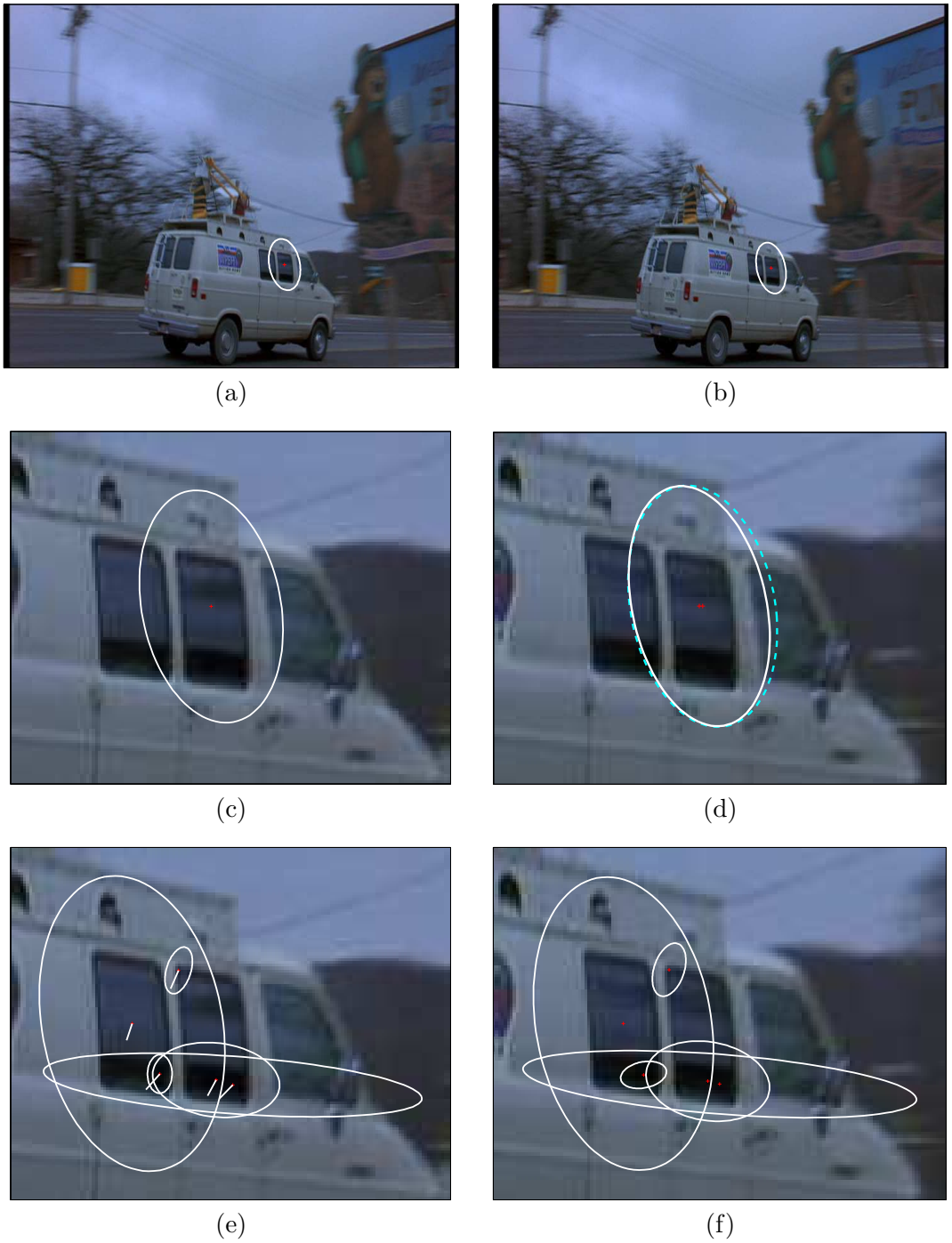
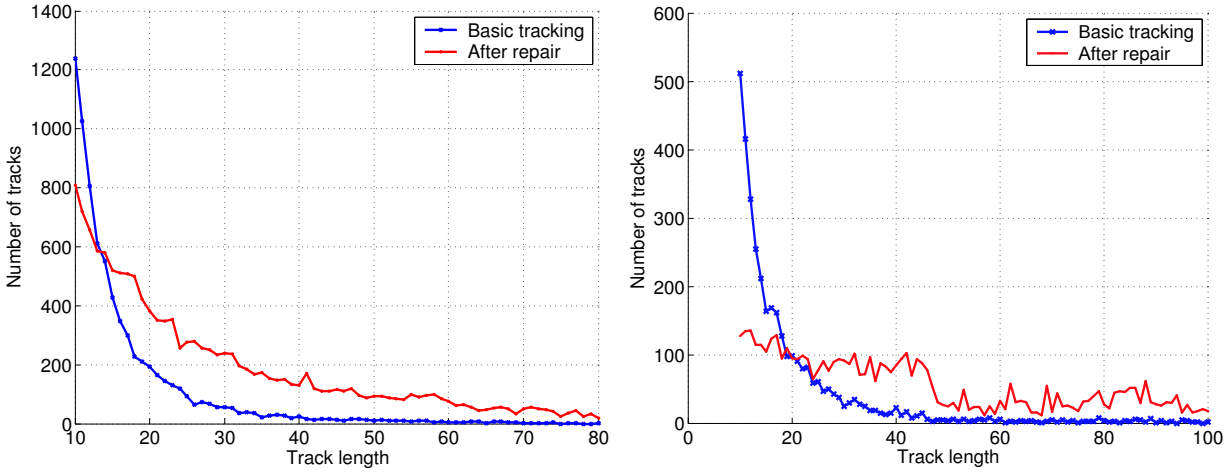


Figure 4.4: **Illustration of the track repair by region propagation.** A region track finishing in frame (a) is extended to the following frame (b). The region from the first frame (close-up shown in (c)) is first transformed to the next frame (dashed ellipse in (d)) and then aligned to the image intensities (solid ellipse in (d)). The initial propagation transformation (translation in this case) is estimated from the five (spatially) nearest already existing basic stage tracks. These are shown in (e) and (f). The lines in (e) show the centroid motion of each of the five tracked regions. See text for more details.



(a) Face (b) Van

Figure 4.5: Histograms of track lengths for (a) the face shot, (b) the van shot shown of figure 4.1 for the basic tracking (section 4.2.1) before and after short range track repair (section 4.2.2). Note the improvement in track length after repair. In both cases the weight of the histogram shifts to the right after repair. The step at around frame 45 after repair in (b) is due to the rich background of trees which lasts for about 45 frames at the beginning of the shot.

Note also that the region propagation can develop tracks on deforming objects where the between-frame region deformation can be modelled by an affine geometric transformation. Figure 4.9 shows an example of such a track on the mouth of a speaking person.

4.2.3 Discussion

The region tracker described in this section is based on establishing correspondence between affine covariant regions in consecutive frames of the video using global multi-view constraints. Similar approach is used for tracking interest points in the context of structure from motion [107, 150] and motion segmentation [149]. It is worth remarking on how tracking affine covariant regions compares to tracking interest points alone [89]. There are two clear advantages in the region case: first, the region appearance is a strong disambiguation constraint, and consequently fewer outliers are generated at every stage; second, more of the image can be tracked using (two types of) regions than just the area surrounding an interest point. The disadvantage is the computational cost, but this is not such an issue in the retrieval situation, where most processing can be done off-line.

Some point tracking approaches also model the dynamics of the camera/object motion using e.g. a Kalman Filter. An example is the impressive MonoSLAM system by Davison *et al.* [31],

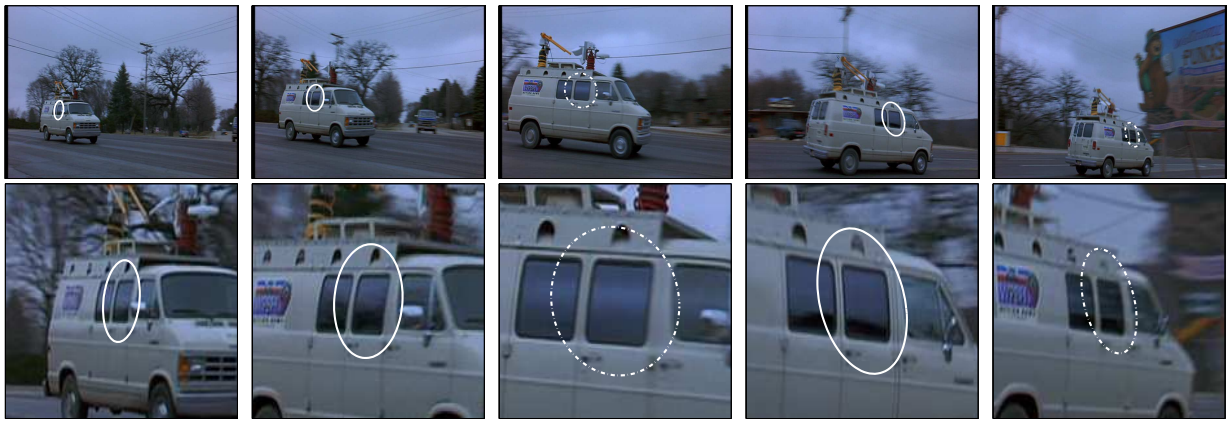


Figure 4.6: Detail of a single region track after repair. Note the significant change in viewpoint. Top row: Five frames from the shot of figure 4.3 with the tracked affine covariant region superimposed. Bottom row: Corresponding close-ups of the tracked region. The solid line denotes regions detected by the affine covariant region detector. The dashdot line denotes regions filled-in by propagation. This particular track extends over 52 frames and consists of 4 basic tracks (a total of 18 detected regions) connected by 34 filled-in regions (including 4 regions which were detected but not associated with any basic track).

which tracks the motion of the camera in real-time by tracking a small number (10-20) of interest point features in the input video stream. Modelling the motion dynamics brings the possibility of predicting the position of points in the next frame, thereby greatly reducing the computational cost of feature search. However, when the model is inaccurate, these predictions might be wrong, causing the tracker to fail.

4.3 Application I: Using multiple exemplars for retrieval

The goal here is for a user to be able to specify an object of interest in a single frame, by defining a query region delineating the object, and this to be sufficient input to retrieve all shots containing instances of that object throughout the movie, even though the object may deform or be imaged from a different visual aspect than that of the query frame (see section 4.6).

To achieve this, tracks of affine covariant regions throughout the shot are used to automatically associate multiple image exemplars of the object – query regions in other frames – and use the associated exemplars to enhance the original user specified query. The idea is illustrated in figure 4.7.

In detail a query region is 'transported' from the query frame to other frames in the shot as

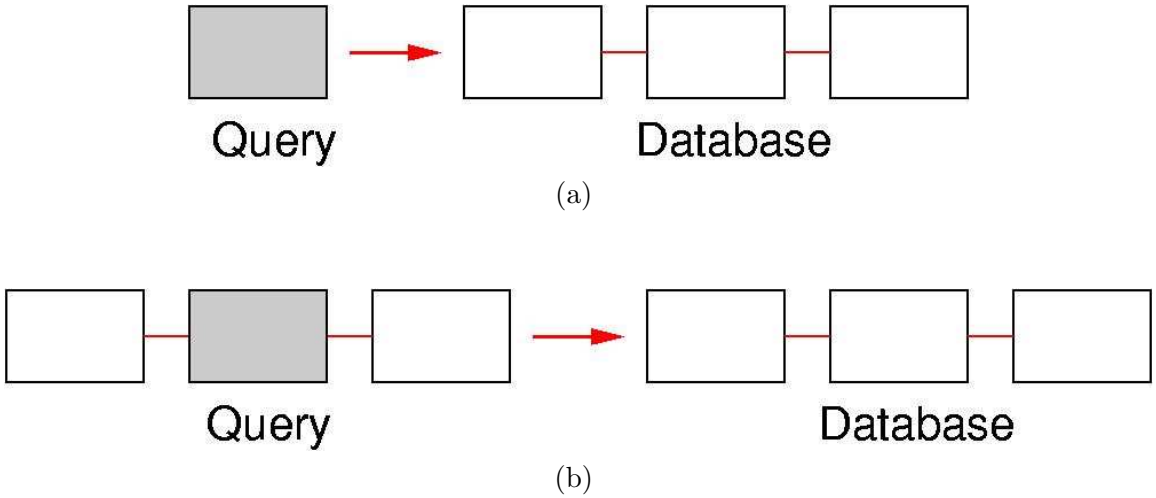


Figure 4.7: Conceptually, we extend the standard paradigm of image based retrieval (a), where the query is defined by a region within a single image, to retrieval at an object-level (b) where an object is defined over multiple images. A query region in the (shaded) query frame acts as a portal to all the keyframes and search regions within a shot associated by the tracked affine regions.

follows: the set of affine covariant regions enclosed by the query region is determined; the tracked regions then determine a corresponding set in each frame of the shot; in turn the rectangular bounding box (or union) of this set determines a query region for that frame. Matching is then carried out for all query regions using the object retrieval method described in chapter 3.

Figure 4.10 shows an example of an enhanced query. A user outlines a query rectangle in a single frame, as shown in figure 4.10a (top). Tracks on affine covariant regions passing through the user outlined rectangle then define associated query rectangles in other frames. The tracks are shown in figure 4.8. Tracking objects with a limited amount of deformation is possible since the region tracking described in section 4.2 allows a covariant region to undergo affine geometric transformation between consecutive frames of the video. A detail of a single region track on a deforming mouth is shown in figure 4.9.

The deforming and rotating object (actor's head talking and turning) is represented automatically by multiple exemplars (instances over multiple frames within one shot). The following sub-section gives implementation details.



(a)

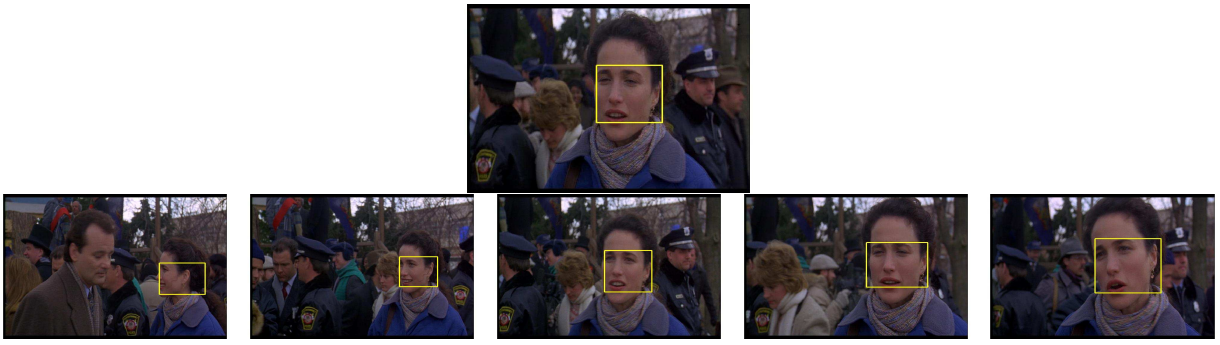


(b)

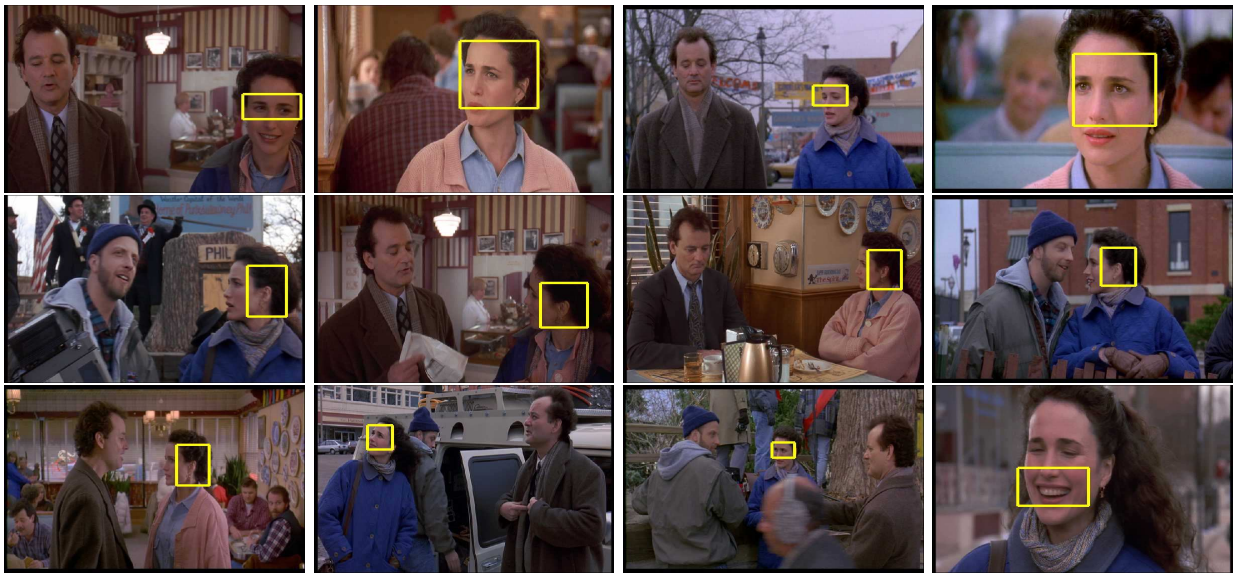
Figure 4.8: Tracking deforming objects. (a) Eight frames (of 133) for the head turning shot. (b) Tracked viewpoint covariant regions on the actor's head. The tracks are selected in one frame by a user query (see text). Only tracks longer than 10 frames are shown here. A detail of the mouth track is given in figure 4.9.



Figure 4.9: Detail of a region track in 10 consecutive frames covering the deforming mouth whilst the actor speaks. This track extends over 28 frames.



(a)



(b)

Figure 4.10: Retrieving a deformable object using multiple exemplars. (a) The user outlined query region (top) in a single frame, and (bottom) 5 (out of 19) automatically associated keyframes and query regions from within the same shot. The associated query regions are obtained as rectangular bounding boxes of the tracks (shown in figure 4.8) passing through the user outlined rectangle in the query frame. Note that full profile views, three quarter views and frontal views with different expressions are associated with the original query frame. (b) The top row shows example of retrieved frames from different shots by searching on only the user outlined query region. The bottom two rows show example retrieved frames by searching on the associated query regions as well. Note that the extended query enables the retrieval of full profile views which would be almost impossible by the original user outlined query. In the first twenty retrieved shots there are five mismatches for other faces and one mismatch for a non-face.

4.3.1 Collating search results from multiple queries

The goal here is to collate search results from multiple associated query frames representing the object level query in order to return a ranked list of shots. In more detail we want to compute a retrieval score Φ_l for shot l, given a set of query frames $\mathcal{S}_q = \{q_i\}$ (with query regions), the set of keyframes $\mathcal{S}_l = \{k_j\}$ belonging to shot l, and a keyframe scoring function $\phi(q, k)$ returning the similarity score between the query region of the query frame q and keyframe k. Here $\phi(q, k)$ is given by the sum of spatial consistency votes and the frequency based score, as explained in section 3.4.3, normalized by the number of affine covariant regions in query frame q. The normalization assures that scores from query frames with different number of query regions are comparable.

Two strategies are used for collating results from multi image queries [7, 25]: (i) votes for a particular shot are accumulated across all the associated query frames and retrieved keyframes, i.e.

$$\Phi_l = \sum_{i=1}^{|\mathcal{S}_q|} \sum_{j=1}^{|\mathcal{S}_l|} \phi(q_i, k_j), \tag{4.1}$$

or (ii) the best matching keyframe from each shot is used to score the whole shot

$$\Phi_l = \max_{i,j} \phi(q_i, k_j). \tag{4.2}$$

The advantage of the first method (equation 4.1) is that a shot can accumulate votes from multiple query frames, whereas false positives tend not to be consistent. For example, if both the query and retrieved shots have profile and frontal view of a face, then the face shot can accumulate votes from both the profile and frontal query frames whereas the false positives would not be the same for the frontal and profile views and would therefore receive lower score. The advantage of the second strategy (equation 4.2) is that it does not overcount scores for longer shots. In the matching examples of figures 4.10, 4.21 and 4.24 the first strategy was used. In the matching example of figure 4.25 the second strategy was used.

4.4 Object extraction by robust sub-space estimation

The previous section used tracked affine covariant regions to associate multiple exemplars of an object. However, the method is limited in that it can't 'see around corners'. For example, if we select the three-quarter view of the van in figure 4.3a (second row), only the side and front of the van will be associated, not the back of the van, because only tracks originating in the original three-quarter view are used. In this section we take the grouping a stage further and partition the tracks into groups with coherent motion. In other words, things that move together are assumed to belong together. For example, in the shot of figure 4.3 the ideal outcome would be the van as one object – grouping the front, side and back even though these are not visible simultaneously in any single frame. We would also expect to obtain several groupings of the background.

The grouping constraint used here is that of common rigid motion, and we assume an affine camera model so the structure from motion problem reduces to linear subspace estimation. For a 3-dimensional object, our objective would be to determine a 3D basis of trajectories $\mathbf{b}_k^i, k = 1, 2, 3,$ (to span a rank 3 subspace) so that (after subtracting the centroid) all the trajectories $\mathbf{x}_j^i$ associated with the object could be written as [177]:

$$\mathbf{x}_j^i = \left(\mathbf{b}_1^i, \mathbf{b}_2^i, \mathbf{b}_3^i\right) (X_j, Y_j, Z_j)^\top \tag{4.3}$$

where $\mathbf{x}_j^i$ is the measured (x, y) position of the jth point in frame i, and (X_j, Y_j, Z_j) is the 3D affine structure.

The maximum likelihood estimate of the basis vectors and affine structure could then be obtained by minimizing the reprojection error

$$\sum_{ij} \left|\left| n_j^i \left(\mathbf{x}_j^i - \left(\mathbf{b}_1^i, \mathbf{b}_2^i, \mathbf{b}_3^i\right) (X_j, Y_j, Z_j)^\top\right)\right|\right|^2 \tag{4.4}$$

where n_j^i is an indicator variable to label whether the point j is (correctly) detected in frame i, and must also be estimated. This indicator variable is necessary to handle missing data.

It is well known [152] that directly fitting a rank 3 subspace to trajectories is often unsuccessful and suffers from over-fitting. For example, in a video shot the inter-frame motion is quite slow so

using motion alone it is easy to under-segment and group foreground objects with the background.

We build in immunity to this problem from the start, and fit subspaces in two stages: first, a low dimensional model (a projective homography) is used to hypothesize groups – this over-segments the tracks. These groups are then associated throughout the shot using track co-occurrences. The outcome is that trajectories are grouped into sets belonging to a single object. In the second stage 3D subspaces are sampled from these sets, without over-fitting, and used to merge the sets arising from each object. These steps are described in the following sub-sections. The complete algorithm is summarized in figure 4.19. This approach differs fundamentally from that of [2, 32] where robustness is achieved by iteratively re-weighting outliers but no account is taken of motion degeneracy.

4.4.1 Basic motion grouping using homographies

To determine the motion-grouped tracks for a particular frame, both the previous and subsequent frames are considered. The aim is then to partition all tracks extending over the three frames into sets with a common motion. To achieve this, homographies are fitted to each pair of frames of the triplet using RANSAC. In each RANSAC iteration, a four-tuple of tracks extending over the three frames is sampled and three homographies ($\mathtt{H}_{12}$, $\mathtt{H}_{13}$, $\mathtt{H}_{23}$) are computed. The set of inlying tracks is computed based on image reprojection error averaged over the three frames. The inlier threshold is set to a generous number of pixels (around 3 here). The inlying set is removed, and RANSAC is then applied to the remaining tracks to extract the next largest motion grouping, etc. This procedure is applied to all triplets of consecutive frames in the shot, i.e. the neighbouring triplets share two frames. In the next step motion groups are linked throughout the shot into an object.

4.4.2 Aggregating segmentation over multiple frames

The problem with fitting motion models to pairs or triplets of frames are twofold: (i) a phantom motion cluster corresponding to a combination of two independent motions grouped together can arise [149], and (ii) an outlying track will be occasionally, but not consistently, erroneously grouped together with one of the motion groups. In our experience these ambiguities tend not to be stable

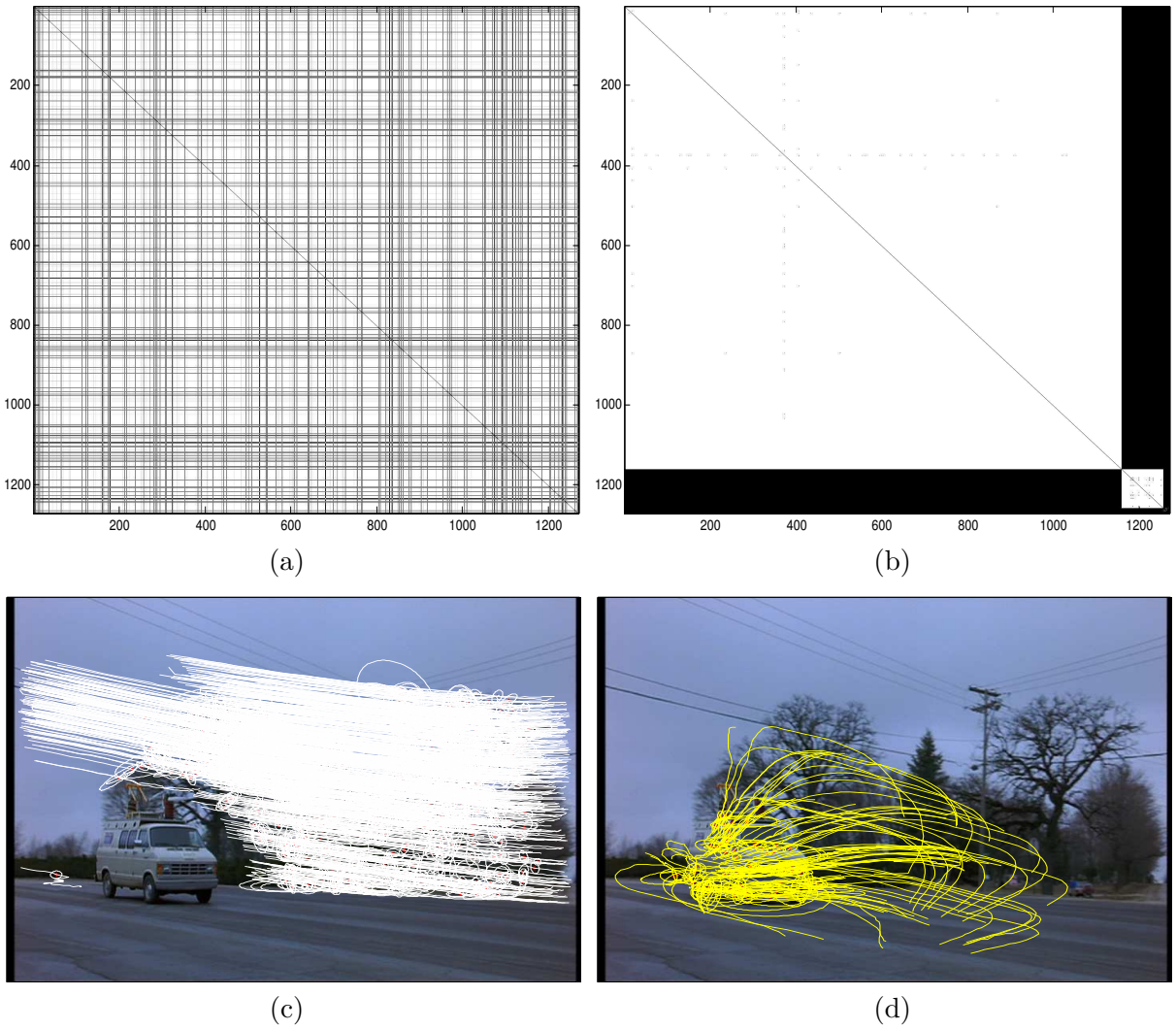


Figure 4.11: **Aggregating segmentation over multiple frames**. (a) The track co-occurrence matrix for a ten frame block of the shot from figure 4.3. White indicates high co-occurrence. (b) The thresholded co-occurrence matrix re-ordered according to its connected components (see text). (c) (d) The sets of tracks corresponding to the two largest components (of size 1157 and 97). The other components correspond to 16 outliers.

over many frames, but rather occasionally appear and disappear. To deal with these problems we devise a voting strategy which groups tracks that are consistently segmented together over multiple frames.

