AN ABSTRACT OF THE DISSERTATION OF

Wei Zhang for the degree of Doctor of Philosophy in Electrical and Computer Engineering on March 16, 2009.

Title: Image Features and Learning Algorithms for Biological, Generic and Social Object Recognition.

Abstract approved:

Thomas G. Dietterich

Automated recognition of object categories in images is a critical step for many real-world computer vision applications. Interest region detectors and region descriptors have been widely employed to tackle the variability of objects in pose, scale, lighting, texture, color, and so on. Different types of object recognition problems usually require different image features and corresponding learning algorithms. This dissertation focuses on the design, evaluation and application of new image features and learning algorithms for the recognition of biological, generic and social objects. The first part of the dissertation introduces a new structure-based interest region detector called the principal curvature-based region detector (PCBR) which detects stable watershed regions that are robust to local intensity perturbations. This detector is specifically designed for region detection for biological objects. Several recognition architectures are then developed that fuse visual information from disparate types of image features for the categorization of complex objects. The described image features and learning algorithms achieve excellent performance on the difficult stonefly larvae dataset. The second part of the dissertation presents studies of methods for visual codebook learning and their application to object recognition. The dissertation first introduces the methodology and application of generative visual codebooks for stonefly recognition and introduces a discriminative evaluation methodology based on a maximum mutual information criterion. Then a new generative/discriminative visual codebook learning algorithm, called iterative discriminative clustering (IDC), is presented that refines the centers and the shapes of the generative codewords for improved discriminative power. It is followed by a novel codebook learning algorithm that builds multiple codebooks that are non-redundant in discriminative power. All these visual codebook learning algorithms achieve high performance on both biological and generic object recognition tasks. The final part of the dissertation describes a socially-driven clothes recognition system for an intelligent fitting-room system. The dissertation presents the results of a user study to identify the key factors for clothes recognition. It then describes learning algorithms for recognizing these key factors from clothes images using various image features. The clothes recognition system successfully enables automated social fashion information retrieval for an enhanced clothes shopping experience.

Image Features and Learning Algorithms for Biological, Generic and Social Object Recognition

by
Wei Zhang

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented March, 16, 2009
Commencement June 2009

Doctor of Philosophy dissertation of Wei Zhang presented on March, 16, 2009

APPROVED:

Major Professor, representing Electrical and Computer Engineering

Director of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

Wei Zhang, Author

ACKNOWLEDGEMENTS

First, I would like to express my deep and sincere gratitude to my advisor, Professor Thomas Dietterich, for his continuous support in the Ph.D. program. He is the most excellent researcher and the wisest person I ever met. His extensive knowledge and his creative way of thinking have been of great value for me. His understanding, encouraging and guidance have provided the best basis for this dissertation.

I am pleased to thank Professor Eric Mortensen, Professor Sinisa Todorovic in Oregon State University, and Professor Linda Shapiro in University of Washington. They are top researchers in computer vision. Their great expertise benefitted me so much in my Ph.D. study. Thanks also to all the wonderful colleagues in the Bug-ID project. It is my great honor to work with them in this fascinating project. I will always remember the good times we had together.

My thanks go out to the rest of my Ph.D. committee members: Professor Alan Fern, Professor Prasad Tadepalli, Professor Xiaoli Fern, and Professor Yun-Shik Lee. Their kindly guidance and help on my Ph.D. program made everything easy and smooth.

A special thanks goes to Dr. James Begole, Dr. Juan Liu and Dr. Maurice Chu in the Palo Alto Research Center. It was such a pleasure to work with them at the intriguing Responsive Mirror project. I learned so much about industrial R&D, documentation and the patent application process, which are valuable for my future career.

Let me also say ‘thank you’ to all the other colleagues and friends: Hongli Deng, Gonzalo Martínez-Muñoz, Nadia Payet and Natalia Larios. It was very enjoyable to study and work with you.

Finally, I would like to thank my parents. All my achievements would not be possible without their love and support.

CONTRIBUTION OF AUTHORS

Zhang et al. (2006) and Deng et al. (2007): Zhang, W. and Deng, H. proposed the idea of the Principal Curvature-Based Region (PCBR) detector, motivated by the original work of Shapiro, L. Deng, H. implemented the PCBR detector and evaluated the PCBR detector using the repeatability criteria. Zhang, W. contributed ideas for the improvement of the PCBR detector and evaluated the PCBR detector for object recognition applications. Dietterich, T., Mortensen, E. and Shapiro, L. gave constructive suggestions in the development of the PCBR detector and the writing of the manuscripts.

Martínez-Muñoz et al. (2009): Martínez-Muñoz, G. proposed and implemented the stacked decision tree ensembles recognition system with Dietterich, T. Zhang, W. generated the image features for experiments and helped in the visual codebook baseline experiments. Payet, N. and Todorovic, S. contributed the edge features and provided assistance in the generation of the image features for experiments.

Larios et al. (2008): Larios, N. proposed the whole recognition framework with Dietterich, T., contributed the codebook learning algorithm and the logistic model trees classifier. Deng, H. and Zhang, W. contributed the PCBR detector and helped in the experiments. The other authors provided assistance in the development of the system and the revision of the manuscript.