The basic motion grouping of section 4.4.1 provides a track segmentation for each triplet of consecutive frames. The goal is to pull out sets of tracks which are consistently grouped together over a wider baseline. This is achieved by a simple clustering algorithm which operates on a track-to-track similarity matrix, where the track-to-track similarity is based on temporal consistency between the two tracks, i.e. the number of frames over which the two tracks co-occur together in one motion segment (which is given by the basic homography based motion grouping).

In more detail the shot is divided into blocks of frames over a wider baseline of n frames ($n = 10$ for example) and a track-to-track co-occurrence matrix W is computed for each block. The element w_{ij} of the matrix W accumulates a vote for each frame where tracks i and j are grouped together. Votes are added for all frames in the block. In other words, the similarity score between two tracks is the number of frames (within the 10-frame block) in which the two tracks were grouped together. The task is now to segment the track voting matrix W into temporally coherent clusters of tracks. This is achieved by finding connected components [27] of a graph corresponding to the thresholded matrix W. To prevent under-segmentation the threshold is set to a value larger than half of the frame baseline of the block, i.e. 6 for the 10 frame block size. This guarantees that each track cannot be assigned to more than one group. Only components exceeding a certain minimal number of tracks are retained. Figure 4.11 shows an example of the voting scheme applied on a ten frame block from the shot of figure 4.3. This simple scheme segments the matrix W reliably and overcomes the phantoms and outliers.

The motion clusters extracted in the neighbouring 10 frame blocks are then associated based on the common tracks between the blocks. This is achieved by gradually progressing through frame blocks in the shot starting from the first block and associating motion clusters which are connected by a significant number of tracks. Significance is measured relative to the number of tracks in both the motion clusters, i.e. two motion clusters in the neighbouring blocks have to share at least 50% of their tracks to be associated. The result is a set of connected clusters of tracks which correspond to independently moving objects throughout the shot.

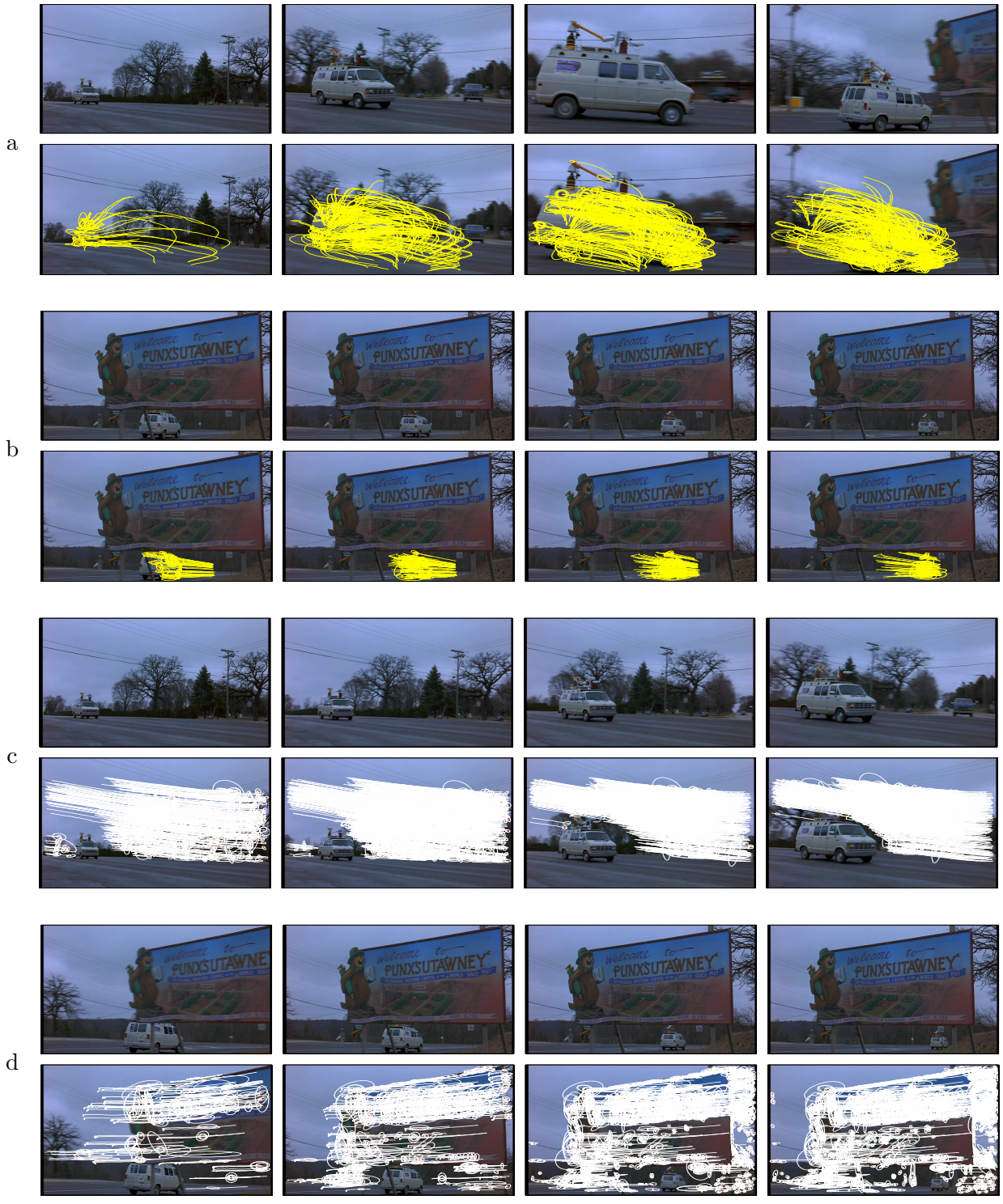


Figure 4.12: Motion grouping example I: Four dominant objects extracted from the van shot of figure 4.3. (a),(b) The first two objects corresponds to the van before (a) and after (b) the occlusion by the post. Note the billboard post right behind the van in the top left image of (b). This post partially occludes the van in 21 frames. (c),(d) The other two objects correspond to the background at the beginning (a) and the end (b) of the shot. The background in the middle of the shot was not tracked due to severe motion blur.

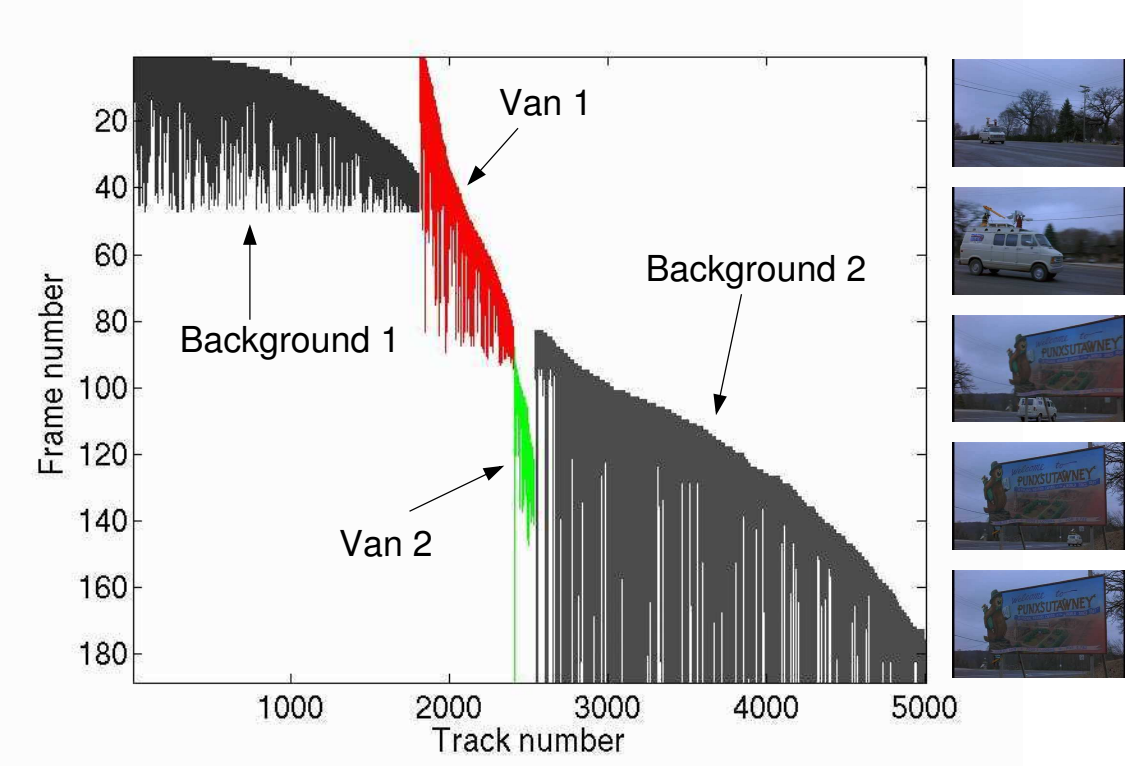


Figure 4.13: The sparsity pattern of the tracked features (after the short range track repair) in the van shot of figure 4.3. The tracks are sorted according to the frame they start in and coloured according to the independently moving objects, that they belong to, as described in section 4.4. The two gray blocks (track numbers 1-1808 and 2546-5011) correspond to the two background objects. The red and green blocks (1809-2415 and 2416-2545 respectively) correspond to the van object before and after the occlusion.

4.4.3 Object extraction

The previous track clustering step usually results in no more than 10 dominant (measured by the number of tracks) motion clusters larger than 20 tracks. The goal now is to identify those clusters that belong to the same moving 3D object. This is achieved by grouping pairs of track-clusters over a wider baseline of m frames ($m > 10$ here). To test whether to group two clusters, tracks from both sets are pooled together and a RANSAC algorithm is applied to all tracks intersecting the m frames. The algorithm robustly fits a rank 3 subspace as described in equation (4.4).

In each RANSAC iteration, four full tracks are selected and full affine factorization is applied to estimate the three basis trajectories which span the three dimensional subspace of the ($2m$ dimensional) trajectory space. All other tracks that are visible in at least five views are projected onto the space. A threshold (1.5 pixels) is set on reprojection error to determine the number of

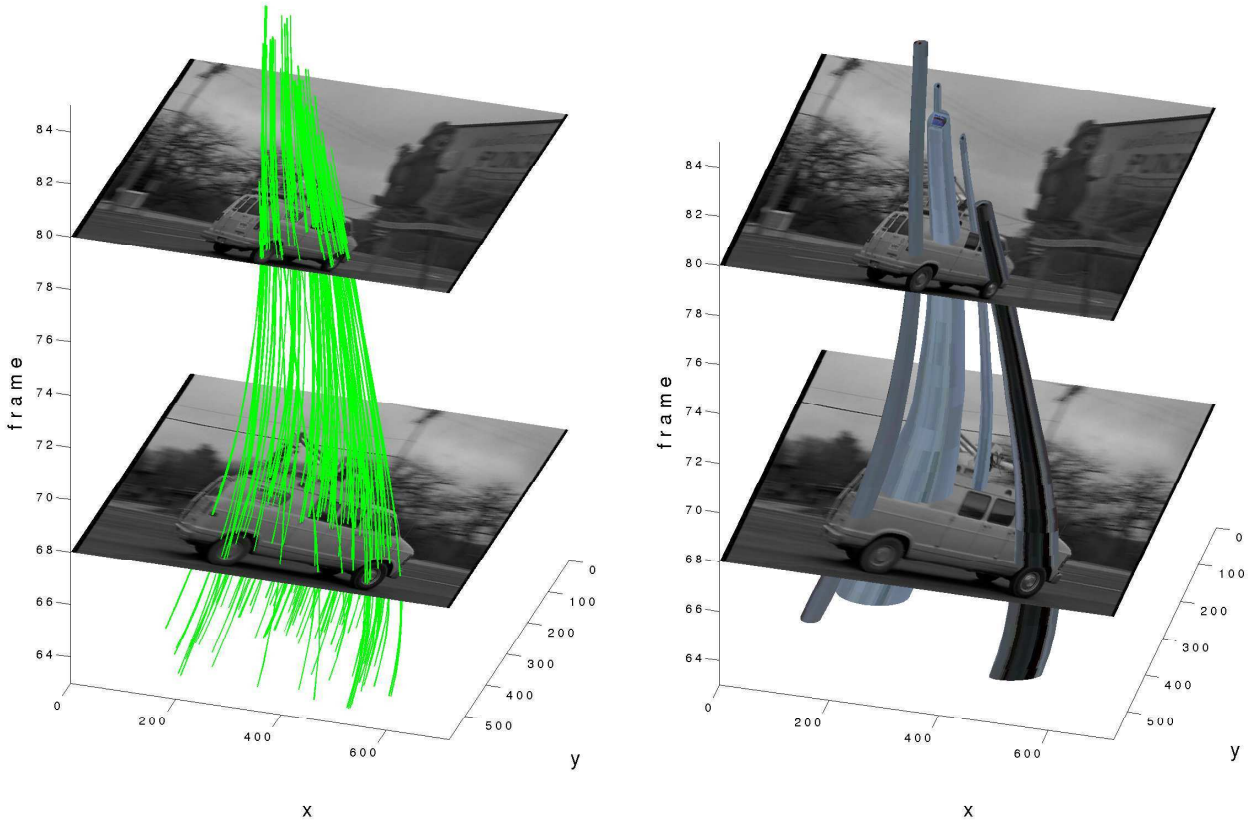


Figure 4.14: **Trajectories following object-level grouping**. Left: A selection of 110 region tracks (out of a total of 429 between these frames) shown by their centroid motion. Right: Five region tracks shown as spatio-temporal "tubes" in the video volume. The frames shown are 68 and 80. Both figures clearly show the foreshortening as the car recedes into the distance towards the end of the shot. The number and quality of the tracks is evident: the tubes are approaching a dense epipolar image [19], but with explicit correspondence; the centroid motion demonstrates that outlier 'strands' have been entirely 'combed' out, to give a well conditioned track set.



Figure 4.15: Motion grouping example II: object-level grouping for a 35 frame shot. Top row: The original frames of the shot. Middle and bottom row: The two dominant (measured by the number of tracks) objects detected in the shot. The number of tracks associated with each object is 721 (car) and 2485 (background).

inliers. To prevent the grouping of inconsistent clusters a high number of inliers (90%) from both sets of tracks is required. When no more clusters can be paired, all remaining clusters are considered as separate objects.

The rigidity based grouping is currently applied only to pairs of track-clusters. Complex objects made of more than two track-clusters could be handled by iterative merging pairs of clusters into larger groups.

4.4.4 Object extraction results

Figure 4.12 shows the four grouped objects for this example shot. Two of the objects correspond to the van (before and after the occlusion by the post, see figure 4.20 in section 4.5) and two correspond to the backgrounds at the beginning and end of the shot. The number of tracks associated with each object are 607 (van pre-occlusion), 130 (van post-occlusion), 1808 (background start) and 2466 (background end). The sparsity pattern of the tracks belonging to different objects is shown in figure 4.13. Each of the background objects is composed of only one motion cluster. The van (pre-occlusion) object is composed of two motion clusters of size 580 and 27 which are joined at the object extraction RANSAC stage. The quality and coverage of the resulting tracks is visualized in



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

Figure 4.16: Motion grouping example III: Object-level grouping for a 153 frame shot where the camera is tracking a van followed by another car. (a) Seven frames of the shot. (b)–(d) The three extracted objects correspond to (b) the van (1108 tracks), (c) the background (4481 tracks) and (d) the other car (210 tracks). The trajectory of the regions is not shown here in order to make the clusters visible.



Figure 4.17: Motion grouping example IV: object-level grouping for a 83 frame shot. Top row: The original frames of the shot. Middle and bottom row: The two dominant (measured by the number of tracks) objects detected in the shot. The number of tracks associated with each object is 225 (landlady) and 2764 (background). The landlady is an example of a slowly deforming object.



Figure 4.18: Motion grouping example V: object-level grouping for a 645 frame shot. Top row: The original frames of the shot where a person walks across the room while tracked by the camera. Middle and bottom row: The two dominant (measured by the number of tracks) objects detected in the shot. The number of tracks associated with each object is 401 (the walking person) and 15,053 (background). The object corresponding to the walking person is a join of three objects (of size 114, 146 and 141 tracks) connected by a long range repair using wide baseline matching, see figure 4.20b. The long range repair was necessary because the tracks are broken twice: once due to occlusion by a plant (visible in frames two and three in the first row) and the second time (not shown in the figure) due to the person turning his back on the camera. The trajectory of the regions is not shown here in order to make the clusters visible.

1. Detect affine covariant regions in each frame of the video.
2. Track the regions throughout the shot (section 4.2.1).
3. Extend tracks and bridge short gaps by region propagation (section 4.2.2).
4. Extract one or more dominant objects from a shot using motion grouping.
   (a) Basic motion grouping. Fit lower dimensional motion model by homography tracking (section 4.4.1).
   (b) Aggregate segmentation over multiple frames based on consistency (section 4.4.2).
   (c) Merge objects into larger ones by fitting 3D subspaces on wider baseline (section 4.4.3).
   (d) Extract all objects larger than t (=20) tracks from the sequence.

Figure 4.19: Object-level grouping algorithm. Associate independently moving objects within a shot using rigid motion consistency.

the spatio-temporal domain in figure 4.14.

Two additional examples of rigid object extraction from different shots are given in figures 4.15 and 4.16. Figures 4.17 and 4.18 show examples of slowly deforming objects. This deformation is allowed because at the first homography based stage rigidity is only applied over a short baseline of three frames.

Computation time: To give some idea of how long the object-level grouping takes we have recorded computation times for the example van shot of figure 4.3. This shot has a total of 187 frames. The region detection and descriptor computation took on average 11 seconds per frame. The basic tracking took 16 minutes ($\sim$5 seconds per frame). The track repair by region propagation took 304 minutes ($\sim$97 seconds per frame). The track repair is currently implemented in Matlab and is the bottleneck of the algorithm. The motion grouping algorithm took 56 minutes of which stage 4a took 12 mins, 4b 23 mins and 4c 21 mins. The different stages refer to the algorithm summary in figure 4.19. The motion grouping algorithm is also entirely implemented in Matlab. All timings are on a 2GHz machine.

4.5 Long range track repair

The object extraction method described in the previous section groups objects that are temporally coherent. The aim now is to connect objects that appear several times throughout a shot, for example an object that disappears for a while due to occlusion. Typically a set of tracks will terminate simultaneously (at the occlusion), and another set will start (after the occlusion). The situation is like joining up a cable (of multiple tracks) that has been cut.

The set of tracks is joined by applying standard wide baseline matching [91, 121, 156] to a pair of frames that each contain the object. There are two stages: first, epsilon-nearest neighbour search on a SIFT descriptor [85] for each region is performed to get a set of putative region matches, and second, this set is disambiguated by a local spatial consistency constraint: a putative match is discarded if it does not have a supporting match within its k-nearest spatial neighbours [124, 138]. Since each region considered for matching is part of a track, it is straightforward to extend the matching to join tracks. The two objects are deemed matched if the number of matched tracks exceeds a threshold. Figure 4.20 shows two examples of long range repair on shots where the object was temporarily occluded.

4.6 Application II: Object-level video matching

The objective here is to retrieve shots within the film containing the object, even though the object may be imaged from a different visual aspect than in the query image region.

Having computed object-level groupings for shots throughout the film, we are now in a position to retrieve object matches given only one visual aspect of the object as a query region. As in the application engineered in section 4.3, a query region in one frame acts as a portal to a set of associated query regions – but here the association is on common 3D motion as described in section 4.4. (In fact since the object has been segmented it is only necessary for the user to 'click' on the object in one frame).

The associated query regions form an implicit representation of the 3D structure, and are sufficient for matching when different visual aspects or parts of the object are seen in different frames of the shot. As shown in figures 4.21 and 4.24, associated frames naturally span the object's



(a) (b)

Figure 4.20: Two examples of long range repair on (a) shot from figure 4.3 where a van is occluded (by a post) which causes the tracking and motion segmentation to fail, and (b) shot from figure 4.18 where a person walks behind a plant. First row: sample frames from the two sequences. Second row: wide-baseline matches on regions of the two frames. The green lines show links between the matched regions. Third row: region tracks on the two objects that have been matched in the shot.

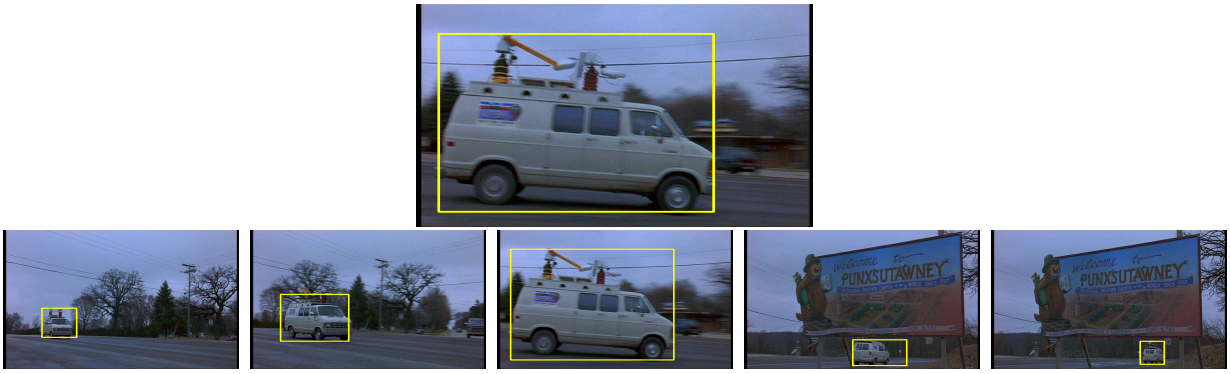
visual aspects contained within the shot.

Examples of object-level matching throughout a database of 5,640 keyframes of the entire movie 'Groundhog Day' is shown in figures 4.21, 4.24 and 4.25. In all cases false positives were also retrieved. Retrieval performance for two of the examples is discussed in more detail in the following section.

4.6.1 Retrieval performance

The van query: Ground truth was obtained for the van query in figure 4.21 by marking all keyframes and shots where the van appears in the movie. In order to be deemed present in a frame, the van was required to be at least 100 pixels across (in frames that are 720×576 pixels).

Precision-recall curves on the shot level for the object-level matching example from figure 4.21 are shown in figure 4.22. In the case of precision-recall curves (a) and (b) where multiple images were used as query frames, each query frame was used to place a separate query and the results from all queries were then pooled together. Retrieved shots were ranked as described in section 4.3.1(p94),



(a)



(b)

Figure 4.21: Object-level video matching I. (a) Top row: the query frame with the query region (side of the van) selected by the user. (a) Second row: 5 (out of 6) associated frames and outlined query regions. The query frame acts as a portal to the frames (and query regions) associated with the object by the motion-based grouping. (b) Top row: example frames retrieved from the entire movie when only the original user selected frame with user outlined region is used. (b) Rows 2-4: Example frames retrieved from the entire movie by the object-level query (second row of (a)). Note that views of the van from the back and front are retrieved. This is not possible with wide-baseline matching methods alone using only the side of the van visible in the query image. In this figure, only true positives are shown. Precision recall curves for this query are shown in figure 4.22.

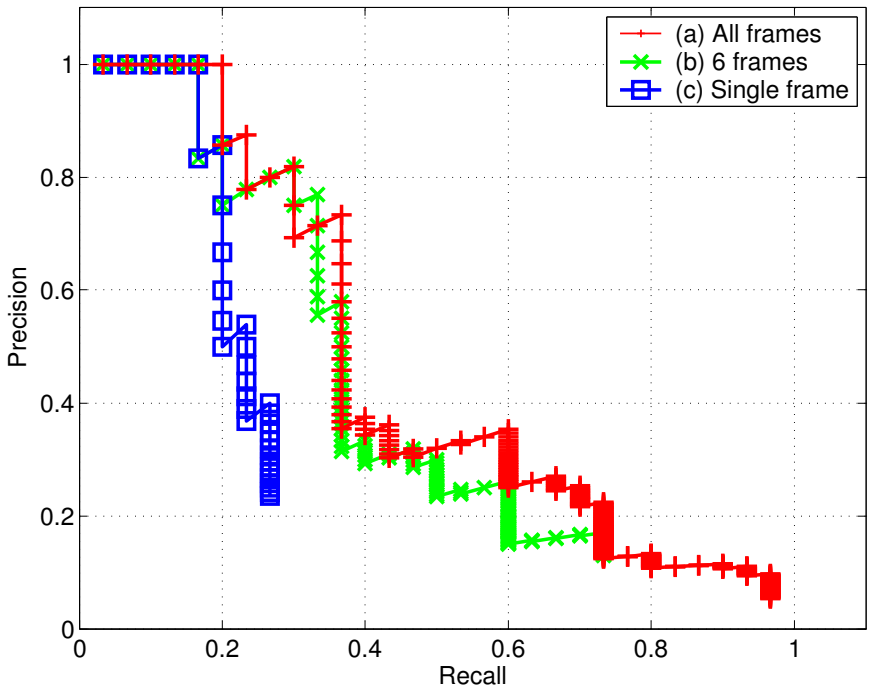


Figure 4.22: Object-level video matching I. Precision recall curve for the van query at the shot level. Examples of retrieved frames are shown in figure 4.21. (a) All frames in the query shot are used as query frames. (b) 6 frames in the shot are used as query frames. (c) A single frame (the original frame with user outlined region) is used as a query frame. Note the limited recall of (c). This is because only shots where the side of the van is visible are retrieved.

equation 4.1. Shots with no spatial consistency votes (section 3.4.3(p51)) were deemed not matched and therefore not retrieved.

Note that the user outlined query frame (curve (c) in figure 4.22) recalls only 27% of all the ground truth shots containing the van. This is because the query frame contains only the side of the van (see figure 4.21a (top)) and therefore it is possible to retrieve only shots where the side of the van is visible. When the object is represented by a set of keyframes (1 per second) naturally spanning its visual aspects (curve (b) in figure 4.22) the recall jumps to 73%. This is because shots containing the front and back of the van are also retrieved. Representing the object by *all* the frames in the shot (curve (a) in figure 4.22) brings the recall to 97%. The slight improvement in precision of curve (a) is mainly due to score accumulation as described in section 4.3.1.

False positives responsible for lower precision at higher recall levels (e.g. 35% precision for 60% recall in figure 4.22(a)) are mainly due to (i) motion blur, which affects the affine covariant region matching, and (ii) generally low number of good matches on the van.

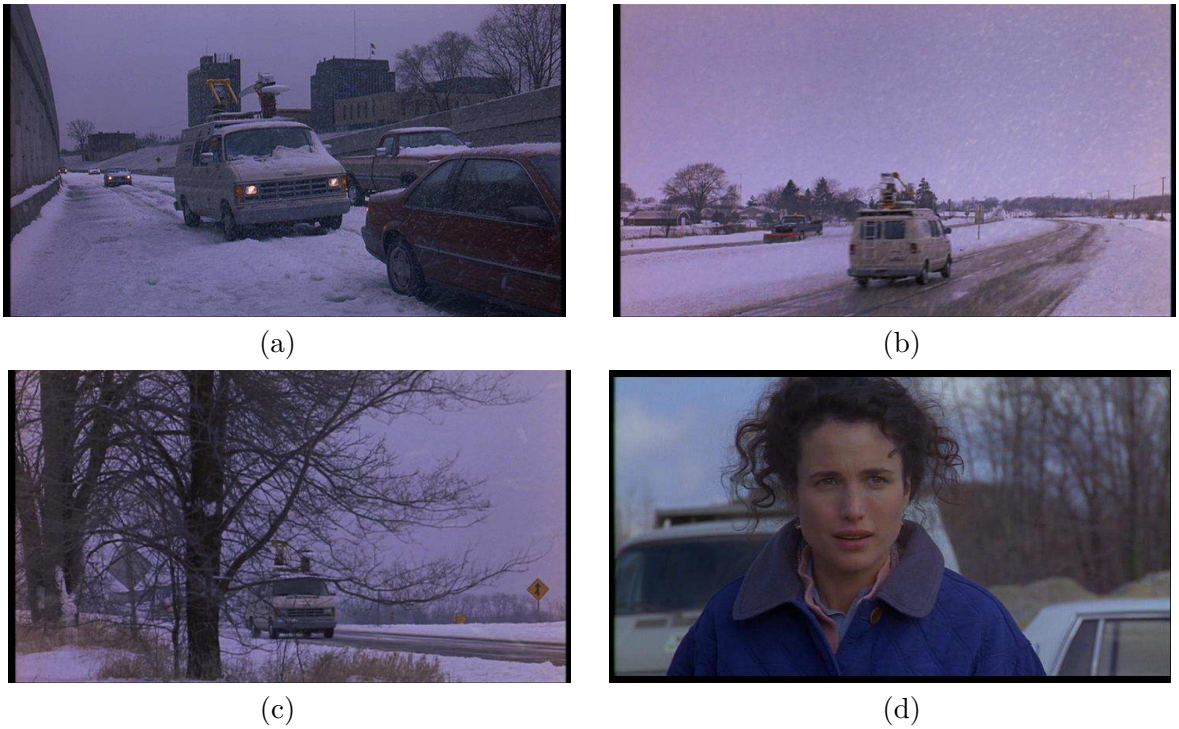


Figure 4.23: Example frames from low ranked (a–c) and missed (d) shots for the van query (figure 4.21). The altered appearance due to snow in (a, b), and partial occlusion (c, d) affects the affine covariant region extraction and matching methods.

The precision could be improved further by the removal of false positives based on a more thorough (and more expensive) verification, e.g. by the image exploration algorithm of [49].

Examples of frames from bottom ranked and missed shots are shown in figure 4.23. They represent very challenging examples for the current object matching method.

The Dining room query: Here the match is on the background location, rather than on the foreground moving object. Ground truth for the query of figure 4.25 was obtained by marking all shots in the movie which are taken in the hotel dining room. The precision-recall curve is shown in figure 4.26. The improved recall of (a) and (b) over (c) is due to the object-level query retrieving shots from the same location but with different background than the original query frame. The improved performance of (a) over (b) is due to better sampling of the background in the beginning of the shot with large camera motion where keyframes (every 25th frame) miss some parts of the background. A better keyframe selection technique [106] based on motion within the shot could be used here.



(a)



(b)

Figure 4.24: Object-level video matching II. (a) Top row: the query frame with query region selected by the user. (a) Bottom row: The associated keyframes. Note that in the associated keyframes the person is visible from the front and also changes scale. See figure 4.18 for the corresponding object segmentation. (b) Example frames retrieved from the entire movie by the object-level query.



(a)



(b)

Figure 4.25: Object-level video matching III. The goal is to retrieve shots in the same location (the hotel dining room). (a) Top row: the query frame with the query region selected by the user. Bottom row: 5 (out of 25) associated keyframes. The object here is the extended background from the object-level grouping example of figure 4.18. The query area in each associated frame is the union of the motion grouped background regions. (b) Top row: Example frames from shots retrieved just by the user selected query frame. Bottom row: Example frames from shots retrieved by the object-level query. Query by the extended background retrieves shots which are from the same location but do not share background with the user selected query frame. The precision-recall curve for this query is shown in figure 4.26.

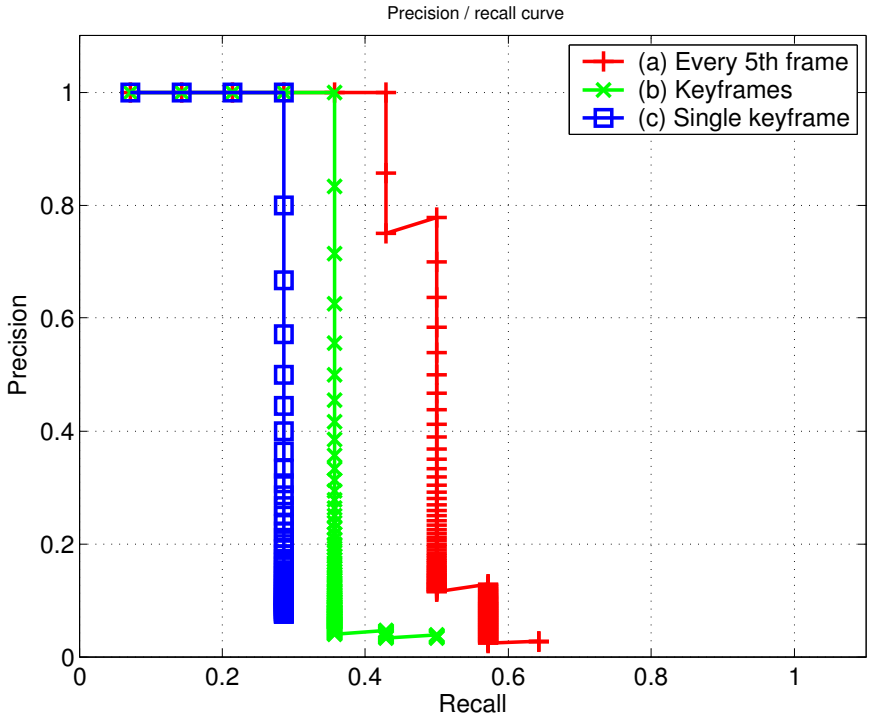


Figure 4.26: Object-level video matching III. Precision-recall curve for the dining room query. Examples of retrieved frames are shown in figure 4.25. (a) Every fifth frame in the query shot is used as a query frame (127 frames in total). (b) 25 keyframes used as query frames. (c) A single frame (the original frame with the user outlined region) is used as a query frame.

Note that some shots from the dining room are still not retrieved. This is because in the missed shots the camera looks at the other side of the room which is not covered in the query shot. To retrieve these shots a higher level reasoning might be required e.g. the temporal editing structure of shots can be used to group shots into scenes [55, 71]. An alternative method of matching only background locations using wide baseline matching is given in [122]. In our work the user has a choice of whether to search on foreground or background object(s).

4.7 Conclusions and discussion

We have demonstrated that information available in video shots can be harnessed to enable object-level grouping and retrieval. This is different in spirit to query enhancement techniques in text retrieval [8], where the high ranked documents are used to enhance the original query. In our case we do not use the retrieved shots or frames to enhance the query but rather we make use of the

temporal continuity of the shot. The enhanced query is then performed by making a sequence of associated queries and collating the results.

There are several other research issues: First, in the matching stage of the current method we plan to represent the shot by entire region tracks ('video tubes') rather than the set of separate query frames/keyframes currently used. Using entire 'video tubes' could help to determine the required density of assocation: Imagine a close-up shot of a speaking person. Deforming region tracks on the person's face would be represented by several different appearance descriptors corresponding to different expressions, e.g. open and closed eyes, whereas region tracks on the (rigid) background would have just one appearance descriptor. 'Video tubes' should provide a complete but at the same time concise representation of video for recognition.

Second, a limitation of the current method is that multiple aspects/de-formations have to be present in the query shot. The next step is to use available region tracks within the (correctly) retrieved shots to perform the associations. For example, if the user supplies a query still image of a frontal view of an actor's face. Querying by this image alone will only return close-to-frontal views of the face with similar facial expressions. However, region tracks in the retrieved shots can be used to associate other views of the face and different expressions, which can then be used in a second set of queries. This process can be iterated. This would have to be done with some care in order to avoid a 'chain reaction' by matching on retrieved false positives.

Chapter 5

Video shot retrieval for face sets with face specific visual vocabulary

5.1 Introduction

The objective of this chapter is to retrieve shots containing particular people/actors in video material using an imaged face as the query. In the previous chapter we have shown face retrieval results using generic local image regions together with a generic visual vocabulary capable of retrieving any object in the video. As humans have special interest in searching for other people, in this chapter we explore an approach specifically designed for retrieving faces.

First, we employ a face detector instead of detecting generic local affine covariant regions in an image. Face detectors are trained from thousands of face examples to detect only human faces and help us to focus attention only on relevant parts of the image. Currently only a frontal face detector is used, but the proposed approach could be generalized to non-frontal faces when reliable arbitrary pose face detectors become available.

Second, we build a face specific visual vocabulary, where local regions are centred at facial features such as eyes, nose and mouth. Such face specific visual words are required to model only different appearances of facial features and no modelling power is wasted on modelling appearance of other objects.

Third, we take further the idea of object level grouping, developed in the previous chapter. Here

human faces are objects and we use motion in the video to associate face exemplars in different frames into a single model. The result is, similarly to the previous chapter, that an object (human face occurring in a video shot) is represented by a set of exemplars associated by tracking. This allows us to represent, say, different face expressions like opening and closing of the mouth. In contrast to the previous chapter, where object level grouping was performed only on the query side and the database was represented by a set of separate keyframes, in this chapter we perform object level grouping on the entire database.

In terms of the challenge faced, we have uncontrolled situations with strong lighting changes, occlusions and self-occlusion. Also we can have multiple people in a frame/shot.

Novelty: Face matching is notoriously difficult, see for example works by Duygulu and Hauptman [35], Eickler *et al.* [37], Fitzgibbon and Zisserman [52] and Satoh *et al.* [118]. Even under quite controlled conditions the variation in the imaged face due to lighting, pose, partial occlusion, and expression, can exceed that due to identity. As mentioned above, the approach we take is to eschew matching single faces but instead match *sets of faces* for each person, with the representation for each person consisting of a distribution over face exemplars. This approach has been investigated in the literature by for example Arandjelovic *et al.* [6], Bart *et al.* [11], Krueger and Zhou [74] and Shakhnarovich *et al.* [128]. However, we bring three areas of novelty: first, sets of face exemplars for each person are gathered automatically in shots using tracking (section 5.2); second, an individual face is represented as a collection of parts. Part based approaches to face recognition were investigated by for example Heisele *et al.* [61] and Wiskott *et al.* [174]. In our work, the feature vector describes local spatial orientation fields (section 5.3.2); third, a face set is represented as a distribution over vector quantized exemplars (section 5.3.3).

Our aim is to build a description which is largely unaffected by scale, illumination, and pose variations around frontal. Expression variation is then represented by a distribution over exemplars, and this distribution (which in turn becomes a single feature vector) is distinctive for each identity. This single feature vector for identity enables efficient retrieval.

Chapter outline: In section 5.2 we describe how face detections are associated into face sets by tracking affine covariant regions on faces. Section 5.3 details descriptors for single faces, and



Figure 5.1: Example face detections of the Julia Roberts' character in the movie 'Pretty Woman'. Note that detections are not always perfectly frontal. Note also successful detections despite varying lighting conditions, changing facial expressions and partial occlusions.

introduces representation and matching methods for sets of faces. In section 5.4 we describe the developed person retrieval application. Section 5.5 presents retrieval results on two feature length movies: 'Pretty Woman' and 'Groundhog Day'. Finally, in section 5.6 we conclude the chapter and discuss possible extensions.

The method will be illustrated on the 'opera' shot from 'Pretty Woman' shown in figure 5.2.

5.2 Obtaining sets of face exemplars by tracking

In this section we describe the method for associating detected faces within a shot in order to have multiple exemplars covering a person's range of expressions.

Face detection: A frontal face detector, developed by Mikolajczyk [95], is run on every frame of the movie. To achieve a low false positive rate a rather conservative threshold on detection strength is used, at the cost of more false negatives. The face detector is based on AdaBoost with weak classifiers built from local orientation detectors. Example face detections are shown in figure 5.1. Face detections in sample frames of the 'opera shot' are shown in figure 5.2. Alternatively, as shown in Choudhury *et al.* [24], a face detector for video could be used.



Figure 5.2: Sample frames from the 'opera shot' with face detections overlaid. The corresponding frame number is shown below each frame. This is a challenging shot with camera motion and multiple people that move around and occlude each other.

5.2.1 Associating detected face exemplars temporally

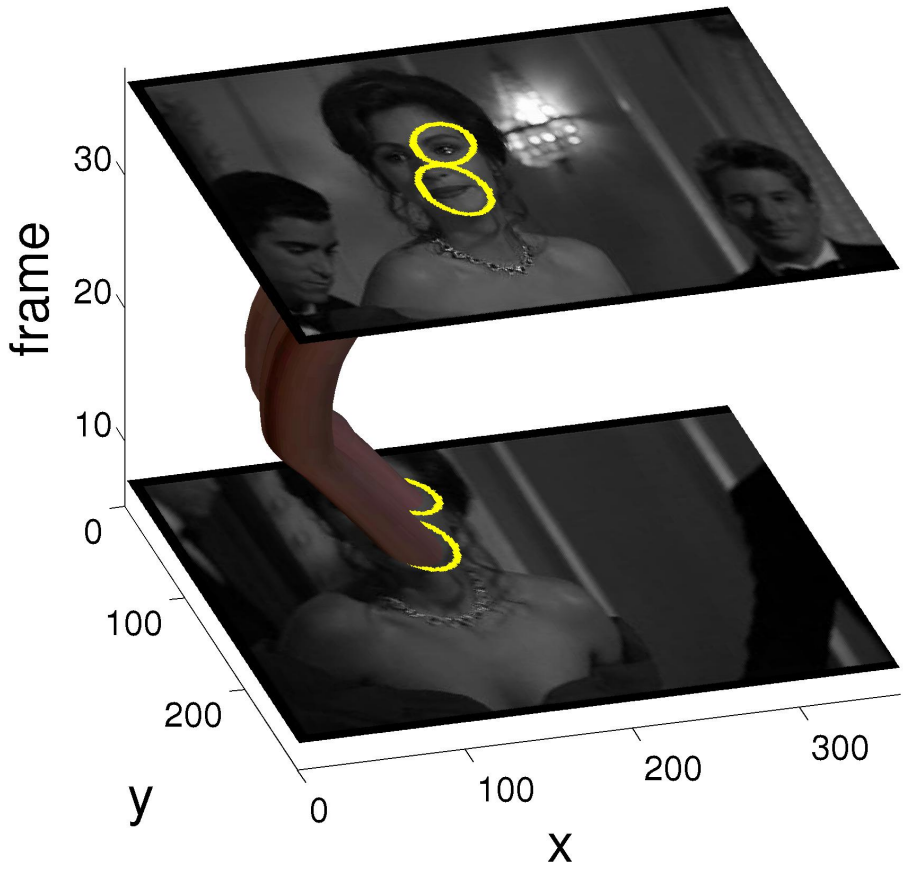
The objective here is to use tracking to associate face detections into *face-tracks* corresponding to the same person within a shot. This is achieved by first running a general purpose region tracker and then associating face detections in different frames based on the region tracks connecting them.

Region tracking: The affine covariant region tracker, described in section 4.2(p83), is used here. Figure 5.3(b) shows a typical set of tracked elliptical regions. This tracking algorithm can develop tracks on deforming objects (a face with changing expressions), where the between-frame region deformation can be modelled by an affine geometric transformation plus perturbations, e.g. a region covering an opening mouth. An example of a region track on the mouth of a speaking person is

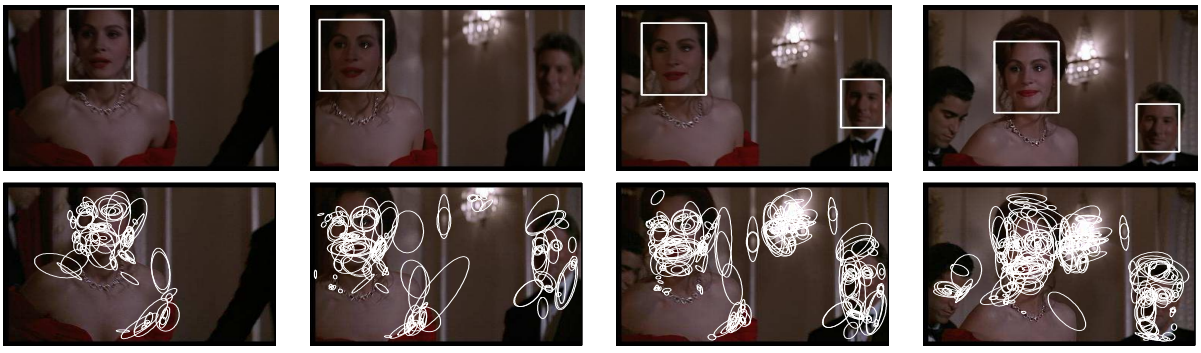
shown in figure 4.9(p92). The outcome is that a person's face can be tracked (by the collection of regions on it) through significant pose variations and expression changes, allowing association of possibly distant face detections. The disadvantage of this tracker is the computational cost but this is not such an issue as the tracking is done off-line. Note, the face detections themselves are not tracked directly because there may be drop-outs lasting over many consecutive frames (e.g. as the person turns towards profile and back to frontal). However, the region tracker survives such changes.

Connecting face detections using region tracks: A typical shot has tens to hundreds of frames with possibly one or more face detections in each frame. Face detections are usually connected by several region tracks as is illustrated in figure 5.3 – think of this as 'tubes' linking the detected rectangular face regions. Note, it is rare for a region track to extend through all face detections of a particular person in the shot. This is because actors often turn from the (detected) frontal pose to profile (causing several regions to be occluded), and occasionally a track may break due to e.g. fast expression change or severe motion blur. For these reasons a naive strategy associating only detections which have common tracks with a single (e.g. the first available) detection would often fail as tracks going from that first detection will eventually die out. To overcome this problem we use a single-link agglomerative grouping strategy which gradually merges face detections into larger groups starting from the closest (most connected) detections. We also utilize a temporal exclusion constraint in the clustering, not allowing face-tracks arising from distinct face detections in a single frame to be grouped.

In more detail, we form an $n \times n$ connectivity matrix $\mathtt{C}$, between all n face detections in the shot, i.e. initially each face detection is in a separate group. Element c_{ij} is set to the number of common region tracks between face detections i and j. The algorithm merges groups of face detections in order of connectivity. The most connected groups are merged first. When two groups, say i and j, are merged the matrix $\mathtt{C}$ is reduced by one row and column. Connectivity between the newly formed group, say l, and all other groups is replaced by $c_{lk} = \max\{c_{ik}, c_{jk}\}$. The temporal exclusion is implemented as a 'cannot link' constraint [72] by setting connectivity to zero for all groups which share the same frame. The merging is run until no two groups can be merged, i.e. have connectivity above a certain threshold (five region tracks in this work). This technique is very successful when

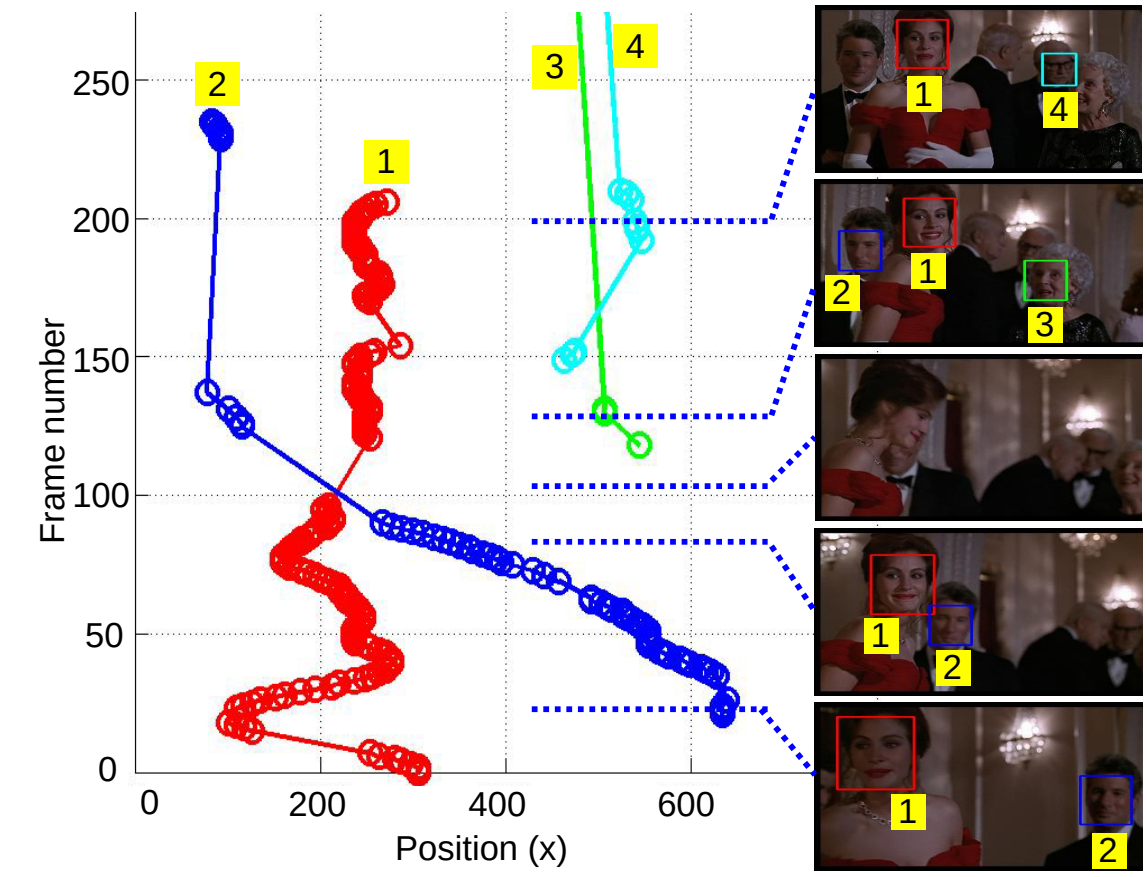


(a)

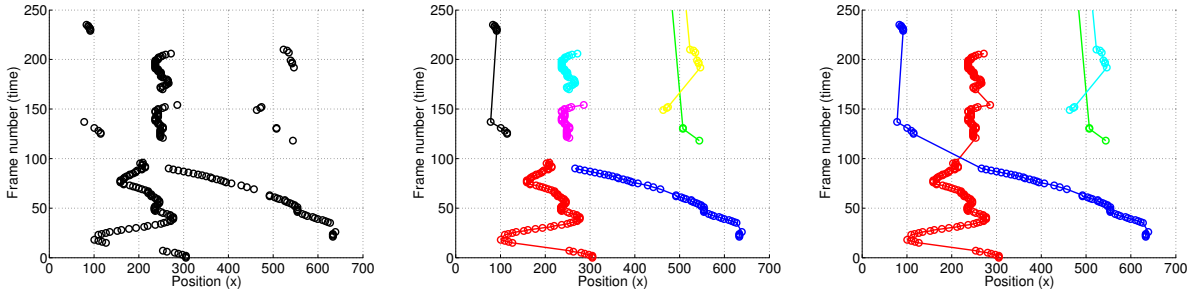


(b)

Figure 5.3: (a) Two region tracks as 'tubes' in the video volume between frames 7 and 37 of the 'opera shot' (shown in full in figure 5.4). The two tracked regions are superimposed in yellow. There are 27 region tracks on the actor's face between the two frames. These 'tubes' allow us to temporally associate face detections in different frames. The 'kink' in the tube arises when the actor moves first left and then right while standing up from a chair. At the same time the camera follows the actor's vertical motion. (b) Top: Four frames from the video volume with face detections superimposed. (b) Bottom: The same four frames with tracked regions superimposed. The frame numbers shown are 7, 17, 27, and 37 (from left).



(a)



(b) – face detections (c) – temporal association (d) – intra-shot matching

Figure 5.4: **Associating face detections within a shot.** (a) Overview of the first 250 frames of a shot where actors 1 and 2 cross while the camera pans to follow actor 1. Around frame 100 actor 1 turns away from the camera while occluding actor 2. Actors 3 and 4 appear and are detected later in the shot. The circles show positions of face detections. Face detections of the same character are colour coded and connected by lines. The thumbnails on the right show face detections numbered and colour coded according to the actor's identity. The face detections in the shot (shown in (b)) are connected temporally into face-tracks (shown in (c)). Note some face-tracks are still broken due to occlusion (actor 2) and self-occlusions (actor 1 turns away from the camera). These face-tracks are subsequently linked using intra-shot face-track matching (shown in (d)). The whole process is fully automatic. The temporal association and the intra-shot matching are described in sections 5.2 and 5.3.4 respectively.

region tracks between nearby face detections are available. An example of temporal associations of face detections is shown in figure 5.4(c).

Note that, ideally, all face detections of the same person in a shot should be grouped. However, this is sometimes impossible using temporal information alone, e.g. if a person is occluded for some time by another object/person. To overcome this we employ within shot matching of existing temporal face-tracks. We return to this matching in section 5.3.4 once we have introduced the face representation necessary for matching.

5.3 Representing and matching sets of face exemplars

In this section we describe our representation of face sets and the matching distance used to compare them. Each face in the (face-track) set is described by a collection of five affinely transformed local spatial orientation fields (SIFT [85] descriptors) based around facial features. The entire set is represented as a single distribution over these local feature descriptors. This turns matching sets of exemplars into comparing probability distributions. The following sections describe each of these steps in more detail.

5.3.1 Facial feature location

The goal here is to localize facial features (left and right eyes, tip of the nose and centre of the mouth) within a face detection. This allows us to place the local face descriptors and affinely deform their support regions to normalize for pose variations. As shown in figure 5.1 the face feature positions within the face detections vary considerably. This is mainly due to varying head pose and noisy face detector output, e.g. over scale.