Zhang and Deng (2008): Zhang, W. and Deng, H. proposed the idea of discriminative evaluation of the generative visual codebooks. Zhang, W. proposed and implemented the Maximum Mutual Information (MMI) evaluation criteria. Deng, H. helped in the evaluation experiments.

Zhang and Dietterich (2008): Zhang, W. and Dietterich, T. proposed the idea of discriminative refinement of the generative visual codebooks. Zhang, W implemented and evaluated the Iterative Discriminative Clustering (IDC) codebook learning algorithm and the bagged decision lists classifier.

Zhang et al. (2008a; 2008b; 2008c): Zhang, W., Matsumoto, T. and Begole, B. proposed the concept and architecture of the Responsive Mirror (RM) system, with the help from the other authors. Zhang, W., Chu, M. and Liu, J. collaborated in the implementation of the vision components in the RM system. Zhang, W. developed the clothes recognition and matching engine and evaluated this engine with the help from Yee, N. and Begole, B.

Zhang et al. (2009): Zhang, W., Surve, A. and Fern, X. proposed the idea of non-redundant codebook learning framework. Zhang, W. implemented the Boost-Resampling algorithm and evaluated on the stonefly dataset. Surve, A. and Fern, X. implemented the Boost-Reweighting algorithm and evaluated on the document classification datasets. Dietterich, T. provided assistance in the development of the framework and the revision of the manuscript.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF TABLES (Continued)

Image Features and Learning Algorithms for Biological, Generic and Social Object Recognition

1 Introduction

1.1 Background and Motivation

As one of the most fundamental and active areas in computer vision, object recognition is commonly used as the general term for the problems of automatically categorizing an object in an image into a set of predefined classes. A typical object recognition system is composed of the following elements (Shapiro and Stockman 2001): 1. Low-level image feature extractor: the feature extractor extracts information relevant to classification from the data input by the sensor. Popular feature extractors include interest region detectors (Mikolajczyk et al. 2005a), region descriptors (Mikolajczyk and Schmid 2005), texture analysis, segmentation (Shapiro and Stockman 2001), and so on. 2. High-level classifier: the image classifier uses the features extracted from the sensed object data to assign the object to one of the *m* designated classes (Duda et al. 2001). In many state-of-the-art object recognition approaches, there is an additional "mid-level" process which generates informative and compact image representations based on the extracted low-level features. The image presentations are then input to the high-level image classifier for object categorization. The most popular mid-level method is the visual codebook approach (Csurka et al. 2004 etc.).

Although it is quite natural for humans to recognize thousands of object categories, object recognition continues to be a challenge for machine vision systems. The difficulty mainly lies in two aspects (see Figures 1−5): (1) large appearance variance in objects due to intrinsic variations, different poses, visual transformations, occlusions, noise, and so on; and (2) Inter-class similarity caused by some visual patterns shared by different classes.

Interest region detectors and region descriptors have been widely employed to tackle these difficulties. An interest region (point) detector detects the regions that are distinctive and invariant to image transformations (see Figure 6 and Figure 7). And for each interest region, a descriptor vector is computed that describes the color, textural or structural information within the region. These feature extractors are advantageous in their robustness to image variations and background clutter (see Section 2.2). They transform an original image into a bag of region descriptor vectors, e.g., bag-of-keypoints (Csurka et al. 2004). Thus the object recognition problem reduces to a multiple-instance classification problem (Dietterich et al. 1997) that classifies the bag-of-keypoints image representation into one of the *m* designated classes. Various learning algorithms have been proposed to train image classifiers based on bags of keypoints, including visual codebook approaches (See Section 2.3) and discriminative feature selection methods (see Section 2.4), and so on. These image features and learning algorithms have achieved significant success in object recognition.

In real-world object recognition tasks, the visual properties of the objects and the settings of the recognition problems may vary significantly in different domains. This dissertation focuses on the recognition of three types of objects: biological objects, generic objects and social objects. Specific challenges arise in each of these domains, as introduced below.

Biological object recognition

Population counts of insects that live in soils, lakes, streams and ocean are good measurements to the health of these ecosystems. But these measurements are usually not easily accessible in current environmental science applications. Manual insect classification is very difficult and time-consuming. It requires a great deal of domain knowledge which is only possessed by a few highly-experienced entomologists worldwide. The OSU “Bug-ID” project (*BugID*) employs cost-effective computer vision and machine learning methods for automated rapid-throughput image capture, recognition and sorting of insects. This dissertation presents work on a specific problem in this project: the automated recognition of stonefly larvae, which are known to be a sensitive indicator of freshwater stream health. Example images of stoneflies are shown in Figure 1.