A probabilistic parts-based 'constellation' model of faces is used to model the joint position (shape) and appearance of the facial features. A similar model was used by Fergus *et al.* [45] and Felzenszwalb and Huttenlocher [43]. To simplify the model, two assumptions are made: (i) the appearance of each feature is assumed independent of the appearance of other features, and (ii) the appearance of a feature is independent of its position. The position of the facial features is modelled as a single Gaussian with full covariance matrix. In contrast to other work [43, 45] the model does not need to be translation invariant as we expect the face detector to have approximately



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

Figure 5.5: (a) Original frame. (b) Close-up with face detection superimposed. (c) Detected facial features (eyes, nose, mouth). (d) Face is represented as a collection of local affinely deformed spatial orientation fields (SIFT descriptors). The green circles illustrate the location and support region for each of the five SIFT descriptors. Note how the position of the local regions adapts to the slightly rotated pose of the head in this case.

normalized the position of the face. The parameters of the model are learnt from around 5,000 hand-labelled face images taken from the web.

An example of detected feature points is shown in figure 5.5(c). The implementation of the facial feature detector is due to Everingham [39].

5.3.2 Representation of single faces

Each face in the set is represented as a collection of local overlapping parts. Part based approaches to face recognition have been used for example by Wiskott *et al.* [174], and have been shown by Heisele *et al.* [61] and Shakhnarovich and Moghaddam [129] to outperform global face description as they cope better with partial occlusions and pose variations. The disadvantage is that the process of facial feature detection is an additional source of possible errors. This becomes a significant factor for more extreme poses [61] where some of the salient components (eyes, mouth, nose) are not visible or extremely distorted. We try to exclude such cases by limiting ourselves to near frontal poses (by using a frontal face detector).

Our face representation consists of a collection of five overlapping local SIFT descriptors [87] placed at the detected feature locations (eyes, mouth, nose) and also at the mid point between the eyes. The intention is to measure local appearance (e.g. of an eye) independently and also, by the support region overlap, (e.g. of the two eyes) some *joint* feature appearance. Each local SIFT descriptor is an eight bin histogram of image gradient orientations at a spatial 3×3 grid. This gives a 72-dimensional descriptor for each local feature position, i.e. the joint feature for the

five regions is a 360-vector. The circular support regions of SIFT descriptors are deformed into ellipses by (the inverse of) an affine geometric transformation which maps feature locations within the face detection into a common canonical frame. This compensates for head pose variation to a certain degree, as is illustrated in figure 5.5(d). The SIFT descriptor has been shown superior to other local descriptors [94] because it is designed to be invariant to a shift of a few pixels in the feature position, and this localization error often occurs in the facial feature detection process. The SIFT descriptor is also invariant to a linear transformation of image intensity within the (local) support region. This in turn makes the face description robust to more local lighting changes, such as shadows cast by the nose.

In some cases there is a gross error in the face or feature detection process, e.g. one of the features is detected outside of the face. We flag such cases by putting limits on the affine rectifying transformation and do not use those as exemplars. It is also possible to use feature positions detected in nearby frames to correct isolated feature outliers, though this is not yet employed here.

5.3.3 Representation of face sets

The goal here is to compactly represent an entire face-track containing a set of (10 to 600) faces. Representing entire face-tracks brings a significant data reduction which is very advantageous in the immediate retrieval scenario, i.e. a query face(-track) needs to be compared only to a few hundred face-tracks in the entire movie (instead of tens of thousands of single face detections).

Each face is a point, $\mathbf{x}$, in the 360-dimensional descriptor space (section 5.3.2) and we assume that faces for a particular character have a certain probability density function $f(\mathbf{x})$ over this space. A face-track of that person than provides a set of samples from $f(\mathbf{x})$. We use a non-parametric model of $f(\mathbf{x})$ and represent each face-track as a histogram over precomputed (vector quantized) face-feature exemplars. A similar representation (over filter responses) has been used in representing texture by, for example, Leung and Malik [83] or Varma and Zisserman [164], and recently has also been applied by Leung [81] to face recognition. An alternative would be to use a mixture of Gaussians as proposed by Arandjelovic *et al.* [6].

In more detail, we assume that the appearance of each facial feature is independent of the appearance of other features and perform vector quantization for each facial feature separately. The

vector quantization is carried out by k-means clustering computed from about 30,000 faces from the movie 'Pretty woman'. The k-means algorithm is initialized using the Progressive Constructive Clustering algorithm. This is a greedy algorithm which starts with a random point as a cluster centre and then runs through the data once, assigning each point to the nearest cluster or generating a new cluster for points where no cluster is within a distance d. Note that the number of clusters is determined automatically by setting a threshold on distance d. The final number of face feature clusters is 537, 523, 402, 834 and 675 for the left eye, the middle of the eyes, the right eye, the mouth and the nose respectively. Random samples from facial feature clusters are shown in figure 5.6.

For each detected face each facial feature is assigned to the nearest cluster centre (e.g. the left eye is coded as one of 537 possibilities). The final representation of a face then is similar to a face identikit where the appearance is composed from the nearest cluster centre for eyes, nose, mouth etc. Each *set* of faces is thus represented by five histograms, p^j, $j = \{1, \ldots 5\}$, one for each facial feature. Each histogram represents a distribution over the cluster centres, where an element p_i^j of p^j is the frequency of occurrence of the ith vector quantized face feature cluster. Note that this representation ignores any image ordering or temporal information within the face-track. Each histogram is normalized to sum to one so that it is a probability distribution. Note also that these histograms represent marginal distributions for each facial feature. This is because features were assumed independent and thus quantized separately. Under this assumption, the joint probability of a particular combination of facial feature exemplars would be obtained by multiplying the corresponding marginal probabilities.

5.3.4 Matching face sets

Having derived the representation of a face-set, we now define a matching measure between two face-sets. This matching measure is applied for connecting face-sets broken due to occlusions within a shot and also for retrieval of face-sets across shots.

Distributions, p^j, introduced above, cover expression changes naturally, for example closed and open eyes, or neutral and smiling faces. It is here that we benefit from matching sets of faces: for example with the correct matching measure a shot containing a smiling person can match a shot containing the same person smiling and neutral.



Figure 5.6: **Quantized facial features.** Each row shows ten random samples from one cluster of a vector quantized facial feature (from top: left eye, right eye, the middle of the eyes, mouth, nose). Each sample is shown as an affinely transformed elliptical region superimposed on an image. The size of the features shown is 3/5 of the actual scale. Note, there is generalization over pose and illumination.

Similarity between two face-tracks, P and Q, each represented by five histograms, p^j and q^j respectively, is computed as

$$d(P,Q) = \frac{1}{5}\sum_{j=1}^{5}\chi^2(p^j, q^j), \tag{5.1}$$

where two corresponding histograms, p^j, q^j, are compared using the χ^2 statistic as

$$\chi^2(p^j, q^j) = \sum_{k=1}^{S_j}\frac{(p_k^j - q_k^j)^2}{(p_k^j + q_k^j)}, \tag{5.2}$$

where S_j is the number of histogram bins. $\chi^2(p,q)$ takes values between 0 and 2, being zero when $p = q$. This statistic is related to the symmetric KL divergence between two probability distributions.

In the implementation, each face-track is represented by a single histogram (with 2,971 bins) obtained by concatenating the individual facial feature histograms.

Matching face sets using the original face descriptors

The method for matching face sets described above is compared to a method where face sets are matched using the original descriptors of facial features. In particular, recall from section 5.3.2 that each face-track is represented as a set of 360-dimensional SIFT vectors, one vector for each face in the set. Let us have two sets of face descriptors, $A = \{\mathbf{a}_i\}$ and $B = \{\mathbf{b}_i\}$, where $\mathbf{a}_i, \mathbf{b}_i \in R^{360}$. The distance between two sets, A and B, is measured as

$$h(A, B) = \min_{\mathbf{a}_i \in A} \min_{\mathbf{b}_i \in B} d(\mathbf{a}_i, \mathbf{b}_i) \tag{5.3}$$

where $d(\mathbf{a}_i, \mathbf{b}_i)$ is Euclidean distance. Note that this measure of dissimilarity is quite different from comparing distributions as proposed above. Here the distance between two sets is the smallest distance between any two elements in the sets. As a result, one pair of very close points can make the distance between otherwise dissimilar sets very small. Moreover, this method is rather expensive. To evaluate equation 5.3 mutual distances between all descriptors in A and B need to be computed and this requires $O(MND)$ computations, where M and N are cardinalities of A and B respectively, and D is the dimensionality of the descriptor space (here 360).

Application: Matching sets of faces within a shot

The face-tracks developed in section 5.2 can be broken due to e.g. occlusion by another person or object, or self-occlusion when the actor turns away from the camera. The goal here is to connect such face-tracks. This is beneficial as it gives larger and more representative sets of faces. It is also an easier task than inter-shot matching as the imaging conditions usually do not change dramatically within a shot. The intra-shot matching is achieved by grouping face-tracks with similar distributions, where the distance between distributions is measured by χ^2 as in (5.1). The grouping is again carried out by the single link clustering algorithm used in section 5.2. Note that the temporal exclusion constraint is used here again. An example of connecting several face-tracks within a shot is shown in figure 5.4. The intra-shot matching performance on ground truth data is given in section 5.5.

5.4 Retrieving sets of faces across shots – Video Google Faces

We have built a person retrieval system for two feature length movies, 'Pretty Woman' and 'Groundhog Day'.

At run time the user outlines a face in a frame of the video, and the outlined region tracks are used to 'jump' onto the closest face-track – a set of face detections. In more detail, the tracks on affine covariant regions propagate detected faces to frames where face detections are missing. The result is a more complete representation of the video, where the user can outline the face of a person in any frame along the face-track, even in frames where the face was not detected due to for example strong shadow or non-frontal pose. Note that face descriptors for matching face-tracks are extracted only from detected faces.

Once the query face-track is determined, the face-tracks within the movie are ranked according to the χ^2 distance to the query face-track. The face retrieval algorithm is summarized in figure 5.7. A screenshot of the running face retrieval system is shown in figure 5.8. In the next section we evaluate retrieval performance on ground truth data and show several retrieval examples.

5.5 Experimental evaluation

We show quantitative results on manually labelled ground truth data from the movie 'Pretty Woman'. Qualitative retrieval results are shown on the movie 'Groundhog Day'. Finally, we show an example of retrieval across movies, where images of an actor in one movie are used to find occurrences of the same actor in another movie.

5.5.1 Evaluation on ground truth

The performance of the proposed method is assessed on 337 shots from 'Pretty Woman'. Ground truth on the identity of the detected faces for the seven main characters of the movie is obtained manually for these shots. The entire movie has 1151 shots and 172,105 frames. The 337 ground truth shots contain 38,846 face detections of which 31,846 have successful facial feature detection. The temporal grouping algorithm of section 5.2 groups these into 776 face tracks of which 431 have more than 10 face detections.

1. **Pre-processing (off-line)**

   - Detect faces in each frame of the video. Find facial features (eyes, nose and mouth) within each face detection (section 5.3.1). Extract descriptor for each face (SIFTs placed at facial features, section 5.3.2).
   - Build face-specific visual vocabulary by clustering facial feature descriptors (section 5.3.3). Assign facial features on each face to the nearest visual word.
   - Associate faces into face-tracks by tracking affine covariant regions on faces (section 5.2).
   - Represent face-tracks by histograms of face-specific visual words (section 5.3.3).
   - Connect face sets broken due to e.g. occlusion by matching within each shot (section 5.3.4).

2. **At run-time (given a user selected query region)**

   - Determine the query face-track from the user selected query region.
   - Rank face-tracks in the video according to χ^2 distance (eq. 5.1) from the query face-track (section 5.3.4).

Figure 5.7: The face retrieval algorithm.

The main parameters of the overall system are the face detection threshold (which controls the number of false positives and negatives); the size of the support regions for the SIFT descriptors; the distance threshold on SIFT responses determining the number of cluster centres for each face feature region; and the threshold on the χ^2 distance used in face-track intra-shot matching.

Intra-shot matching: The intra-shot matching algorithm is applied to the 66 (out of the 337 ground truth) shots that contain more than two face-tracks. The 143 original face-tracks from these shots are grouped into 90 face-tracks. The precision is 98.1% (1.9% incorrect merges, i.e. one incorrect merge) and recall is 90.7%, i.e. 9.3% possible merges were missed. Examples of several successful intra-shot matches on the 'opera' shot are shown in figure 5.4.

Inter-shot matching: Inter-shot matching is evaluated on a ground truth set of 269 face-tracks of only the seven main characters (after intra-shot matching). The 269 face-tracks contain 25,366 face detections. Example retrievals are shown in figures 5.9 and 5.10. In some cases the precision-

(a)



(b)



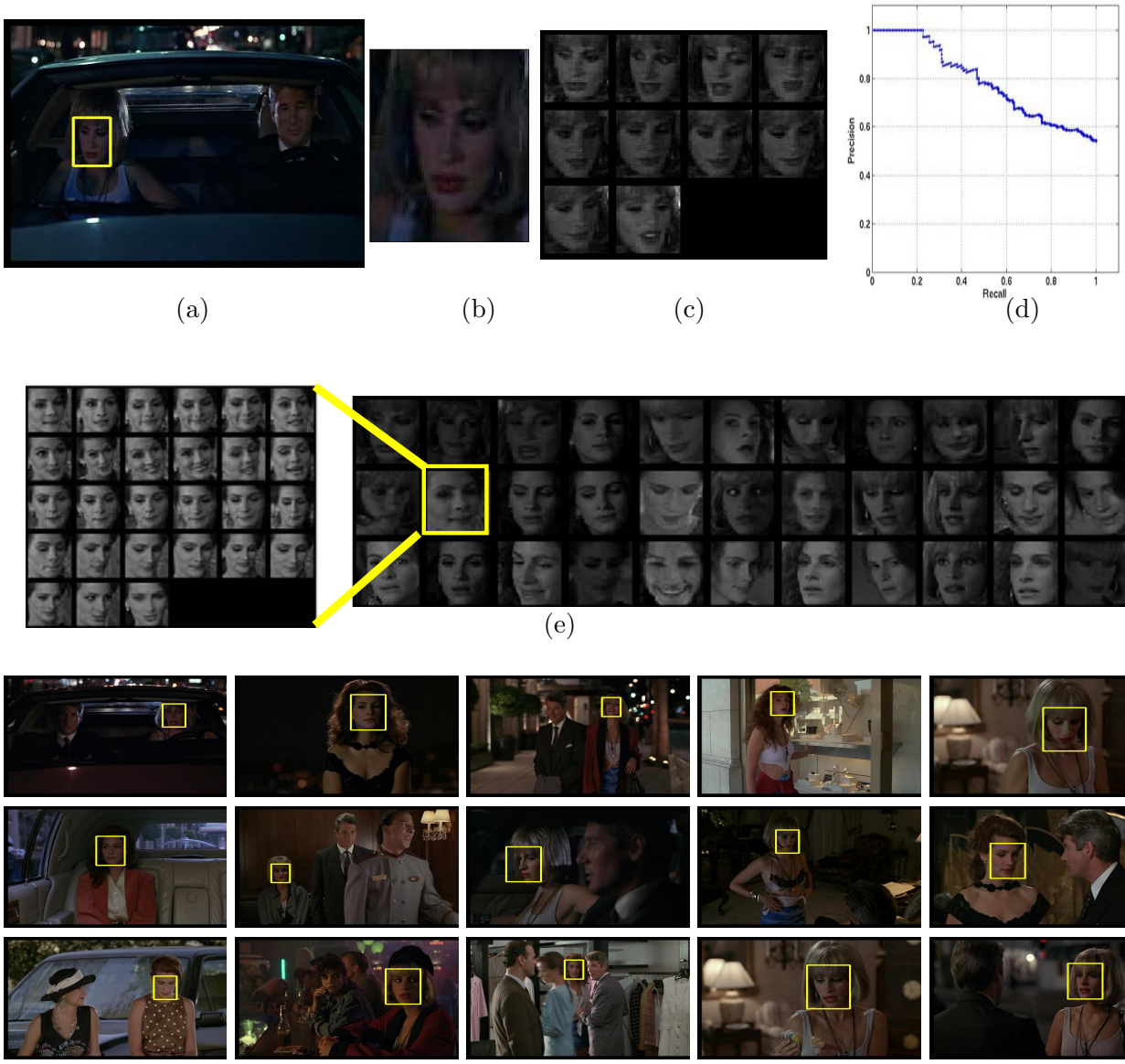
Figure 5.8: **A face query example on 'Groundhog Day':** (a) A frame with user specified query region. (b) A screenshot of the retrieval system showing the first ten ranked face-tracks. Each face-track is displayed by three thumbnails showing (from left to right) the first frame of the corresponding shot, the first face detection in the face-track, shown in white, and the last frame of the shot.

recall curve does not reach 100% recall. This is because face-tracks with non-overlapping histograms ($\chi^2(p,q) = 2$) are not shown.

Retrieval performance evaluated over all queries is shown in table 5.1. The evaluation criteria is precision at 20% recall. This criteria puts emphasis on lower recall levels. Four methods are evaluated: (a) the proposed approach where face-track distributions are ranked based on χ^2 distance (eq. 5.1) from the query face-track distributions; (b) face-tracks are represented by all extracted descriptors and the face-tracks are ranked using the *'minmin'* distance given by equation 5.3; (c) each face-track is represented by a single face descriptor chosen at random; (d) each face-track is represented by the mean of all descriptors over the entire face-track. Method (c) is run five times with different random descriptors chosen for each track and an average result is shown. In (c) and (d) face-tracks are ranked using Euclidean distance between the single 360-dimensional descriptors representing each track. Retrieval performance is compared to chance (e), which is the prior probability of each character, i.e. the proportion of face-tracks of each character in the data.

The 'χ^2' method (a) and the 'minmin' method (b) have on average similar performance, but method (a) is almost 70 times faster. This is because method (a) represents each face-track by a single histogram, whereas in (b) all descriptors for each face-track are considered. Methods (c) and (d) on average perform worse than (a) and (b). This might be attributed to rather impoverished face-track representation: in (c) only a single face descriptor describes the entire face-track; in (d) all descriptors in a face-track are averaged. In terms of speed, the single descriptor method (c) is still about ten times slower than the χ^2 method (a). This is due to the sparsity of the face-track distributions.

All methods exhibit a surprisingly high variance. In other words, the retrieval performance strongly depends on the particular query face-track. This is due to several reasons including: (i) failures in facial feature detection, which corrupt the face descriptor; (ii) strong shadows, or a significantly non-frontal pose may result in retrieving face-tracks with similar lighting or pose irrespective of the identity; (iii) an unusual hair-style or a hat may also negatively affect results. This is because descriptor support regions are rather large and capture also the face boundary. On average however, capturing the face boundary is beneficial. Decreasing the size of the descriptor support lowers the average retrieval performance.



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

(e)

(f)

Figure 5.9: Example retrieval of the main character from the movie 'Pretty woman'. (a) The query frame with the query face outlined in yellow. (b) close-up of the face. (c) The associated set of 10 face detections in this shot. (d) Precision-Recall curve. (e) Right: the first 33 retrieved face sets shown by the first face detection in each set. Left: example of a retrieved face set. (f) Example face detections from the first 15 retrieved face sets superimposed on the original frames. Note the extent of pose, lighting and expression variation among the retrieved faces. For this character, the number of relevant face-tracks in the ground truth set is 145.

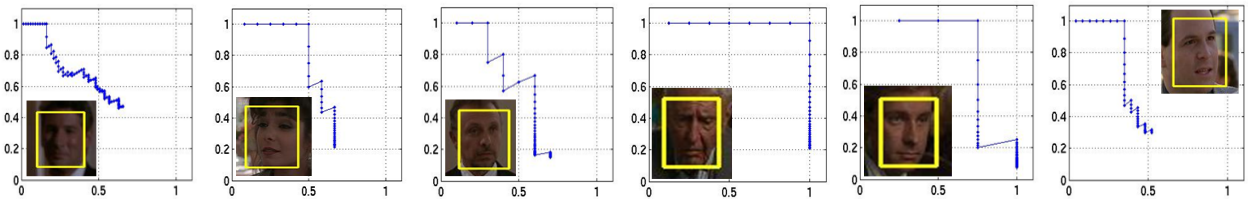


Figure 5.10: Retrieval examples of the other six main characters from the movie 'Pretty woman'. The graphs show precision (y-axis) vs. recall (x-axis). Thumbnails show close-ups of the first face detection from each query set. The number of relevant face-tracks in the ground truth set is 67, 12, 10, 8, 4 and 14 for each character respectively (from left).

	 1	 2	 3	 4	 5	 6	 7	Mean.
(a) χ^2	0.78$\pm$0.12	0.65$\pm$0.21	0.69$\pm$0.41	0.59$\pm$0.44	0.65$\pm$0.33	1.00$\pm$0.00	1.00$\pm$0.00	**0.77**
(b) MinMin	0.88$\pm$0.13	0.65$\pm$0.23	0.64$\pm$0.41	0.55$\pm$0.47	0.65$\pm$0.32	1.00$\pm$0.00	1.00$\pm$0.00	**0.77**
(c) Min1	0.82$\pm$0.14	0.60$\pm$0.22	0.51$\pm$0.39	0.55$\pm$0.44	0.63$\pm$0.31	1.00$\pm$0.00	1.00$\pm$0.00	**0.73**
(d) Mean	0.82$\pm$0.13	0.65$\pm$0.22	0.59$\pm$0.39	0.56$\pm$0.46	0.55$\pm$0.35	1.00$\pm$0.00	1.00$\pm$0.00	**0.74**
(e) Chance	0.52	0.26	0.04	0.06	0.08	0.02	0.02	—

Method	Average query time [s]
(a) χ^2	0.07
(b) MinMin	4.80
(d) Min1	0.72

Table 5.1: Ground truth evaluation on 269 face-tracks of the seven main characters in the movie 'Pretty Woman'. The top table shows precision at 20% recall for the seven main characters and four different methods. For each person results are averaged over all face-tracks (of that person) used as a query and mean and standard deviation are shown. The last column shows results averaged over all characters. See text for explanation of methods (a)-(d). The last row (e) represents chance. The bottom table shows query time averaged over all queries for methods (a), (b) and (d).

5.5.2 Results on another movie

We have also applied the person retrieval system to the movie 'Groundhog Day'. The movie contains 141,146 frames and 752 shots. 36,572 detected faces are grouped into 596 face tracks with more than 10 face detections. Facial feature descriptors are clustered into 671, 630, 502, 918 and 654 visual words for the left eye, the middle of the eyes, the right eye, the mouth and the nose respectively. An example search for one of the main characters is shown in figure 5.11. An example search for an auxiliary character is shown in figure 5.12. Finally, figure 5.13 shows an example of retrieval across movies. A shot of the actor 'Bill Murray' from the movie 'Lost in translation' (2003) was used to



Figure 5.11: Searching for one of the main characters in the movie 'Groundhog Day'. The top row shows the query frame with the query face outlined in yellow. The next five rows show the first 25 retrieved face-tracks (from left to right and top to bottom). Each face-track is shown by a single frame. The first false positive is ranked 30th.

query the movie 'Groundhog Day' (1993). Note that there is a 10 year difference between the two movies. In this example the 'minmin' distance given by equation 5.3 was used. Retrieval using 'minmin' distance takes about 10 seconds. If χ^2 distance is used, the assignment of visual words (performed using exhaustive search) takes about 0.13 seconds and computing the χ^2 distance takes another 0.15 seconds. The cost of the 35-fold speed-up against the 'minmin' approach is a slight loss in performance. The first false positive appears 16th (instead of 42nd using 'minmin') and in total there are 17 (instead of 15 using 'minmin') false positives in the first 100 retrieved face-tracks. All timings are on a 2GHz Pentium.



Figure 5.12: Searching for one of the auxiliary characters in the movie 'Groundhog Day'. Note that searching for auxiliary characters is challenging since they appear much less often in the movie. (a) The query frame with the query face outlined in yellow. (b) Query face close-up. (c) The first 12 retrieved face-tracks (from left to right and top to bottom). Each face-track is shown by a single frame (top) with matched face detection outlined in yellow and close-up of the face (bottom). There is only one false positive (ranked 9th and labelled by X) in the first 12 retrieved face-tracks.

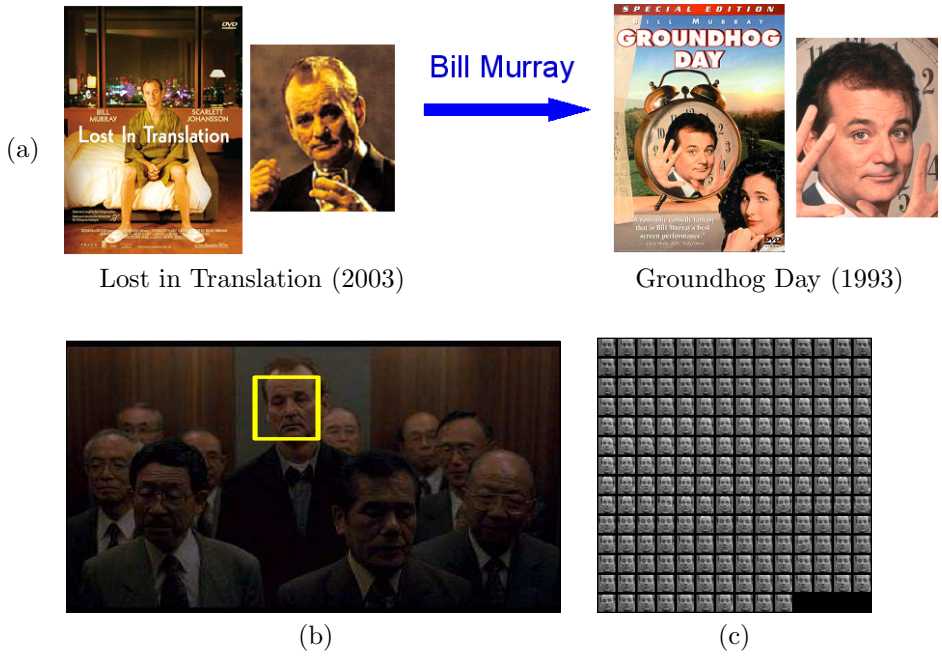


Figure 5.13: **Matching faces across movies.** (a) The goal is to search for Bill Murray in the movie 'Groundhog Day' using a face-track from the movie 'Lost in Translation'. (b) The query frame with the query face outlined in yellow. (c) There are 192 face detections associated in the query face-track. (d) The first 30 face-tracks retrieved from 'Groundhog Day' (from left to right and top to bottom). Each face-track is shown by a single frame. The first false positive is ranked 42nd and there are 15 false positives in the first 100 retrieved faces.

5.6 Conclusions and discussion

We have developed a representation for sets of faces which has the dual advantage that it is distinctive (in terms of inter-person vs. intra-person matching), and also is in a vector form suitable for efficient matching using nearest neighbour or inverted file methods. Using this representation for sets of faces of each person in a shot reduces the matching problem from $O(10^4)$ face detections over the entire movie, to that of matching a few hundred probability distributions. This enables immediate retrieval at run time.

As discussed in section 5.5.1, several factors such as strong shadows or significantly non-frontal poses may still negatively affect the face matching performance. Moreover, generalization coming from a single face track depends on the amount of pose/expression variations present within the face track. One possible path to overcome these difficulties would be to facilitate generalization across face tracks and individuals by, for example, off-line learning of an appropriate similarity metric between faces from labelled training data [23].

This work may be improved in several ways, for example: (i) extending the intra-shot matching to clustering over the entire movie (with constraints provided by the exclusion principle); (ii) using the exclusion principle to provide negative exemplars for retrieval at run time; (iii) currently we explored person retrieval using (near) frontal faces, but other attributes such as hair or clothing could be added to the feature vector [38].

Finally, the idea of developing a class specific visual vocabulary (here built for faces) and associating exemplars available in video to obtain additional generalization (here over facial expressions) can be applied for retrieval of instances of other object classes, provided object detectors are available. Imagine, for example, deforming clothes on people, cars viewed from different visual aspects, or deforming skin on walking animals.

Chapter 6

Discovering object classes in image collections

6.1 Introduction

Previous chapters have been concerned with retrieval of object instances and faces of a particular person. In this chapter, we focus on retrieval of object classes. In more detail, we would like to retrieve all occurrences of an object class (e.g. 'airplanes'), given a single query example of that class (i.e. a particular example of an airplane). As will be shown later in this chapter, the retrieval approach taken in previous chapters achieves only limited performance, mainly due to the intra-class variations present in the data. The approach we take here instead is first to learn (or discover) object class models from large amounts of unlabelled 'training' data, and then to use those models for object class retrieval. In particular, we apply the probabilistic semantic analysis model (pLSA) [64, 65] from the statistical text analysis literature, reviewed in section 2.6. As in previous chapters, documents are images, and we quantize local appearance descriptions to form visual 'words'. As will be explained in more detail in section 6.3, here, the quantization is 'coarser', as we wish to capture some amount of intra-class variation.

Unlike previous chapters, we do not use the spatial layout of visual words to improve the image ranking. This is essentially the 'bag of words' model, reviewed in section 2.2.2. While losing spatial information between visual features looks like a rather impoverished image representation, there are

several reasons for optimism: (i) as opposed to old corner detectors, modern feature descriptors have become powerful enough to encode very complex visual stimuli, making them quite discriminative; (ii) natural images are also very redundant (i.e. given a bag of features from an image, it is highly unlikely to find another natural image with the same features); (iii) because features are allowed to overlap in the image, some spatial information is implicitly preserved (i.e. randomly shuffling bits of the image around will almost certainly change the bag of words description). So, while these spatial relationships must eventually be taken into account, here we are investigating how far the bag of words model can be pushed in the image domain for retrieval of visual object classes.

Chapter outline: Section 6.2 introduces the statistical model, and section 6.3 describes the image representation used. To explain and assess performance, in section 6.4 we apply the pLSA model to sets of images for which the ground truth labelling is known. In section 6.5, we discuss in more detail what is being learnt by the pLSA model, and give some insights into how the pLSA model learning works. Finally, in section 6.6, we apply the learnt models to the task of object class retrieval in movies and image databases.

6.2 The topic discovery model

We will describe the model here using the original terms 'documents' and 'words' as used in the text literature. Our visual application of these (as images and visual words) is then given in the following sections.

Suppose we have N documents containing words from a vocabulary of size M. The corpus of text documents is summarized in a M by N co-occurrence table $\mathtt{N}$, where $n(w_i, d_j)$ stores the number of occurrences of a word w_i in document d_j. This is the bag of words model. In addition, there is a hidden (latent) topic variable z_k associated with each occurrence of a word w_i in a document d_j.

The joint probability $P(w_i, d_j, z_k)$ is assumed to have the form of the graphical model shown in figure 6.1(a). Marginalizing over topics z_k and dividing by $P(d_j)$ determines the conditional probability $P(w_i|d_j)$:

$$P(w_i|d_j) = \sum_{k=1}^{K} P(z_k|d_j)P(w_i|z_k), \tag{6.1}$$

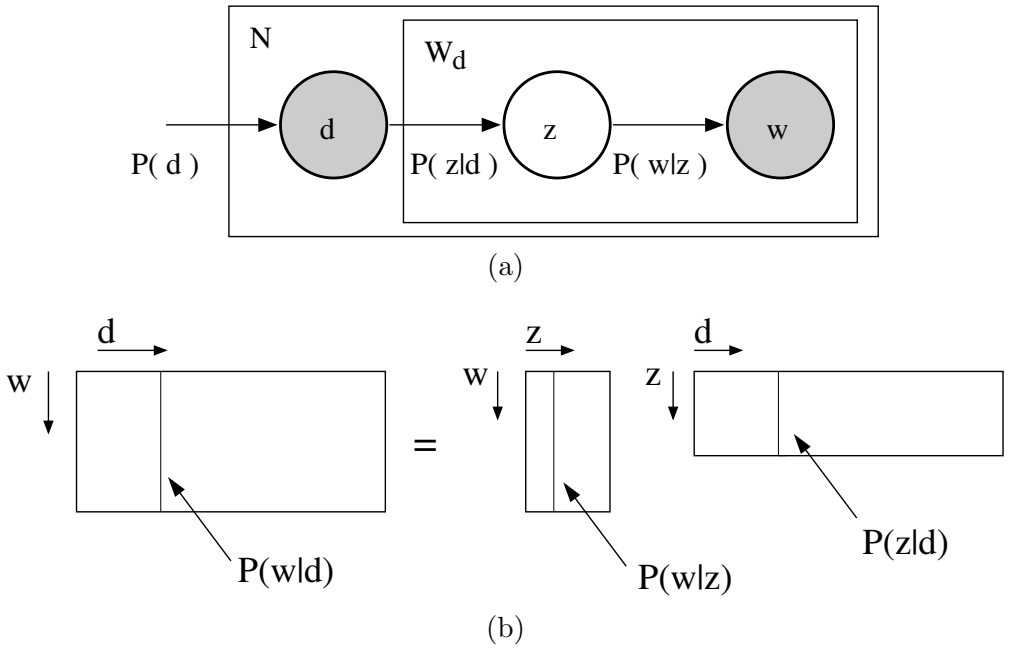


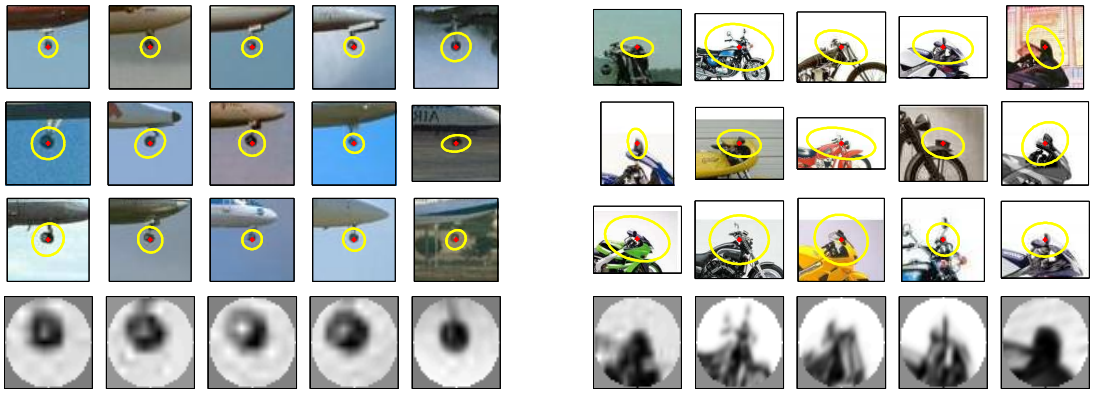
Figure 6.1: (a) pLSA graphical model, see text. Nodes inside a given box (plate notation) indicate that they are replicated the number of times indicated in the top left corner. Filled circles indicate observed random variables; unfilled are unobserved. (b) In pLSA the goal is to find the topic specific word distributions $P(w|z_k)$ and corresponding document specific mixing proportions $P(z|d_j)$ which make up the document specific word distribution $P(w|d_j)$.

where $P(z_k|d_j)$ is the probability of topic z_k occurring in document d_j; and $P(w_i|z_k)$ is the probability of word w_i occurring in a particular topic z_k.

The model (6.1) expresses each document as a convex combination of K topic vectors. This amounts to a matrix decomposition as shown in figure 6.1(b) with the constraint that both the vectors and mixture coefficients are normalized to make them probability distributions. Essentially, each document is modelled as a mixture of topics – the histogram for a particular document being composed from a mixture of the histograms corresponding to each topic.

Fitting the model involves determining the topic vectors which are common to all documents and the mixture coefficients which are specific for each document. The goal is to determine the model that gives high probability to the words that appear in the corpus, and a maximum likelihood estimation of the parameters is obtained by maximizing the objective function:

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N} P(w_i|d_j)^{n(w_i,d_j)}, \tag{6.2}$$



(a) (b)

Figure 6.2: Two examples of visual words. (a) A wheel of an airplane. (b) A motorbike handle. In each case, the top three rows show 15 occurrences of this visual word in different images with the elliptical region superimposed. The bottom row shows affine normalized regions for the top row of images. Note that the normalized regions appear quite similar – which is why they are grouped in the same cluster. In the original images, the elliptical regions exhibit intra-class variation, and varying scale (the scaling is removed in this display as the ellipses are size normalized for visibility).

where $P(w_i|d_j)$ is given by (6.1).

As shown in appendix A.1, this is equivalent to minimizing the Kullback-Leibler divergence between the measured empirical distribution $\tilde{P}(w|d)$ and the fitted model. The model is fitted using the Expectation Maximization (EM) algorithm [65]. In the E-step, posterior probability over topics, $P(z_k|w_i, d_j)$, is computed for each word w_i and document d_j as

$$P(z_k|w_i, d_j) = \frac{P(w_i|z_k)P(z_k|d_j)}{\sum_{l=1}^{K} P(w_i|z_l)P(z_l|d_j)}. \tag{6.3}$$

M-step updates distributions $P(w|z)$ and mixing weights $P(z|d)$ using the current estimate of the posterior given by eq. 6.3:

$$P(w_i|z_k) \propto \sum_{j=1}^{N} n(w_i, d_j)P(z_k|w_i, d_j), \tag{6.4}$$

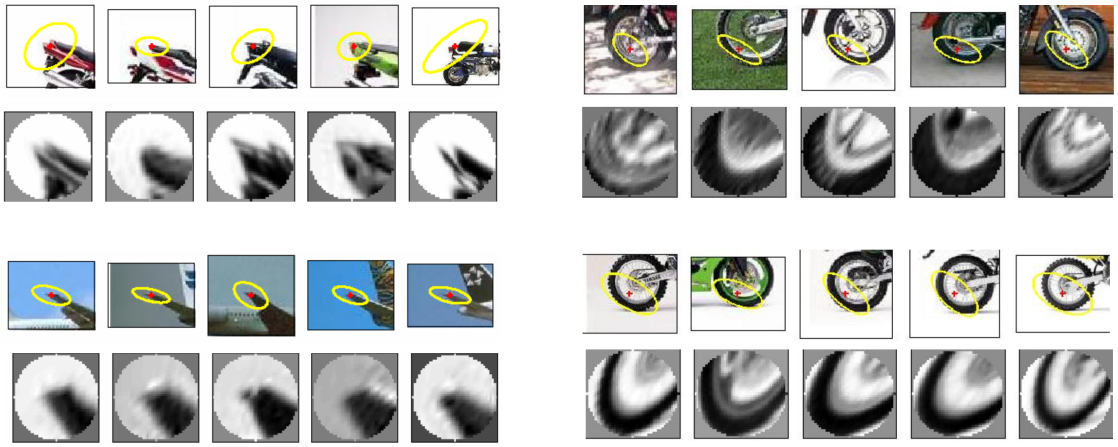
$$P(z_k|d_j) \propto \sum_{i=1}^{M} n(w_i, d_j)P(z_k|w_i, d_j). \tag{6.5}$$

6.3 Obtaining visual words

We seek a vocabulary of visual words which will be insensitive to changes in viewpoint and illumination. To achieve this, similarly to chapters 3 and 4, we use vector quantized SIFT descriptors [85] computed on affine covariant regions [91, 93, 121]. Affine covariance gives tolerance to some amount of viewpoint changes; SIFT descriptors, based on histograms of local orientation, gives some tolerance to illumination change. Vector quantizing these descriptors gives tolerance to appearance variations within an object category. In contrast to descriptor quantization in chapter 3, which was targeted to retrieval of particular objects, in this chapter we use a 'coarser' quantization to only about 1,000 visual words, which should give us tolerance to some amount of within class appearance variations. Others have used similar vector quantized descriptors for object classification [29, 104], but in a supervised setting.

As described in section 3.2, two types of affine covariant regions are computed for each image. Each region is described by a SIFT descriptor. In contrast to chapter 3, there is no rotation of the patch, i.e. the descriptors are rotation variant. Alternatively, the SIFT descriptor could be computed relative to the the dominant gradient orientation within a patch, making the descriptor rotation invariant [85]. The SIFT descriptors are vector quantized into the visual 'words' for the vocabulary. The vector quantization is carried out here by k-means clustering computed from about 300K regions. The regions are those extracted from a random subset (about one third of each category) of images of airplanes, cars, faces, motorbikes and backgrounds. About 1K clusters are used for each of the Shape Adapted and Maximally Stable regions, and the resulting total vocabulary has 2,224 words. The number of clusters, k, is clearly an important parameter. The intention is to choose k to determine words which give some intra-class generalization. As in chapter 5, the k-means clustering was initialized using the Progressive Constructive Clustering algorithm. This is a greedy algorithm which starts with a random point as a cluster center and then runs through the data once, assigning each point to the nearest cluster or generating new cluster for points where no cluster is within distance d. Note that number of clusters is determined automatically by setting the threshold on distance d. This vocabulary is used for all the experiments throughout this chapter. Examples of visual words are shown in figure 6.2.

It is also worth noting that visual words can be ambiguous. In text analysis, a word with two



(a) (b)

Figure 6.3: (a) **Visual polysemy.** 10 examples from one visual word occurring on different (but locally similar) parts on different object categories (motorbikes and airplanes). (b) **Visual synonyms.** Five examples from two different visual words representing a similar part of an object (wheel of a motorbike). Each example occurrence is shown by the elliptical region overlaid over the original image (top) and the normalized region (bottom).

(or more) different meanings is called polysemous (e.g. 'bank' as in (i) a money keeping institution, or (ii) a river side). As shown in figure 6.3(a), we observe the analogue of polysemy in our visual words, however, the topic discovery models can (to some extent) cope with these. A polysemous word would have a high probability in two different topics. The hidden topic variable associated with each word occurrence in a particular document can assign such a word to a particular topic depending on the context of the document. We return to this point in section 6.4.3.

As shown in figure 6.3(b), we also observe visual analogue to synonyms. Synonyms are two or more words that have the same meaning (e.g. 'student' and 'pupil').

6.4 Experiments

Given a collection of unlabelled images, our goal is to automatically discover/classify the visual categories present in the data and localize them in the image. To understand how the algorithms perform, we train on image collections for which we know the desired visual topics.

We investigate three areas: (i) topic discovery – where categories are discovered by pLSA clustering on all available images, (ii) classification of unseen images – where topics corresponding to object categories are learnt on one set of images, and then used to determine the object categories

present in another set, and (iii) object detection – where we also wish to determine the location and approximate segmentation of object(s) in each image.

We use two datasets of objects, one from Caltech [41, 45] and the other from MIT [153]. The Caltech datasets depict one object per image. The MIT dataset depicts multiple object classes per image. We report results for the three areas first on the Caltech images, and then in section 6.4.4 show their application to the MIT images.

Caltech image data sets. Our data set consists of images of five categories from the Caltech 101 datasets (as previously used by Fergus *et al.* [45] and Fei-Fei *et al.* [41] for supervised classification). The categories and their number of images are: faces, 435; motorbikes, 800; airplanes, 800; cars rear, 1155; background, 900. The reason for picking these particular categories is pragmatic: they are the ones with the greatest number of images per category. All images have been converted to grayscale before processing. Otherwise they have not been altered in any way, with one notable exception: a large number of images in the motorbike category (2) and airplane category (3) have a white border around the image which we have removed since it was providing an artifactual cue for object class.

Baseline method – k-means (KM): To understand the contributions of the topic discovery model to the system performance, we also implemented an algorithm using the same features of word frequency vectors for each image, but without the final statistical machinery. The standard k-means procedure is used to determine k clusters from the word frequency vectors, i.e. the pLSA clustering on KL divergence is replaced by Euclidean distance and each document is hard-assigned to exactly one cluster. For each experiment, the k-means algorithm is run five times with different random initializations and the lowest distortion result is taken.

Model learning: The pLSA EM algorithm is initialized randomly and typically converges in 40–100 iterations. One iteration takes about 2.3 seconds on 4K images with 7 fitted topics and ∼300 non-zero word counts per image (Matlab implementation on a 2GHz PC). For each experiment we run the pLSA EM algorithm five times with random initializations and take the result with highest likelihood.

Ex	# of images	Categories	K	pLSA		KM baseline	
				%	#	%	#
(1)	3,190	4	4	98	70	72	908
(2)	4,190	4 + bg	5	78	931	54	1976
(2)*	4,190	4 + bg	6	76	1072	59	1737
(2)*	4,190	4 + bg	7	83	768	54	1678
(2)*	3,690 + 400	4 + bg-fxd	7	93	238	–	–

Table 6.1: Summary of the experiments. Column '%' shows the classification accuracy measured by the average of the diagonal of the confusion matrix. Column '#' shows the total number of misclassifications. See text for a more detailed description of the experimental results. In the case of (2)* the two/three background topics are allocated to one category. Evidently the baseline method performs poorly, showing the power of the pLSA clustering.

6.4.1 Topic discovery

In each experiment images are pooled from a number of original datasets, and the pLSA and baseline models are fitted to the ensemble of images (with no knowledge of the image's labels) for a specified number of topics, K. For example, in experiment (1) the images are pooled from four categories (airplanes, cars, faces and motorbikes) and models with $K = 4$ objects (topics) are fitted. In the case of pLSA, the model determines the mixture coefficients $P(z_k|d_j)$ for each image (document) d_j (where $z \in \{z_1, z_2, z_3, z_4\}$ for the four topics). An image d_j is then classified as containing object k according to the maximum of $P(z_k|d_j)$ over k. This is essentially a one against many (the other categories) test. Since here we know the object instances in each image, we use this information as a performance measure. A confusion matrix is then computed for each experiment.

(1) Images of four object categories with cluttered backgrounds. The four Caltech categories have cluttered backgrounds and significant scale variations (in the case of cars rear). An interesting observation comes from varying the number of topics, K. In the case of $K = 4$, we discover the four different categories in the dataset with very high accuracy (see table 6.1). In the case of $K = 5$, the car dataset splits into two subtopics. This is because the data contains sets of many repeated images of the same car. Increasing K to 6 splits the motorbike data into sets with a plain background and cluttered background. Increasing K further to 7 and 8 'discovers' two more sub-groups of car data containing again other repeated images of the same/similar cars.

It is also interesting to see the visual words which are most probable for an object, by selecting those with high topic specific probability $P(w_i|z_k)$. These are shown for the pLSA model for the



Figure 6.4: The most likely words (shown by 5 examples in a row) for four learnt topics in experiment (1): (a) Faces, (b) Motorbikes, (c) Airplanes, (d) Cars.

case of $K = 4$ in figure 6.4.

Thus, for these four object categories, topic discovery analysis cleanly separates the images into object classes, with reasonable behavior as the number of topics increases beyond the number of objects. The most likely words for a topic appear to be semantically meaningful regions.

(2) Images of four object categories plus "background" category. Here we add images of an explicit "background" category (indoor and outdoor scenes around Caltech campus) to the above experiment (1). The reason for adding these additional images is to give the methods the opportunity of discovering background "objects".

The confusion tables as K is varied are shown as images in figure 6.5. It is evident, for example, that for pLSA the first topic confuses faces and backgrounds to some extent. The results are summarized in table 6.1. The case of $K = 7$ with three topics allocated to the background gives the best performance. Examples of the most likely visual words for the three discovered background topics are shown in figure 6.6.

In the case of many of the Caltech images there is a strong correlation of the foreground and backgrounds (e.g. the faces are generally against an office background). This means that in the absence of other information the learnt topic (for faces for example) also includes words for the background. In classification, then, some background images are erroneously classified as

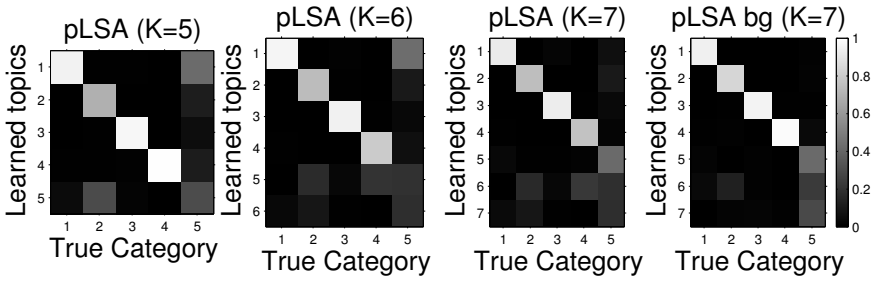


Figure 6.5: Confusion tables for experiment (2) for increasing number of topics (K=5,6,7) and pLSA with 7 topics and fixed background respectively. Brightness indicates number. The ideal is bright down the diagonal for the first four classes. Note how the background (category 5) splits into 2 and 3 topics (for K=6 and 7 respectively) and that some amount of the confusion between categories and background is removed.

True Class →	Faces	Moto	Airp	Cars	Backg
Topic 1 - Faces	94.02	0.00	0.38	0.00	1.00
Topic 2 - Motorb	0.00	83.62	0.12	0.00	1.25
Topic 3 - Airplan	0.00	0.50	95.25	0.52	0.50
Topic 4 - Cars rear	0.46	0.88	0.38	98.10	3.75
Topic 5 - Bg I	1.84	0.38	0.88	0.26	41.75
Topic 6 - Bg II	3.68	12.88	0.88	0.00	23.00
Topic 7 - Bg III	0.00	1.75	2.12	1.13	28.75

Table 6.2: Confusion table for experiment (2) with three background topics fixed. The mean of the diagonal (counting the three background topics as one) is 92.9%. The total number of miss-classified images is 238. The discovered topics correspond well to object classes.

faces. If the background distributions were to be fixed, then when determining the new topics the foreground/backgrounds are decorrelated because the backgrounds are entirely explained by the fixed topics, and the foreground words would then need to be explained by a new topic.

Motivated by the above, we now carry out a variation in the learning where we first learn three topics on a separate set of 400 background images alone. This background set is disjoint from the one used in experiment (2). These topics are then frozen, and a pLSA decomposition with seven topics (four to be learnt, three fixed) are again determined for experiment (2). The confusion table classification results are given in table 6.2. It is evident that the performance is improved over not fixing the background topics above.

Discussion: In the experiments it was necessary to specify the number of topics K, however Bayesian [147] or minimum complexity methods [10] can be used to infer the number of topics implied by a corpus. It should be noted that the baseline k-means method achieves nowhere near

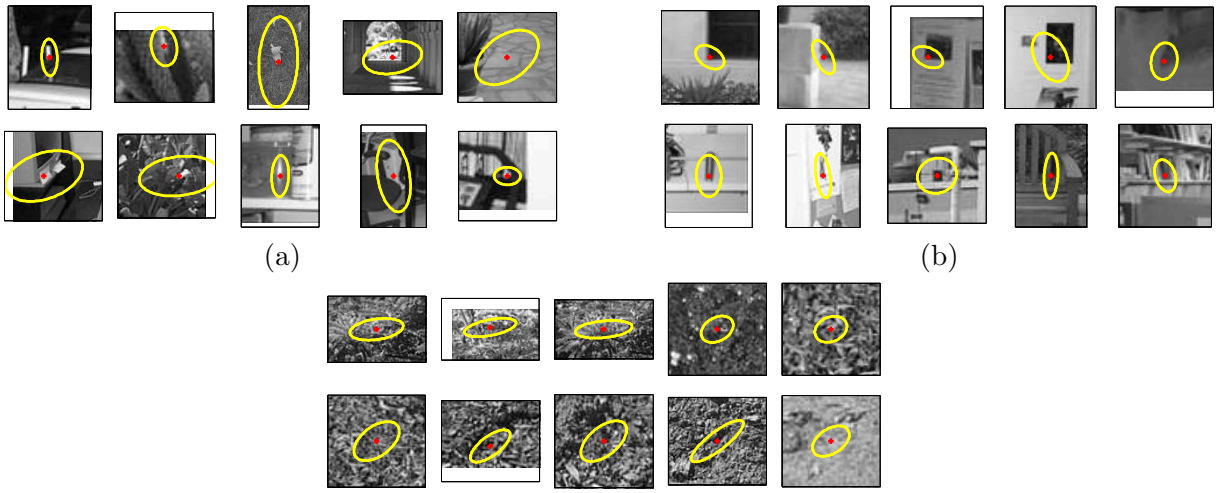


Figure 6.6: The most likely words (shown by 5 examples in a row) for the three background topics learned in experiment (2): (a) Background I, mainly local feature-like structure, (b) Background II, mainly corners and edges coming from the office/building scenes, (c) Background III, mainly textured regions like grass and trees. For topic numbers refer to figure 6.7(c).

the level of performance of the pLSA method. This demonstrates the power of using the proper statistical modelling here.

6.4.2 Classifying new images

The learned topics can also be used for classifying new images, a task similar to the one in Fergus *et al.* [45]. In the case of pLSA, the topic specific distributions $P(w|z)$ are learned from a separate set of 'training' images. When observing a new *unseen* 'test' image, the document specific mixing coefficients $P(z|d_{test})$ are computed using the 'fold-in' heuristic described in [64]. In particular, the unseen image is 'projected' on the simplex spanned by learned $P(w|z)$, i.e. the mixing coefficients $P(z_k|d_{test})$ are sought such that the Kullback-Leibler divergence between the measured empirical distribution $\tilde{P}(w|d_{test})$ and $P(w|d_{test}) = \sum_{k=1}^{K} P(z_k|d_{test})P(w|z_k)$ is minimized. This is achieved by running the EM algorithm in a similar manner to that used in learning, but now only the coefficients $P(z_k|d_{test})$ are updated in each M-step. The learned $P(w|z)$ are kept fixed.

(3) Multiway classification of unseen images. To compare performance with Fergus *et al.* [45], experiment (2) was modified such that only the 'training' subsets for each category (and all background images) from [45] were used to fit the pLSA model with 7 topics (four object topics and

True Class →	Faces	Motorb	Airplan	Cars rear
Topic 1 - Faces	99.54	0.25	1.75	0.75
Topic 2 - Motorb	0.00	96.50	0.25	0.00
Topic 3 - Airplan	0.00	1.50	97.50	0.00
Topic 4 - Cars rear	0.46	1.75	0.50	99.25

Table 6.3: Confusion table for unseen test images in experiment (3) – classification against images containing four object categories, but no background images. Note there is very little confusion between different categories. See text.

Object categ.	pLSA (a)	pLSA (b)	Fergus *et al.* [45]
Faces	5.3	3.3	3.6
Motorbikes	15.4	8.0	6.7
Airplanes	3.4	1.6	7.0
Cars rear*	21.4 / 11.9	16.7 / 7.0	9.7

Table 6.4: Equal error rates for image classification for pLSA and the method of Fergus *et al.* [45]. Test images of a particular category were classified against (a) testing background images (test performed in [45]) and (b) testing background images *and* testing images of all other categories. The improved performance in (b) is because our method exhibits very little confusion between different categories. (*) The two performance figures correspond to training on 400 / 900 background images respectively. In both cases, classification is performed against an unseen test set of road backgrounds (as in [45]), which was folded-in. See text for explanation.

three background topics). The 'test' images from [45] were than 'folded in' as described above. For example in the case of motorbikes the 800 images are divided into 400 training and 400 test images. The test is one against many (the other categories). Each test image is assigned to object topic k with maximum $P(z_k|d_{test})$ (background topics are ignored here). The confusion table is shown in table 6.3. The mean of the diagonal of the confusion table is 98.2%. This compares favorably with the constellation model of Fergus *et al.* [45], which achieves on the same four classes performance of 86.8% (measured by the mean of the diagonal of the confusion table). This is despite the fact that our method is essentially unsupervised (we come back to this point bellow).

(4) Binary classification of category against background. Up to this point the classification test has been one against many. In this test we examine performance in classifying (unseen) images against (unseen) background images. The pLSA model is fitted to training subsets of each category and only 400 (out of 900) background images. Testing images of each category and background images are 'folded-in'. The mixing proportion $P(z_k|d_{test})$ for topic k across the testing images d_{test} (i.e. a row in the landscape matrix $P(z|d)$ in figure 6.1b) is then used to produce a ROC curve for the topic k. Equal error rates for the four object topics are reported in table 6.4.

Note that for Airplanes and Faces our performance is similar to that of [45] despite the fact that our 'training' is unsupervised in the sense that the identity of the object in an image is *not known*. This is in contrast to [45], where each image is labelled with an identity of the object it contains, i.e. about 5×400 items of supervisory data vs. one label (the number of topics) in our case.

In the case of motorbikes we perform worse than [45] mainly due to confusion between motorbike images containing a textured background and the textured background topic. The performance on Cars rear is poor because Car images are split between two topics in training (a similar effect happens in experiment (1) for K=6). This splitting can be avoided by including more background images. In order to make results comparable with [45], Cars rear images were classified against a completely new background dataset containing mainly empty roads. This dataset was not seen in the learning stage and had to be 'folded-in' which makes the comparison on Cars rear slightly unfair to the topic discovery approach.

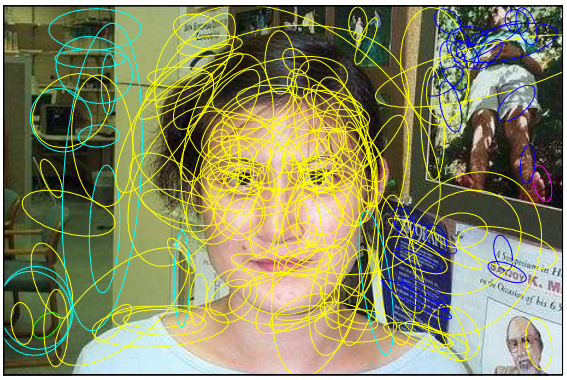
6.4.3 Segmentation

In this section we investigate the image's spatial segmentation that have been discovered by the model fitting. As a first thought, it is absurd that a bag of words model could possibly have anything useful to say about image segmentation, since all spatial information has been thrown away. However, the pLSA model delivers the posteriors given by equation 6.3, and consequently for a word occurrence in a particular document we can examine the probability over different topics.

Figure 6.7 shows an example of 'topic segmentation' induced by $P(z_k|w_i, d_j)$ for the case of experiment (2) with 7 topics. Figure 6.8 shows examples of topic segmentation for unseen images (experiment (3), 7 topics). In particular, we show only visual words with $P(z_k|w_i, d_j)$ greater than 0.8. There is an impressive alignment of the words with the corresponding object areas of the image. Note the words shown are not simply those most likely for that topic. Rather, from (6.3), they have high probability of that topic *in this image*. This is an example of overcoming polysemy – the probability of the particular word depends not only on the probability that it occurs within that topic (face, say) but also on the probability that the face topic has for that image, i.e. the evidence for the face topic from other regions in the image.



(a)



(b)

Topic	$P(topic\|image)$	**# regions**
1 Face (yellow)	0.48	128
2 Motorbikes (green)	0.07	1
3 Airplane (black)	0.00	0
4 Cars (red)	0.03	0
5 Backg I (magenta)	0.09	1
6 Backg II (cyan)	0.17	12
7 Backg III (blue)	0.15	23

(c)

Figure 6.7: Image as a mixture of visual topics (Experiment (2)) using 7 learned topics). (a) Original image. (b) Image as a mixture of a face topic (yellow) and background topics (blue, cyan). Only elliptical regions with topic posterior $P(z|w,d)$ greater than 0.8 are shown. In total 7 topics were learnt for this dataset which contained faces, motorbikes, airplanes, cars, and background images. The other topics are not significantly present in the image since they mostly represent the other categories and other types of background. Table (c) shows the mixture coefficients $P(z|d)$ for this particular image. In total there are 693 elliptical regions in this image of which 165 (102 unique visual words) have $P(z|w,d)$ above 0.8 (those shown in (b)). The topics assigned to visual words, dependent on the other words in the analyzed image, correspond well to object categories.

6.4.4 MIT image dataset results

The MIT dataset [154, 161] contains 2,873 images of indoor and outdoor scenes and is significantly more varied than just the four classes in the Caltech dataset used in the previous section. The dataset also includes large digital photographs (up to 2,048 $\times$ 1,536 pixels). To reduce the computational expense, these are downsampled to have their larger dimension 300 pixels while preserving the aspect ratio. Again all images have been converted to grayscale before processing.

Visual words are computed for each image as outlined in section 6.3. Note that the vocabulary of 2,224 visual words clustered from the Caltech images is used here. This tests the generalization performance of the Caltech vocabulary on a new unseen data.

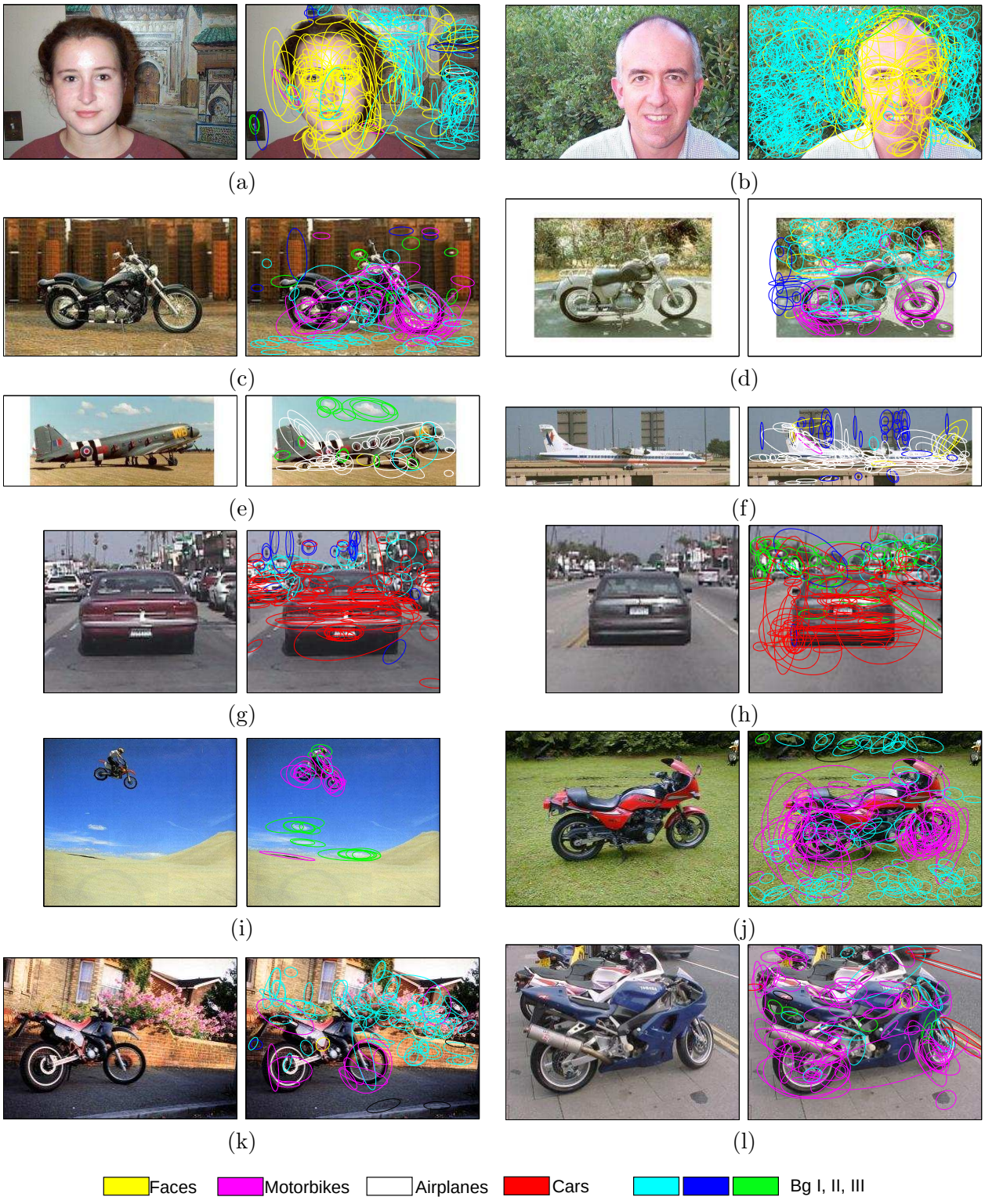


Figure 6.8: Examples of segmentation for new unseen images using pLSA. Topics were learned from a set of training images from each category and all background images. Each new image was 'folded in', see text. Two examples from each category are shown: Faces (a,b), Motorbikes (c,d), Airplanes (e,f), Cars rear (g,h). Four examples from the ETH motorbike dataset [79] are shown in (i–l). Note the significant changes in scale and viewpoint. The colour key for the seven different topics is shown at the bottom of the figure.

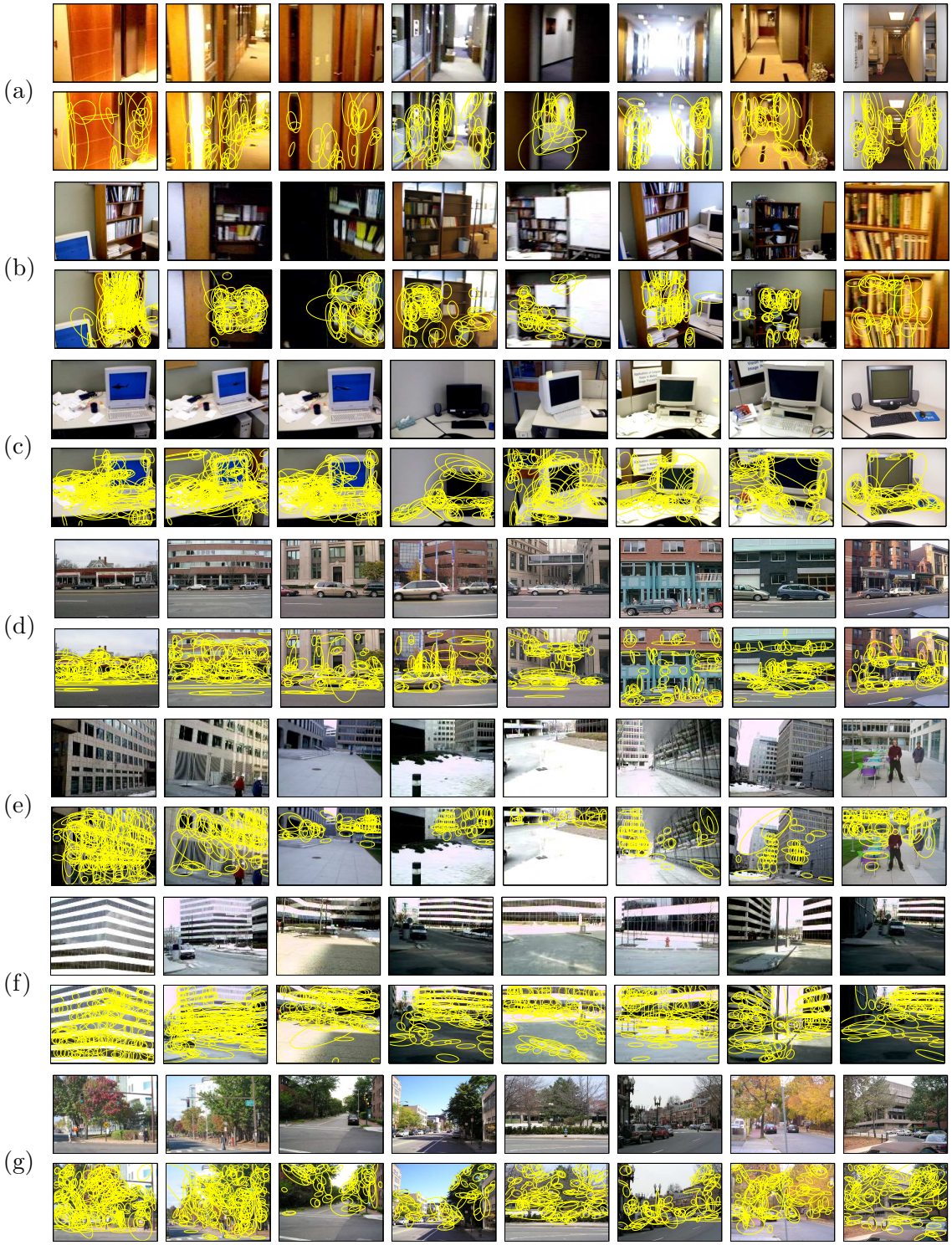


Figure 6.9: Example segmentations induced by seven (out of 20) discovered topics on the MIT dataset. Examples from the first 20 most probable images for each topic are shown. For each topic the top row shows the original images and the bottom row shows visual words belonging to that particular topic in that image. Note that we can give semantic interpretation to these topics: (a) covers corridors in 18 out of the top 20 images; (b) covers bookshelves in 17 out of the top 20 images; (c) covers computers in 19 out of the top 20 images; (d)-(f) cover different types of buildings in 18, 19, and 17 out the top 20 images respectively; (g) covers trees and grass in 18 out of the top 20 images.



Figure 6.10: Example segmentations on the MIT dataset for the 20 topic decomposition. Left: the original image. Middle: all detected regions superimposed. Right: the topic induced segmentation. Only topics (b), (c), (d), (e) and (g) from figure 6.9 are shown. The colour key is: (b) 'bookshelves' - magenta; (c) 'computers' - cyan; (d,e) 'buildings' - red; (g) 'trees','grass' - green. In each segmentation (the third column) only the two colored topics are shown and only visual words with $P(z|d,w) > 0.5$ are shown.

The annotation for the MIT dataset is incomplete, i.e. many object instances are not labelled. For this reason we present only qualitative results on this data.

Topic discovery and segmentation: We fit pLSA with $K = 20$ topics to the entire dataset. Figure 6.9 shows examples of most probable images for 7 of the 20 learned topics. These topics, more so than the rest, have a clear semantic interpretation, and cover objects such as computers, buildings and trees. For each topic shown, images with high $P(z|d)$ for that topic are shown. The overlaid visual words are those with $P(z|d, w) > 0.8$.

Figure 6.10 shows examples of topic induced segmentations in images with multiple objects, e.g. 'computers' and 'bookshelves' or 'buildings' and 'trees'. In each topic induced segmentation only visual words with $P(z|d, w) > 0.5$ are shown. Note that the results demonstrate that images can be accessed by the multiple objects they contain (in contrast to GIST [103], for example, which classifies an entire image).

It is also interesting to observe the outcome of the pLSA learning when the number of topics K is varied. For two learned topics the data is split into indoor and outdoor categories with around 80% accuracy. With increasing K a topic structure starts to emerge, but several of the topics discovered separately for $K = 20$ are merged together, for example for $K = 4$ (topic labels refer to figure 6.9): (i) 'building' topics (d-f) are discovered as one topic; (ii) the 'computer' topic (c) includes additional office scenes; (iii) the 'grass and trees' topic (g) includes additional street scenes; (iv) the 'corridor' topic (a) is merged with the 'bookshelves' topic (b) and some other indoor and outdoor scenes. This behaviour suggests that there might not be a single correct number of topics to learn. Instead, a hierarchy of topics [17, 63] might be more appropriate.

6.5 Discussion: What is being learnt by pLSA

It is interesting to inspect in more detail what is being learnt by the pLSA algorithm. In particular, we examine the results of experiment (2) of section 6.4.1, where 7 topics are fitted to 4,090 images containing four object categories and background images. Figure 6.11 illustrates the result of the pLSA matrix decomposition. Note that the cost minimized by fitting the pLSA model is a weighted

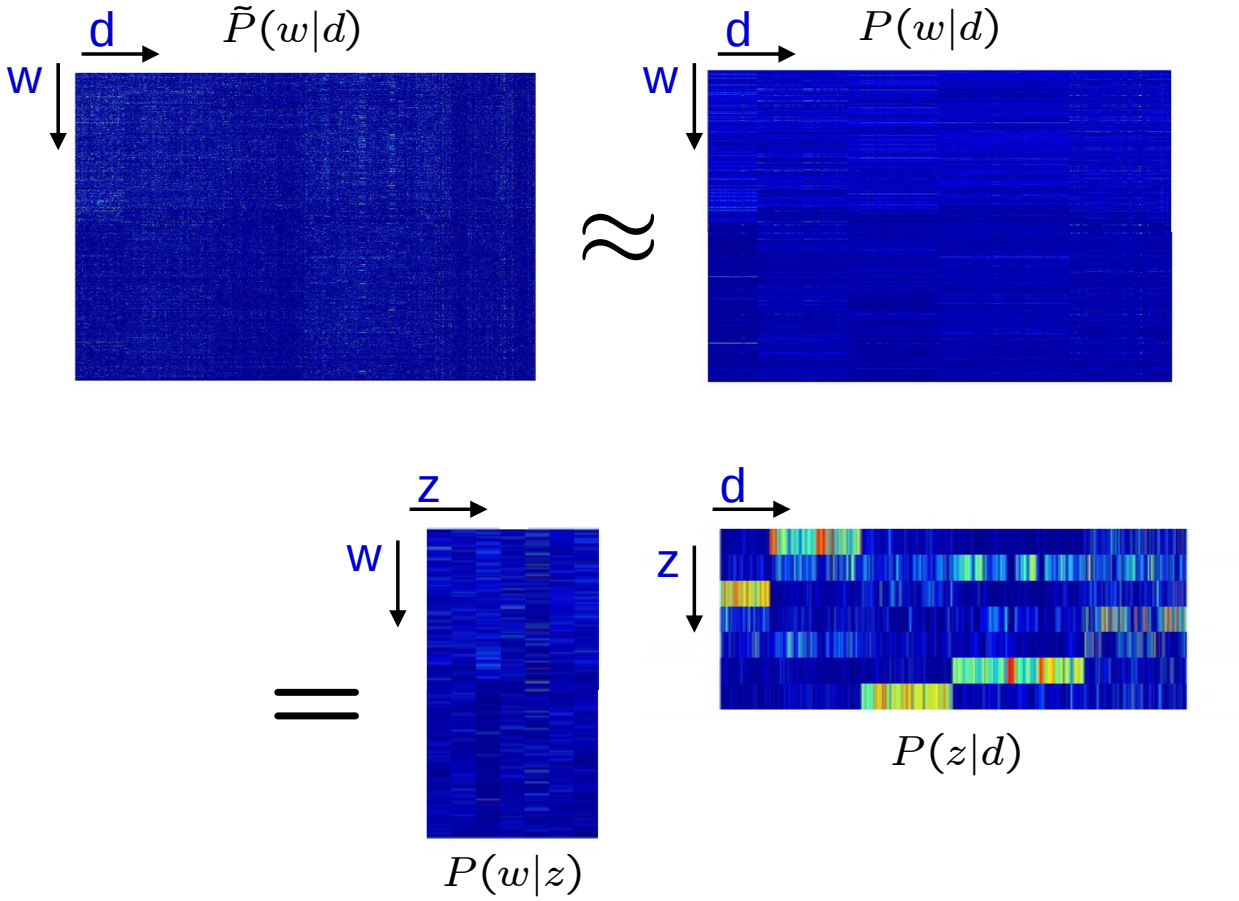


Figure 6.11: pLSA model fitting illustrated on data from experiment (2) of section 6.4.1. Red and yellow colours indicate high values. Dark blue colors indicate low values. w, d, z indicate (visual) words, documents (images) and topics respectively. In this experiment there are 2,224 words, 4,090 images and 7 topics. The measured empirical distributions $\tilde{P}(w|d)$ are approximated by model distributions $P(w|d)$, which are decomposed into topic distributions $P(w|z)$ common to all documents and mixing weights $P(z|d)$ specific to each document. Note that the structure of the data is visible in the matrix of topic mixing weights $P(z|d)$. This is because images are sorted according to class into faces, motorbikes, airplanes, cars and background; and blocks of images belonging to one object class that have high value of a particular topic mixing weight are visible. For example the first block of images with high mixing weight for topic 3 contains 435 images of faces and the next block of images with high topic mixing weight for topic 1 contains 800 images of motorbikes.

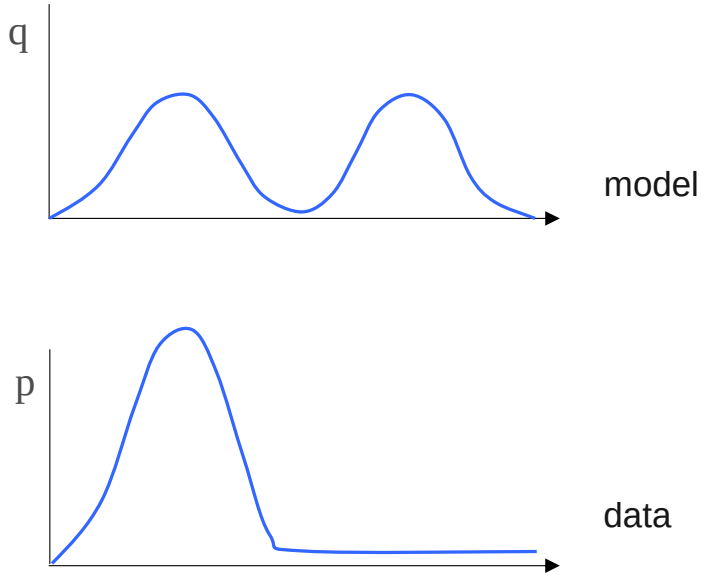


Figure 6.12: Illustration of the asymmetry of the Kullback-Leibler (KL) divergence (given by equation 6.7) between two distributions p and q. Note that in this case $D(p||q)$ would be low despite the fact that p is missing the second 'mode' present in q. This is because KL divergence, $D(p||q)$, is 'weighted' by p and therefore q can have high probability mass in regions where p is negligible. In the case of pLSA, $D(\tilde{P}(w|d_j) \,||\, P(w|d_j))$ is minimized, i.e. p is the measured empirical distribution of the data, $\tilde{P}(w|d_j)$, and q is the model distribution, $P(w|d_j)$.

sum of Kullback-Leibler (KL) divergences,

$$D(\tilde{P}(w|d_j) \,||\, P(w|d_j)) \tag{6.6}$$

between the measured empirical distributions $\tilde{P}(w|d_j)$ and model distributions $P(w|d_j)$ over all documents d_j. This is shown in detail in appendix A.1. The Kullback-Leibler divergence is a measure of dissimilarity between two probability distributions, say p and q, and is defined as

$$D(p||q) = \sum_i p_i \log \frac{p_i}{q_i}. \tag{6.7}$$

KL divergence is always non-negative (and zero if $p = q$) but is not symmetric, i.e. $D(p||q) \neq D(q||p)$. The asymmetry of the KL divergence is an important property and is illustrated in figure 6.12. A similar illustration can be found in [172].

The asymmetry of the KL divergence allows the model (topic) distributions to be 'multimodal' with different 'modes' of the distribution modelling different appearances and intra-class variations.

The measured distribution in a particular image does not have to contain all the modes and still be well explained by the model distribution in terms of low KL divergence given by equation (6.6). For example, in the case of, say, motorbikes, the model distribution can have high probability for visual words capturing, say, different types of wheels. The visual word distribution measured in a particular motorbike image does not have to contain all the different 'wheel' visual words and still have low KL divergence to the model distribution provided visual words present in the image have high model distribution probability. These points are illustrated on the 'face' object class in figures 6.13 and 6.14.

6.6 Application: object class retrieval

The goal here is to retrieve all occurrences of an object class (e.g. 'airplanes') from a database of images, given a single image as a query (e.g. 'a particular example of an airplane'). This is a challenging task as we have to overcome the intra-class variability, i.e. we would like to find all kinds of airplanes including, say, 'passenger jets' even when the query is, say, 'a fighter jet'. Note also, that the amount of labelled training data is very limited. The user specifies a single positive example – the query image.

The approach we take is to fit the pLSA model to the image collection and 'discover' a number of topics and their corresponding image specific mixing weights. As was shown in previous sections, some of the learnt topics correspond fairly well to object classes. A novel query image is then expressed in terms of the learnt topics. This gives us a vector of mixing weights, indicating the proportion of each topic (object) in the image. The retrieval is then performed by comparing the 'object' mixing weights of the query image with the 'object' mixing weights of all images in the collection.

In more detail, given a collection of images and their visual word histograms, we compute common topics, $P(w|z)$, and their corresponding image specific mixing weights $P(z|d)$. A novel query image is expressed in terms of the learnt topic vectors by the 'fold-in' strategy, outlined in section 6.4.2, which gives us the query image mixing weights $P(z|d_q)$. The similarity score between the query vector, $P(z|d_q)$, and document vector, $P(z|d)$, is computed using the normalized scalar product (3.4). This is the strategy used in [64], but other scoring functions such as L_1 distance or

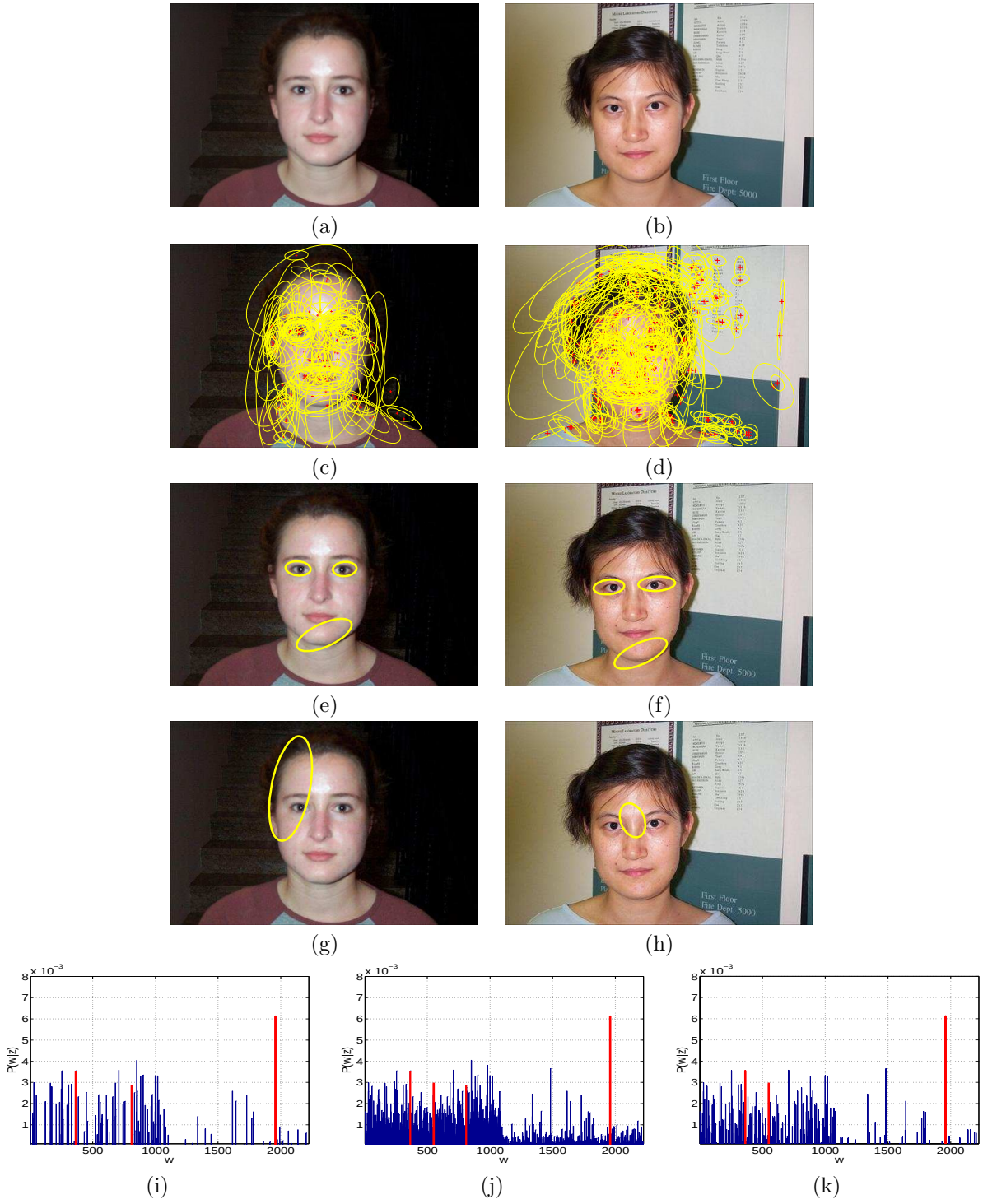
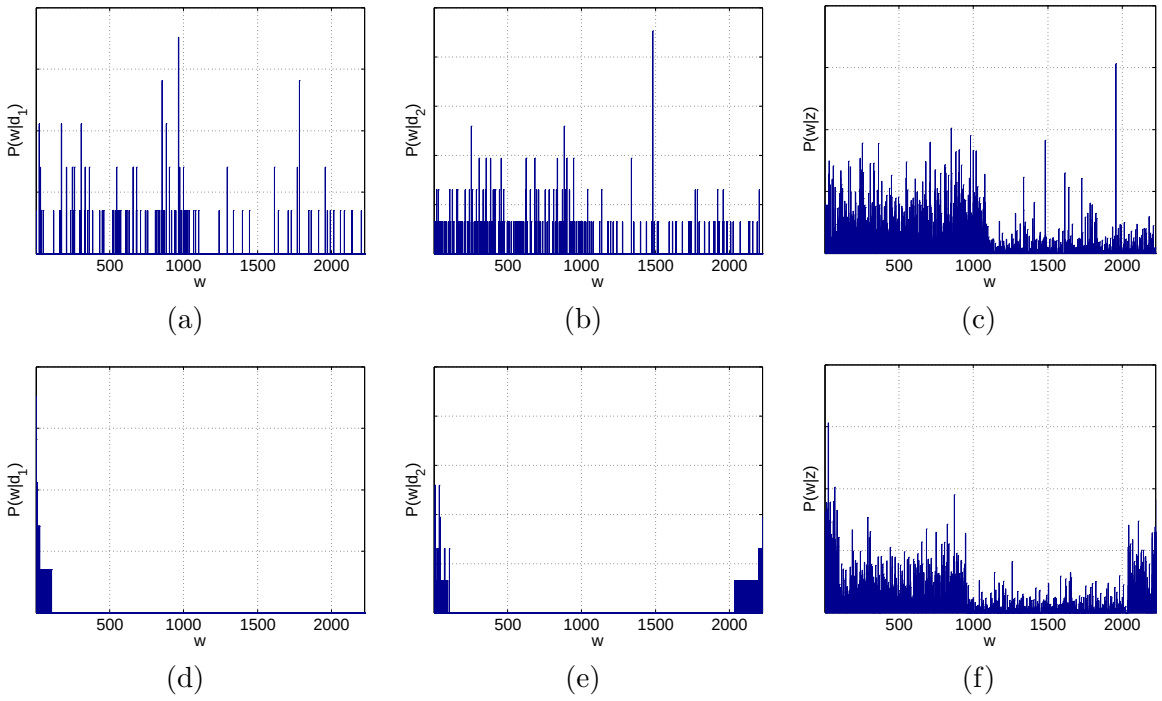


Figure 6.13: (a),(b) Two images from the 'face' object class. (c),(d) All detected visual words. (e),(f) Examples of 'face' visual words detected in both images – the 'eye' visual word (1959) and the 'chin' visual word (362). (g),(h) Examples of 'face' visual words detected only in one image and missing in the other image. Visual words 809 and 550 are shown in (g) and (h) respectively. (i)-(k) The distribution of visual words, $P(w|z)$, for the face topic learned in experiment (2) of section 6.4.1. (i) and (k) show only visual words present in images (a) and (b) respectively. (j) shows the entire distribution. Visual words 362, 550, 809 and 1959 are highlighted in red.



(a) (b) (c)

(d) (e) (f)

(g)

$D(P(w\|d_1)\|\|P(w\|d_2))$	=	4.38
$D(P(w\|d_2)\|\|P(w\|d_1))$	=	4.76
$D(P(w\|d_1)\|\|P(w\|z))$	=	1.95
$D(P(w\|d_2)\|\|P(w\|z))$	=	1.69

Figure 6.14: (a) and (b) show distributions of visual words for image (a) and image (b) of figure 6.13, respectively. (c) shows the face topic distribution learned in experiment (2) of section 6.4.1. (d)-(f) The same distributions as in (a)-(c) but sorted such that visual words present in (a) are sorted with decreasing probability $P(w|d_1)$ in bins 1-106, visual words present in (b) but missing in (a) are sorted with increasing probability $P(w|d_2)$ in bins 2038-2224, and visual words not present in either (a) or (b) are sorted by their original index in bins 107-2037. Note from (f) that: (1) some visual words with high face topic probability $P(w|z)$ are present in (a); (2) some visual words with high face topic probability $P(w|z)$ are present only in (b) but not in (a), and (3) many other visual words with high $P(w|z)$ are not present in either (a) or (b). Table (g) shows measured KL divergence between distributions (a)-(c). Note that KL divergence between $P(w|d_1)$ and $P(w|d_2)$ is high but both $P(w|d_1)$ and $P(w|d_2)$ have low KL divergence to $P(w|z)$.

KL divergence are certainly possible.

In following we show object class retrieval on two databases. First, quantitative retrieval results are shown on the Caltech database, where ground truth object labelling is available. Second, qualitative results are shown on retrieval of object classes from an entire feature length movie. In both cases, results are compared with a baseline method, which is described next.

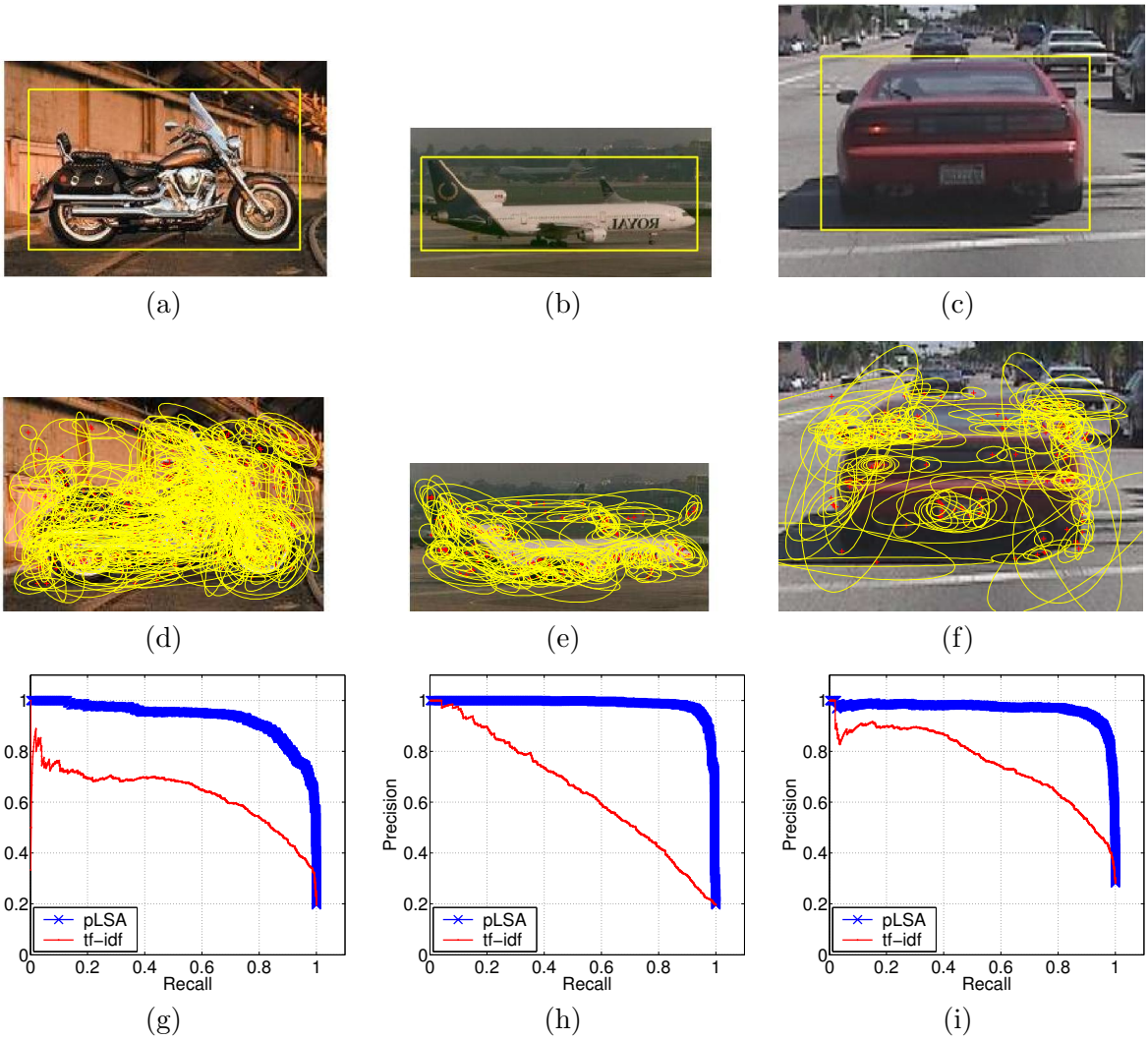
Baseline retrieval method: The baseline method implements the standard retrieval using visual word frequencies and tf–idf weighting as described in section 3.4.1. In particular, images are represented using the bag-of-words representation over the vocabulary created from a subset of the Caltech database as described in section 6.3. The similarity score between the query vector and document vector is measured using the normalized scalar product given by 3.4.

6.6.1 Retrieval within the Caltech database

Here we evaluate retrieval on the Caltech database used in experiment (2) of section 6.4.1. The database contains 4,090 images of four object classes and background. First, a $K(= 7)$ topic pLSA model is fitted to all the data. Given a query image with a user outlined bounding box, a histogram of visual words within the bounding box is 'folded-in' as described in section 6.4.2. The resulting vector of topic mixing weights is used for retrieval. Figure 6.15 shows precision-recall curves for three retrieval examples within the Caltech database. The table of figure 6.15(j) shows mean average precision for four object classes averaged over ten randomly selected query images for each class. For each query the bounding box was manually selected by the user. Note that the pLSA method outperforms the baseline method in all four cases with the largest difference in the case of 'airplane' retrieval.

6.6.2 Retrieval from a movie

Here we perform object class retrieval on a database of 5,000 keyframes from the movie 'Pretty Woman'. The processing and the retrieval algorithm are summarized in figure 6.16. Note that both the query image and all the database images are *not* used in any stage of the 'learning', i.e. the vocabulary building, and the pLSA topic learning. Note also that the only supervision provided is the number of topics to learn on the Caltech data ($K = 7$). In the retrieval stage the user provides



Mean Average Precision

Object class	pLSA	tf-idf
Faces	0.95	0.89
Motorbikes	0.88	0.69
Airplanes	0.92	0.37
Cars	0.83	0.69

(j)

Figure 6.15: **Object class retrieval within the Caltech database.** Each column shows one example query within the Caltech database. (a-c) The original query image with the user outline bounding box. (d-f) Query images with detected affine covariant regions superimposed. Only regions with the centre within the bounding box are shown. (g)-(i) Precision-recall graphs. Each graph shows a precision-recall curve for (1) retrieval based on pLSA topic mixing weights (blue crosses) and (2) retrieval using the baseline method based on tf-idf weighted visual word frequency vectors (red dots). Note that the pLSA retrieval significantly outperforms the baseline retrieval method in all the three cases. (j) Average Precision (area under the precision-recall curve) for four object classes in the Caltech database and the two retrieval schemes. The results shown are averaged over ten randomly selected query images for each class.

1. **Learning topic vectors (off-line)**
   Given a set of unlabelled training images of several object classes (e.g. faces, motorbikes, airplanes, cars, background):

   - Detect affine covariant regions in each image. Represent each region by a SIFT descriptor (section 6.3).
   - Build a visual vocabulary by clustering regions from a subset of the images. Assign each region descriptor to the nearest cluster centre (section 6.3).
   - Fit a $K(= 7)$ topic pLSA model, i.e. compute topic ('object') vectors $P(w|z)$ common to all images (section 6.2).

2. **Image database pre-processing (off-line)**
   Given a set of unlabelled images (e.g. keyframes from an entire movie):

   - Detect affine covariant regions in each image. Represent each region by a SIFT descriptor. Assign each region to the nearest cluster centre (obtained in step 1). Represent each image by a visual word histogram (section 6.3).
   - Express each image in terms of topic vectors $P(w|z)$ learned in step (1), i.e. compute mixing weights $P(z|d)$ indicating the proportion of each topic in the image (section 6.4.2).

3. **At run-time**
   Given a novel query image with a user selected query region:

   - Detect affine covariant regions. Represent each region by a SIFT descriptor. Assign each region to the nearest cluster centre (obtained in step 1). Represent the image by a visual word histogram (section 6.3).
   - Express the query image in terms of topic vectors $P(w|z)$ learned in step (1), i.e. compute mixing weights $P(z|d_{new})$ indicating proportion of each topic in the image (section 6.4.2).
   - Rank database images based on similarity of their topic mixing weight vectors $P(z|d)$ to the the query image mixing weight vector $P(z|d_{new})$ (section 6.6).

Figure 6.16: The object class retrieval algorithm.

a single image (and a query region) indicating the object class of interest (e.g. an airplane to search for airplanes).

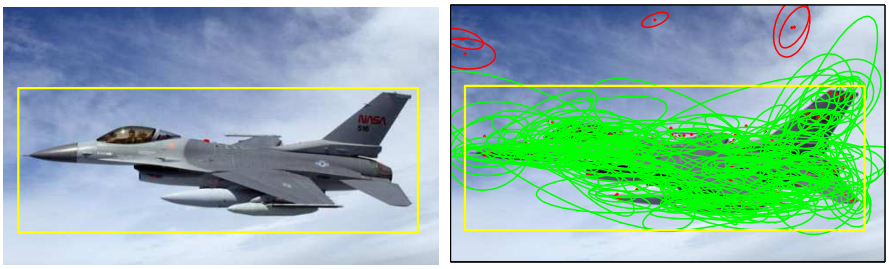
Figures 6.17 and 6.18 show two examples of 'airplane' retrieval within the movie 'Pretty Woman'. Note the amount of intra-class variation handled by the algorithm. The user specified query image is a 'fighter jet' and a large 'passenger jet' in figures 6.17 and 6.18 respectively, whereas the images retrieved from the movie contain a 'personal jet'. This is possible because the retrieval is performed using the topic mixing weights $P(z|d)$. The intuition is that images containing airplanes have high mixing weight for the airplane topic. Recall from the discussion in section 6.5 that two images can have a high 'airplane' topic mixing weight despite not having many common visual words. This is possible due to the 'multimodality' of the 'airplane' topic vector, learned from many training airplane images, capturing the intra-class variations.

Figures 6.19 and 6.20 show two retrieval examples of 'faces' and 'cars' respectively. In all cases the object class retrieval is compared with the baseline method described in section 6.6. In the case of the baseline method, the 'idf' part of the weights is estimated on the database of movie keyframes.

6.7 Conclusions and discussion

We have demonstrated that it is possible to learn visual object classes simply by looking: we identify the object categories for each image with the high reliabilities shown in table 6.1, using a corpus of unlabelled images. Furthermore, using these learnt topics for classification, we reproduce the experiments (training/testing) of [45], and obtain very competitive performance – despite the fact that [45] had to provide about (400× number of classes) supervisory labels, and we provide one label (number of topics). Furthermore, we have applied the proposed object discovery approach to a more challenging MIT dataset, which contains indoors and outdoors scenes with a significant amount of variation. Figures 6.9 and 6.10 indicate that semantically meaningful topics are discovered even in this more challenging data.

We have explored an extreme approach where no spatial propagation of information is used. Despite this essentially orderless image representation, visual words with the highest posterior probabilities for each object correspond fairly well to the spatial locations of each object. This is



(a)

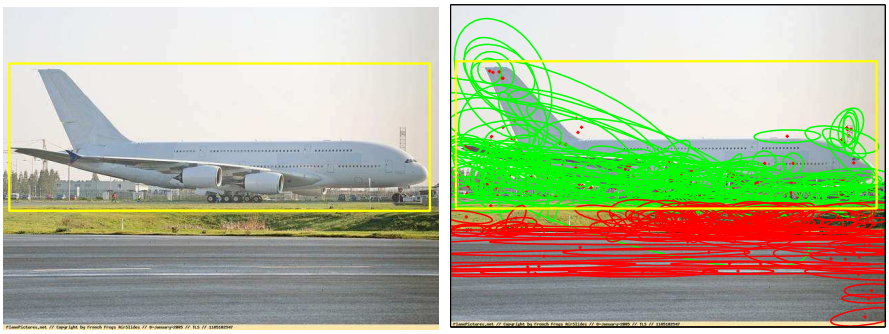


(b)



(c)

Figure 6.17: **Object class retrieval. Example I.** (a) Left: The query image (downloaded from the internet) with the user outlined query rectangle. (a) Right: The query image with all detected affine covariant regions superimposed. Regions inside the query rectangle are shown in green. (b) Top 20 retrieved keyframes from the movie 'Pretty Woman' using the topic (object) specific mixing weights $P(z|d)$. (c) Top 20 retrieved keyframes using the visual word frequency vectors and 'tf-idf' weighting. Note how in (b) keyframes containing airplanes are retrieved while in (c) the top ranked keyframes are false positives.



(a)

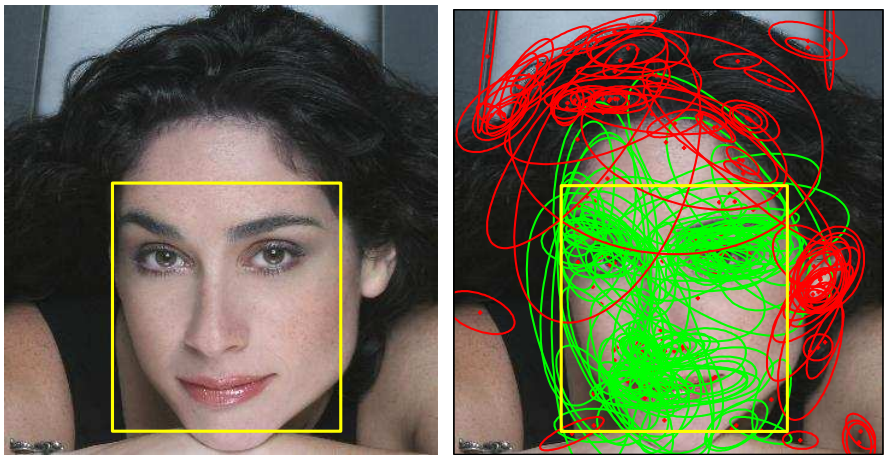


(b)



(c)

Figure 6.18: **Object class retrieval. Example II.** (a) Left: The query image (downloaded from the internet) with the user outlined query rectangle. (a) Right: The query image with all detected affine covariant regions superimposed. Regions inside the query rectangle are shown in green. (b) Top 20 retrieved keyframes from the movie 'Pretty Woman' using the topic (object) specific mixing weights $P(z|d)$. (c) Top 20 retrieved keyframes using the visual word frequency vectors and 'tf-idf' weighting. Note how in (b) keyframes containing airplanes are retrieved while in (c) the top ranked keyframes are false positives



(a)

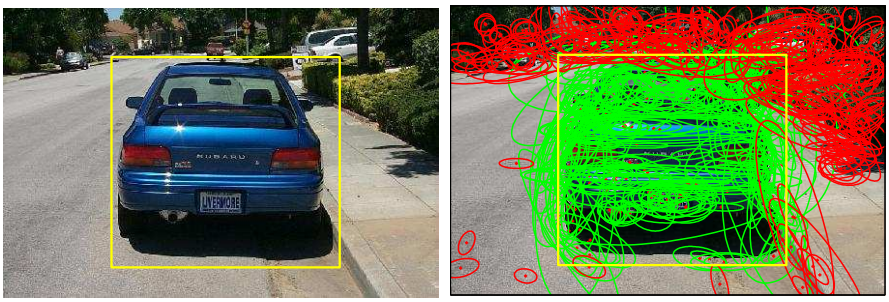


(b)



(c)

Figure 6.19: **Object class retrieval. Example III.** (a) Left: The query image (downloaded from the internet) with the user outlined query rectangle. (a) Right: The query image with all detected affine covariant regions superimposed. Regions inside the query rectangle are shown in green. (b) Top 15 retrieved keyframes from the movie 'Pretty Woman' using the topic (object) specific mixing weights $P(z|d)$. (c) Top 15 retrieved keyframes using the visual word frequency vectors and 'tf-idf' weighting. Here keyframes in both (b) and (c) contain faces.



(a)



(b)



(c)

Figure 6.20: **Object class retrieval. Example IV.** (a) Left: The query image (downloaded from the internet) with the user outlined query rectangle. (a) Right: The query image with all detected affine covariant regions superimposed. Regions inside the query rectangle are shown in green. (b) Top 15 retrieved keyframes from the movie 'Pretty Woman' using the topic (object) specific mixing weights $P(z|d)$. (c) Top 15 retrieved keyframes using the visual word frequency vectors and 'tf-idf' weighting. Note how in (b) keyframes containing cars are retrieved while in (c) the top ranked keyframes are mostly false positives

rather remarkable considering our use of the bag of words model.

The learnt object models have also been applied to retrieval of object classes in a database of several thousand keyframes from a feature length movie.

For future work we plan to investigate the following areas: (i) Currently we have shown successful object discovery results on a small set (4-7) object classes and several thousand images. We plan to study performance as the number of classes (and images) is scaled-up by orders of magnitude. (ii) Results of section 6.4.4 suggest, that there might not be a single correct number of topics to learn. Instead, a hierarchy of topics [17, 63] might be more appropriate. (iii) The proposed approach works with no supervision. It would be interesting to investigate the scenario when a small amount of auxiliary information is available. For example, some images might have a textual annotation specifying some objects present in the image [34]. (iv) The visual words currently employed capture mostly local appearance. An additional visual vocabulary based on quantized descriptors for boundary/local shape might significantly improve performance for shape based object classes, e.g. bottles, horses or camels [46, 131]. (v) Recently [3, 20], promising results have also been reported with a vocabulary based local appearance sampled on a dense spatial grid in an image. This should allow the appearance of objects like 'sky' or 'water' to be captured, where sparse region detectors might not fire. (vi) Finally, an interesting area of research would be extending the basic 'bag of words' object model employed in this chapter with some amount of spatial information [44, 145, 146].

Chapter 7

Discussion

In this chapter we summarize thesis contributions, review some recent developments in the literature, and discuss avenues for future research.

7.1 Contributions of the thesis

In this thesis, we have proposed models and methods for efficient visual search for objects in images and videos. In particular, we focused on retrieval of object instances, object categories, and human faces. Retrieval results were shown on entire feature length movies. The major contributions are summarized below:

- In chapter 3, we applied efficient techniques from text retrieval to visual search for particular objects. This required a visual analogy of a textual word, and we provided it by quantizing the appearance of local affine covariant regions. Object retrieval results were shown for several objects despite camera viewpoint changes, illumination variations, partial occlusion, and background clutter.

- In chapter 4 we developed a novel representation suitable for retrieval of 3D or deformable objects. The representation is based on storing several exemplars capturing different visual aspects of a 3D object or various appearances of a deforming object. The representation is built automatically from video shots, using novel tracking and motion segmentation techniques. In the implementation there are three areas of novelty. First, we developed a novel

tracking algorithm using affine covariant regions, which is able to develop very long tracks on moving objects and track through some amount of deformation. Second, we devised a novel motion segmentation algorithm able to isolate region tracks on moving foreground objects from background tracks. The motion segmentation algorithm is based on a sequence of weak grouping steps, and can deal with degenerate motions, missing data, and outliers. Third, we showed that objects can be associated across partial or total occlusions using matching.

- In chapter 5, we developed a novel representation for detected frontal faces in video. Individual face detections are first associated into groups by tracking affine covariant regions on faces. A set of faces of a particular person is then represented by a distribution of visual words from a face-specific visual vocabulary. The proposed representation is used for retrieval of faces of a particular person in a video.

- In chapter 6, we used the probabilistic latent semantic analysis (pLSA) model to discover visual object classes in image collections. The novelty lies in: (i) applying the pLSA model in visual domain; (ii) showing that 'bag-of-words' models of several visual object classes can be learnt, in an unsupervised way, from a collection of unlabelled images. Furthermore, these models are then applied for object category retrieval in a feature length movie.

7.2 Recent developments

In this section we discuss recent developments in the literature, many of which explicitly build on the work described in this thesis.

- Ho and Newman [62], and Wang, Zha and Cippola [170] apply visual words and text retrieval indexing methods for navigating mobile robots with mounted camera sensors. The particular applications are loop-closing for Simultaneous Localization and Mapping (SLAM), and robot localization, respectively.

- Nister and Stewenius [99] develop a hierarchical visual vocabulary, where visual words are organized in a tree. Grauman and Darrell [56] develop a 'pyramid match kernel' where quantization of appearance space is performed by histogram binning on multiple levels of

'coarseness'.

- Rothganger *et al.* [113], similarly to our work described in chapter 4, build 3D object models automatically from video shots. Their representation is also based on affine covariant regions, but involves computing the 3D structure of the object.

- Fei-Fei and Perona [42], independently of us, used the Latent Dirichlet Allocation (LDA) model for categorizing scenes (outdoor, indoor, inside city, etc.) in a *supervised* setting. The LDA model is a Bayesian version of the pLSA model used in chapter 6 for unsupervised object category recognition. Similarly to Fei-Fei and Perona, Quelhas *et al.* [109] applied the pLSA model for categorizing scenes, again in a supervised setting.

- Fergus *et al.* [44] proposed an extension of the pLSA model, incorporating spatial relations between visual words. The model was successfully applied for object category recognition.

- Sudderth *et al.* [146, 145] proposed hierarchical models of scenes, objects, and parts, also incorporating spatial relations between visual words.

- Russel *et al.* [117] use multiple image segmentations to propose spatial groupings of visual words within images. Topic discovery methods are then used to recover objects, which correspond to visual word groupings segmented consistently in the image collection.

7.3 Future work

In this section we outline possible directions for future research.

Object instance retrieval

- We have shown successful object retrieval results within up to two feature length movies, essentially searching through 300,000 frames indexed by more than 11,000 keyframes. One direction of future work is scaling-up, with the ultimate goal of indexing billions of images available on-line. One step in this direction is the work of Nister and Stewenius [99]. They show successful object retrieval on a database of 50,000 CD covers. Their vocabulary, organized in a tree, also allows for the fast insertion of new images into the database.

- In our work we build a fixed visual vocabulary from one or two movies and the generalization performance of the visual vocabulary is fairly limited. The next step would be learning a universal visual vocabulary, with better generalization to new unseen objects. Alternatively, visual vocabulary might not be static but instead might evolve over time when new images are added to the database.

- We have developed an efficient method for indexing the appearance of local patches. Another interesting research direction would be integrating appearance and spatial relations between regions into one efficient indexing scheme. Geometric hashing [176], popular in the early 1990s, might be a good source of inspiration.

- The current object representation, based on local regions, requires at least lightly textured objects. The next step may be developing efficient representations for object contours based on image edges, and should enable retrieval of textureless objects (mugs, bottles) or wiry objects (chairs, bicycles) [22, 97].

- In chapter 4, we have developed methods for obtaining representations of 3D (and deformable) objects, based on motion within a video shot. One difficulty with the proposed approach is that to represent multiple visual aspects of a 3D object, such aspects have to be present in the video shot. A possible solution may be an object based query expansion, where multi-aspect models would be built in an incremental way from multiple shots retrieved from different parts of the video. For example, imagine querying for the front of a van. Some highly ranked shots might also contain, say, the side of the van, which might be associated with the current model by motion within the shot. This new enhanced model, containing both front and side of the van, might then be used for another set of queries. This process might be iterated until all shots containing the van are found.

Retrieval of faces

- Chapter 5 shows successful retrieval of faces of a particular person in video. One possible extension would be automatic clustering of faces within the entire movie. Recently, Berg *et al.* [16] clustered faces detected in images appearing on news websites. Names mentioned in

the text surrounding images were used as a weak supervisory signal. A similar approach can be explored in the context of faces in videos. The weak supervisory signal can be obtained, for example, from movie scripts. Additional cues available in the video can be also explored. For example, lip movements can be used to detect people who speak in the video.

Retrieval of object classes

- In chapter 6 we have demonstrated that 'bag-of-words' models of object categories can be learnt from an unlabelled collection of images in an unsupervised way. Currently, we have shown results on a small number of object categories (4-7). One of the main goals for future work is scaling-up to tens or hundreds of object categories. One way to achieve this might be increasing the discriminative ability of the 'bag-of-words' model, by including spatial information in the form of: (i) spatial groupings of visual words based on multiple bottom-up image segmentations, as recently suggested by Russell *et al.* [117]; or (ii) explicitly including spatial relations of visual words in the probabilistic model, as suggested by Fergus *et al.* [44] or Sudderth *et al.* [146].

- Current results also suggest that objects might be organized in a hierarchy. Ideally, such hierarchies of objects should be discovered automatically from image collections, for example by the cluster-abstraction model of Hofmann [63] or its Bayesian extension [17].

- The current visual vocabulary captures mostly local appearance. Many object classes (e.g. horses, cows) have a defining shape or contour, common across many instances, but the actual object appearance can vary substantially from instance to instance. The next step would be learning new vocabularies, representing object-contours. Initial work in this direction has been done by Opelt *et al.* (2006) [105].

- Another very interesting research direction is building 'visual vocabularies' for motions and actions, by including spatio-temporal information available in the video. In this thesis, we have developed some of the basic tools for such analysis, for example a reliable 'visual word' tracker, able to cope with some amount of deformation. This opens-up the possibility of modelling and recognizing actions and interactions between objects and people in the video.

Appendix A

Appendix

A.1 Derivation of pLSA cost function

Here we derive the cost function minimized when fitting the probabilistic latent semantic analysis (pLSA) model described in chapter 6. Similar derivation can be found in [66].

Learning parameters of the pLSA model can be posed as a maximum likelihood learning problem. The likelihood of a collection of documents is written as

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N} P(w_i, d_j)^{n(w_i, d_j)}, \tag{A.1}$$

where

- $n(w_i, d_j)$ are the observed counts of word i in document j and $i \in \{1, \ldots, M\}$, $j \in \{1, \ldots, N\}$, i.e. we have M words and N documents.
- $n(d_j) = \sum_{i=1}^{M} n(w_i, d_j)$ is the total number of words in document d_j.
- $\tilde{P}(w_i|d_j) = \frac{n(w_i, d_j)}{n(d_j)}$ is the empirical distribution computed from the observed counts.
- $P(w_i, d_j) = P(d_j)P(w_i|d_j) = P(d_j) \sum_{k=1}^{K} P(z_k|d_j)P(w_i|z_k)$, where $P(d_j)$, $P(z_k|d_j)$ and $P(w_i|z_k)$ are the model parameters.

Taking logarithm of A.1, we can further write

$$\begin{aligned}
\mathcal{L} &= \log L && \text{(A.2)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(w_i,d_j) && \text{(A.3)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j)P(w_i|d_j) && \text{(A.4)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j) + \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(w_i|d_j) && \text{(A.5)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j) + \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\frac{n(d_j)}{n(d_j)}\log\left(P(w_i|d_j)\frac{\tilde{P}(w_i|d_j)}{\tilde{P}(w_i|d_j)}\right) && \text{(A.6)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j) - \sum_{i=1}^{M}\sum_{j=1}^{N} n(d_j)\tilde{P}(w_i|d_j)\left(\log\frac{\tilde{P}(w_i|d_j)}{P(w_i|d_j)} - \log\tilde{P}(w_i|d_j)\right) && \text{(A.7)}\\
&= \sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j) - \sum_{j=1}^{N} n(d_j)\sum_{i=1}^{M}\tilde{P}(w_i|d_j)\log\frac{\tilde{P}(w_i|d_j)}{P(w_i|d_j)} + \\
&\quad + \sum_{j=1}^{N} n(d_j)\sum_{i=1}^{M}\tilde{P}(w_i|d_j)\log\tilde{P}(w_i|d_j) && \text{(A.8)}\\
&= \underbrace{\sum_{i=1}^{M}\sum_{j=1}^{N} n(w_i,d_j)\log P(d_j)}_{A} \underbrace{- \sum_{j=1}^{N} n(d_j)\ D(\tilde{P}(w|d_j)\ ||\ P(w|d_j))}_{B} - \\
&\quad \underbrace{- \sum_{j=1}^{N} n(d_j)\ H(\tilde{P}(w|d_j))}_{C} && \text{(A.9)}
\end{aligned}$$

The goal is to maximize loglikelihood $\mathcal{L}$ with respect to $P(d_j)$ and $P(w_i|d_j)$, given measured counts $n(w_i,d_j)$. Note that the sum A in (A.9) can be maximized independently by $P(d_j) = n(d_j)/\sum_{j=1}^{N} n(d_j)$ (shown below) and the sum C in (A.9) is a constant dependent only on the measured data. The goal therefore remains to minimize the sum B in (A.9). A local minimum of this cost can be found using the expectation maximization (EM) algorithm, as outlined in [65].

Note that the sum B is a weighted sum of Kullback-Leibler divergences $D(\tilde{P}(w|d_j)\ ||\ P(w|d_j))$ between measured empirical distributions $\tilde{P}(w|d_j)$ and model distributions $P(w|d_j)$ for all documents d_j. Different documents are weighted by the number of words they contain, i.e. documents with more words are weighted more.

Note that the sum C is a weighted sum of entropies of measured empirical distributions $\tilde{P}(w|d_j)$.

The term A can be maximized independently of the term B, as A depends only on $P(d_j)$ and B depends only $P(w|d_j)$. All the remaining terms in both A and B are measured from the data. Note that, as defined above, $n(d_j) = \sum_{i=1}^{M} n(w_i,d_j)$ is the total number of words in document d_j.

The goal is to maximize A, with respect to $P(d_j)$, $j \in \{1, \ldots, N\}$, subject to the constraint that $P(d_j)$ sums to one,

$$\sum_{j=1}^{N} P(d_j) = 1. \tag{A.10}$$

This can be achieved by augmenting term A with appropriate Lagrange multiplier, λ,

$$\mathcal{G} = \sum_{j=1}^{N} n(d_j) \log P(d_j) + \lambda \left(1 - \sum_{j=1}^{N} P(d_j) \right). \tag{A.11}$$

Setting partial derivatives of $\mathcal{G}$ with respect to $P(d_j)$ to zero gives

$$\frac{\partial \mathcal{G}}{\partial P(d_j)} = \frac{n(d_j)}{P(d_j)} - \lambda = 0. \tag{A.12}$$

By combining (A.12) with constraint (A.10) we get

$$P(d_j) = \frac{n(d_j)}{\sum_{j=1}^{N} n(d_j)}. \tag{A.13}$$

A.2 Processed movies

Following is the list of feature length movies, which have been indexed for visual search.



Casablanca (1942, B&W)

Featuring Humphrey Bogart and Ingrid Bergman

Director: Michael Curtiz

Frames: 147,542
Shots: 675



Dressed To Kill (1946, B&W)

Featuring Basil Rathbone as Sherlock Holmes

Director: Roy William Neill

Frames: 128,544
Shots: 595



Groundhog Day (1993)

Featuring Bill Murray and Andie MacDowell

Director: Harold Ramis

Frames: 141,146
Shots: 752



Pretty Woman (1990)

Featuring Richard Gere and Julia Roberts

Director: Garry Marshall

Frames: 172,105
Shots: 1,151



Run Lola Run (1998)

Featuring Franka Potente and Moritz Bleibtreu

Director: Tom Tykwer

Frames: 109,175
Shots: 1,111

Bibliography

[1] http://www.robots.ox.ac.uk/∼vgg/research/vgoogle/.

[2] H. Aanaes, R. Fisker, K. Astrom, and J. M. Carstensen. Robust factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1215–1225, 2002.

[3] A. Agarwal and B. Triggs. Hyperfeatures - multilevel local coding for visual recognition. In *Proceedings of the European Conference on Computer Vision*. Springer, 2006. To appear.

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

[5] F. Aherne, N. Thacker, and P. Rockett. The bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1998.

[6] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell. Face recognition with image sets using manifold density divergence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:581–588, 2005.

[7] J. Aslam and M. Montague. Models for metasearch. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 276–284. ACM Press, 2001.

[8] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, ISBN: 020139829, 1999.

[9] K. Barnard, P. Duygulu, N. de Freitas, D. Forsyth, D. Blei, and M. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, February 2003.

[10] A. R. Barron and T. M. Cover. Minimum complexity density estimation. *IEEE Trans. on Information Therory*, 4:1034–1054, 1991.

[11] E. Bart, E. Byvatov, and S. Ullman. View-invariant recognition using corresponding object fragments. In *Proceedings of the European Conference on Computer Vision*, pages 152–165, 2004.

[12] A. Baumberg. Reliable feature matching across widely separated views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 774–781, 2000.

[13] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proceedings of the International Conference on Computer Vision*, pages 675–82, 1998.

[14] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, 2002.

[15] 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*, pages I:26–33, June 2005.

[16] T. Berg, A. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. Learned-Miller, and D. Forsyth. Names and faces in the news. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 848–854, 2004.

[17] D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, 2004.

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

[19] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7–56, 1987.

[20] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *Proceedings of the European Conference on Computer Vision*, 2006.

[21] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *7th Int. WWW Conference*, 1998.

[22] O. Carmichael and M. Hebert. Shape-based recognition of wiry objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(12):1537–1552, 2004.

[23] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:539–546, 2005.

[24] R. Choudhury, C. Schmid, and K. Mikolajczyk. Face detection and tracking in a video by propagating detection probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1215–1228, 2003.

[25] A. Chowdhury, S. Beitzel, and E. Jensen. Analysis of combining multiple query representations with varying lengths in a single engine. In *Proceedings of the International Conference on Information Technology: Coding and Computing*, pages 236–241, 2002.

[26] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[27] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

[28] 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*, pages I:10–17, 2005.

[29] 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.

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

[31] I. Davison, A. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007. To appear.

[32] F. De la Torre and M. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1–3):117–142, 2003.

[33] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(9):391–407, 1990.

[34] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the European Conference on Computer Vision*, pages IV:97–112, 2002.

[35] P. Duygulu and A. Hauptman. What's news, what's not? associating news videos with words. In *Proceedings of the International Conference on Image and Video Retrieval*, 2004.

[36] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proceedings of the International Conference on Computer Vision*, pages II:726–733, 2003.

[37] S. Eickeler, F. Wallhoff, U. Iurgel, and G. Rigoll. Content-Based Indexing of Images and Video Using Face Detection and Recognition Methods. In *ICASSP*, 2001.

[38] M. Everingham, J. Sivic, and A. Zisserman. Hello! My name is... Buffy – Automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*, 2006.

[39] M. R. Everingham and A. Zisserman. Regression and classification approaches to eye localization in face images. In *7th International Conference on Automatic Face and Gesture Recognition (FG2006)*, pages 441–446, 2006.

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

[41] 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 *IEEE CVPR Workshop of Generative Model Based Vision*, 2004.

[42] 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*, pages II:524–531, 2005.

[43] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1), 2005.

[44] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *Proceedings of the International Conference on Computer Vision*, pages II:1816–1823, October 2005.

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

[46] 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 I:242–256. Springer-Verlag, May 2004.

[47] V. Ferrari, T. Tuytelaars, and L. Van Gool. Wide-baseline multiple-view correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 718–725, 2003.

[48] V. Ferrari, T. Tuytelaars, and L. Van Gool. Integrating multiple model views for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 105–112, 2004.

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

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

[51] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach.*, 24(6):381–395, 1981.

[52] A. W. Fitzgibbon and A. Zisserman. On affine invariant clustering and automatic cast listing in movies. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 3, pages 304–320. Springer-Verlag, 2002.

[53] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:891–906, 1991.

[54] C. Goad. Special purpose automatic programming for 3D model-based vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 371–381, 1983.

[55] T. Goedeme, T. Tuytelaars, L. Van Gool, J. Sivic, and A. Zisserman. *Cognitive Vision Systems, EC Project Final Report, IST-2000-29404*, chapter Location and Object Matching and Discovery in Video. In Press, 2005.

[56] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings of the International Conference on Computer Vision*, pages I:357–364, October 2005.

[57] T. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, 2004.

[58] 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.

[59] 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.

[60] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[61] B. Heisele, P. Ho, J. Wu, and T. Poggio. Face recognition: component–based versus global approaches. *Computer Vision and Image Understanding*, 91(1–2):6–21, 2003.

[62] K. Ho and P. Newman. Multiple map intersection detection using visual appearance. In *3rd International Conference on Computational Intelligence, Robotics and Autonomous Systems*, 2005.

[63] T. Hofmann. The cluster-abstraction model: Unsupervised learning of topic hierarchies from text data. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 682–687, 1999.

[64] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

[65] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 43:177–196, 2001.

[66] T. Hofmann and J. Puzicha. Statistical models for co-occurrence data. Technical Report AIM-1625, MIT, 1998.

[67] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[68] D. W. Jacobs. Linear fitting with missing data: applications to structure-from-motion and to characterizing intensity images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–212, 1997.

[69] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of the International Conference on Computer Vision*, pages I:604–610, 2005.

[70] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, pages I:228–241. Springer-Verlag, May 2004.

[71] J. R. Kender and B. L. Yeo. Video scene segmentation via continuous video coherence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 367–373, 1998.

[72] D. Klein, S. Kamvar, and C. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *In Proceedings of the International Conference on Machine Learning*, pages 307–314, 2002.

[73] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[74] V. Krueger and S. Zhou. Examplar-based face recognition from video. In *Proceedings of the European Conference on Computer Vision*, pages IV:732–746, 2002.

[75] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Extending pictorial structures for object recognition. In *Proceedings of the British Machine Vision Conference*, 2004.

[76] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, New York, NY, USA, 2001. ACM Press.

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

[78] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using affine-invariant regions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 319–324, June 2003.

[79] 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.

[80] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:822–828, 2005.

[81] T. Leung. Texton correlation for recognition. In *Proceedings of the European Conference on Computer Vision*, pages I:203–214, 2004.

[82] T. Leung and J. Malik. Recognizing surfaces using three-dimensional textons. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 1010–1017, Kerkyra, Greece, September 1999.

[83] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, June 2001.

[84] T. Lindeberg and J. Gårding. Shape-adapted smoothing in estimation of 3-d depth cues from affine distortions of local 2-d brightness structure. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, LNCS 800, pages 389–400, May 1994.

[85] 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.

[86] 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, September 2001.

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

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

[89] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[90] A. Mahindroo, B. Bose, S. Chaudhury, and G. Harit. Enhanced video representation using objects. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, pages 105–112, 2002.

[91] 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.

[92] 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 I:525–531, 2001.

[93] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, pages I:128–142. Springer-Verlag, 2002.

[94] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:257–263, 2003.

[95] 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*, pages I:69–82. Springer-Verlag, May 2004.

[96] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1/2):43–72, 2005.

[97] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. In *Proceedings of the British Machine Vision Conference*, 2003.

[98] H. Murase and S. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision*, 14(1), 1995.

[99] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[100] S. Obdrzalek and J. Matas. Local affine frames for image retrieval. In *Proceedings of the International Conference on Image and Video Retrieval*, volume 1, pages 318–327, 2002.

[101] S. Obdrzalek and J. Matas. Sub-linear indexing for large scale object recognition. In *Proceedings of the British Machine Vision Conference*, 2005.

[102] P. Ogilvie and J. Callan. Language models and structured document retrieval. In *In the Proceedings of the Initiative for the Evaluation of XML Retrieval Workshop (INEX 2002)*, 2002.

[103] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[104] A. Opelt, A. Fussenegger, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *Proceedings of the European Conference on Computer Vision*, 2004.

[105] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *Proceedings of the European Conference on Computer Vision*, 2006.

[106] M. Osian and L. Van Gool. Video shot characterization. *Machine Vision and Applications Journal*, 15(3):172–177, 2004.

[107] M. Pollefeys, R. Koch, and L. Van Gool. Self calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, pages 90–96, 1998.

[108] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C (2nd Ed.)*. Cambridge University Press, 1992.

[109] P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *Proceedings of the International Conference on Computer Vision*, pages I:883–890, 2005.

[110] D. Ramanan, D. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1:271–278, 2005.

[111] L. G. Roberts. Machine perception of three-dimensional solids. Technical Report 315, Lincoln Laboratory, 1963.

[112] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D object modeling and recognition using local affine-invariant descriptors and multi-view spatial constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 272–280, 2003.

[113] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 914–921, 2004.

[114] C. Rothwell, D. Forsyth, A. Zisserman, and J. Mundy. Extracting projective structure from single perspective views of 3D point sets. In *Proceedings of the International Conference on Computer Vision*, pages 573–582, 1993.

[115] C. Rothwell, A. Zisserman, D. Forsyth, and J. Mundy. Planar object recognition using projective shape representation. *International Journal of Computer Vision*, 16(2), 1995.

[116] Y. Rubner, C. Tomasi, and L.J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[117] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006. To appear.

[118] S. Satoh, Y. Nakamura, and T. Kanade. Name-It: Naming and detecting faces in news videos. *IEEE Multimedia*, 6(1):22–35, 1999.

[119] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 636–643, July 2001.

[120] F. Schaffalitzky and A. Zisserman. Automated scene matching in movies. In *Proceedings of the International Conference on Image and Video Retrieval, London*, LNCS 2383, pages 186–197. Springer-Verlag, 2002.

[121] 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.

[122] F. Schaffalitzky and A. Zisserman. Automated location matching in movies. *Computer Vision and Image Understanding*, 92:236–264, 2003.

[123] 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.

[124] C. Schmid. *Appariement d'Images par Invariants Locaux de Niveaux de Gris*. PhD thesis, L'Institut National Polytechnique de Grenoble, Grenoble, 1997.

[125] 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.

[126] C. Schmid and A. Zisserman. Automatic line matching across views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 666–671, 1997.

[127] 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*, 2000.

[128] G. Shakhnarovich, J. Fisher, and T. Darrel. Face recognition from long-term observations. In *Proceedings of the European Conference on Computer Vision*, pages III:851–868, 2002.

[129] G. Shakhnarovich and B. Moghaddam. Face recognition in subspaces. In S.Z. Li and A.K. Jain, editors, *Handbook of face recognition*. Springer, 2004.

[130] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–743, 1997.

[131] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *Proceedings of the International Conference on Computer Vision*, pages I:503–510, 2005.

[132] H.-Y. Shum, I. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9):855–867, 1995.

[133] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval (CIVR 2005), Singapore*, pages 226–236, 2005.

[134] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *Proceedings of the International Conference on Computer Vision*, pages 370–377, 2005.

[135] J. Sivic, F. Schaffalitzky, and A. Zisserman. Efficient object retrieval from videos. In *Proceedings of the 12th European Signal Processing Conference (EUSIPCO '04), Vienna, Austria*, September 2004.

[136] J. Sivic, F. Schaffalitzky, and A. Zisserman. Object level grouping for video shots. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, volume 2, pages 85–98. Springer-Verlag, May 2004.

[137] J. Sivic, F. Schaffalitzky, and A. Zisserman. Object level grouping for video shots. *International Journal of Computer Vision*, 67(2):189–210, 2006.

[138] 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 II:1470–1477, October 2003.

[139] J. Sivic and A. Zisserman. Efficient visual content retrieval and mining in videos. In *Pacific-Rim Conference on Multimedia, (PCM 2004), Tokyo, Japan*, November 2004.

[140] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:488–495, 2004.

[141] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[142] J. R. Smith, M. R. Naphade, A. Natsev, and J. Tesic. Multimedia research challenges for industry. In *Proceedings of the International Conference on Image and Video Retrieval*, pages 28–37, 2005.

[143] C. Snoek, M. Worring, J.-M. Geusebroek, D. Koelma, J. Seinstra, and A. Smeulders. The semantic pathfinder: Using an authoring metaphor for generic multimedia indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006. To appear.

[144] G. Stockman. Object recognition and localization via pose clustering. *Computer Vision, Graphics and Image Processing*, 40:361–387, 1987.

[145] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed dirichlet processes. In *Advances in Neural Information Processing Systems*, 2005.

[146] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Proceedings of the International Conference on Computer Vision*, pages II:1331–1338, 2005.

[147] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. In *Proc. NIPS*, 2004.

[148] D. W. Thompson and J. L. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *Proceedings of the International Conference on Robotics and Automation*, pages 208–220, 1987.

[149] P. H. S. Torr. *Motion segmentation and outlier detection*. PhD thesis, Dept. of Engineering Science, University of Oxford, 1995.

[150] P. H. S. Torr, A. W. Fitzgibbon, and A. Zisserman. The problem of degeneracy in structure and motion recovery from uncalibrated image sequences. *International Journal of Computer Vision*, 32(1):27–44, August 1999.

[151] P. H. S. Torr, R. Szeliski, and P. Anadan. An integrated bayesian approach to layer extraction from image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):297–304, 2001.

[152] P. H. S. Torr, A. Zisserman, and S. Maybank. Robust detection of degenerate configurations for the fundamental matrix. *Computer Vision and Image Understanding*, 71(3):312–333, September 1998.

[153] A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. In *Advances in Neural Information Processing Systems*, 2004.

[154] 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*, volume 2, pages 762–769, Washington, DC, June 2004.

[155] M. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[156] T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proceedings of the 11th British Machine Vision Conference, Bristol*, pages 412–425, 2000.

[157] `http://images.google.com/`.

[158] `http://video.google.com/`.

[159] `http://www-nlpir.nist.gov/projects/trecvid/`.

[160] `http://www.google.com/`.

[161] `http://www.pascal-network.org/challenges/VOC/databases.html#MIT`.

[162] `http://www.pascal-network.org/challenges/VOC/voc2006/index.html`.

[163] L. Van Gool, T. Tuytelaars, and A. Turina. Local features for image retrieval. In H. Burkhardt, H.-P. Kriegel, and R. Veltkamp, editors, *State-of-the-Art in Content-Based Video and Image Retrieval*, chapter 2, pages 21–41. Kluwer Publishers, 2001.

[164] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 3, pages 255–271. Springer-Verlag, May 2002.

[165] M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 691–698, June 2003.

[166] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1–2):61–81, April 2005.

[167] M. Varma and A. Zisserman. Unifying statistical texture classification frameworks. *Image and Vision Computing*, 22(14):1175–1183, 2005.

[168] 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*, 2001.

[169] C. Wallraven and H. Bulthoff. Automatic acquisition of exemplar-based representations for recognition from image sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Models vs. Exemplars*, 2001.

[170] J. Wang, H. Zha, and R. Cippola. Coarse-to-fine vision-based localization by indexing scale-invariant features. *IEEE Transactions on systems, man, and cybernetics*, 36(2):413–423, 2006.

[171] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *Proceedings of the European Conference on Computer Vision*, pages 18–32, 2000.

[172] J. Winn. *Variational Message Passing and its Applications*. PhD thesis, University of Cambridge, 2003.

[173] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the International Conference on Computer Vision*, pages I:756–763, 2005.

[174] L. Wiskott, J. Fellous, N. Krueger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779, 1997.

[175] I. H. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, ISBN:1558605703, 1999.

[176] H. J. Wolfsonm and I. Rigoutsos. Geometric hashing: An overview. *IEEE Comput. Sci. Eng.*, 4(4):10–21, 1997.

[177] L. Zelnik-Manor and M. Irani. Multi-view subspace constraints on homographies. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, volume 2, pages 710–715, 1999.

[178] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classifcation of texture and object categories: An in-depth study. Technical Report RR-5737, INRIA Rhône-Alpes, Nov 2005.