Automated recognition of stoneflies is extremely challenging. First, there are large intra-class variations in the images. Geometric and photometric properties of the larvae change significantly with their age. The stonefly larvae consist of bloblike parts, very elongated legs, and wiry antennae. Bloblike insect parts may produce specularities, which in turn may be easily confused with the insect's canonical photometric properties. Due to the multiple legs and antennae, the insects are highly articulated, and thus appear in different poses in the images. As an additional challenge, the image acquisition system cannot completely immobilize the insects while taking pictures. As a result, the insects are imaged from non-perfect dorsal views. Second, the differences between categories can be very small. For example, the appearance of two categories of stoneflies: Cal and Dor (shown in the top two rows in Figure 1) are so similar to each other that sometimes they are not distinguishable by human eyes. An experiment (Larios et al. 2008) showed that the biology students and faculty in the Oregon State insect ID group, who have been trained to recognize images of these two categories, achieve only 78.6% classification accuracy on images of these two categories. All these challenges make categorizing stonefly images an extremely difficult task.

Generic object recognition

Recognizing the generic categories of objects (e.g., cars, bikes, etc.) in images of natural scenes is a fundamental problem in computer vision. Benchmark object recognition datasets such as the Caltech (*Caltech*, see Figure 2), GRAZ (*GRAZ*, see Figure 3) and PASCAL datasets (*PASCAL06*, see Figure 4) have been collected and evaluated extensively by different research groups. These generic object recognition problems are quite challenging. First, the images in these datasets are photos of the objects taken in uncontrolled natural scenes; they usually contain very complex background clutter. The objects within a single class vary significantly in their size, pose, color, texture, and so on. Thus the recognition system needs to be robust to background noise and invariant to image transformations. Second, oftentimes only a limited number of labeled training images are provided in these datasets. Therefore the learning algorithm is required to have low risk of overfitting.

Social object recognition

An intelligent fitting-room system called the "Responsive Mirror (RM)" (Zhang et al. 2008b; Begole et al. 2008) has been developed. The RM system enables online shopping capabilities during the in-store shopping experience using computer vision and machine learning techniques. The "social" comparison component in the RM system automatically retrieves the clothes that are most "similar" and "different" to the query clothes from the image dataset, as shown in Figure 5. A new social vision problem – socially-driven clothes recognition is defined in order to realize this function. Clothes recognition is difficult in a number of ways: the social nature of the problem definition; the real-time requirement; the low resolution of the images; the high intra-class variation in clothes images; the deformable configurations of the clothes, and so on. An additional challenge is the small number of clothes images available for system development due to the high cost of data collection. Similar to generic object recognition problems, the recognition system needs to be robust to overfitting.

In summary, objects in different domains usually require different image features and learning algorithms to achieve satisfactory recognition performance. This dissertation focuses on the design, evaluation and application of new image features and learning algorithms for the recognition of biological, generic and social objects. The purpose is to develop novel object recognition methods that not only perform well on the specific object classes, but also generalize to other tasks of similar nature.

1.2 Contributions

The contributions of this dissertation are outlined as follows:

1. This dissertation presents a novel structure-based semi-local interest region detector – the PCBR detector – for the extraction of salient regions from non-rigid biological objects (Zhang et al. 2006; Deng et al. 2007). PCBR detects the regions that have a coherent interior and that are surrounded by distinctive curvilinear structure. PCBR complements previous intensity-based detectors and shows high robustness to local intensity perturbation and intra-class variation. The PCBR detector achieves excellent performance on various object recognition tasks.

2. This dissertation describes new object recognition methods (Zhang et al. 2006; Martínez-Muñoz et al. 2009) that fuse disparate types of low-level image features for accurate recognition of complex objects. The image features generated by different combinations of interest region detectors and descriptors are selected via supervised learning to explore the most discriminative object patterns for recognition. These methods are general frameworks that can be applied to any object recognition tasks using bag-of-keypoints image representations. All of these methods achieve high performance on the challenging stonefly recognition dataset and generic object recognition datasets.

3. The success of an object recognition system using bag-of-keypoints image representations highly relies on proper generalization of these image features. This dissertation studies a simple and efficient method of feature generalization – the visual codebook. Generative visual codebooks (Larios et al. 2008) are successfully adapted to the challenging stonefly recognition problem by building a separate codebook for each type of image feature and then concatenating the image attribute vectors generated from the codebooks. The Maximum Mutual Information (MMI) evaluation criterion (Zhang

and Deng 2008) is presented to measure the discriminative power of generative visual codebooks built from different combinations of region detectors and descriptors. We also present a novel generative/discriminative visual codebook learning algorithm (Zhang and Dietterich 2008) that refines the centers and shapes of the codebook entries using class label supervision for improved discriminative power. Finally, a new non-redundant visual codebook learning algorithm (Zhang et al. 2009) is developed which builds multiple codebooks that are non-redundant in discriminative power. All these new visual codebook learning algorithms advance the research in this direction. They also give high performance on real-world biological and generic object recognition datasets.

4. Finally, this dissertation studies a class of largely unexplored object recognition problems: the recognition of objects with social meaning. It focuses on a specific problem in this class – clothes recognition – and presents a solution to this difficult problem (Zhang et al. 2008a; Zhang et al. 2008b; Zhang et al. 2008c). A user study is first conducted to identify the most important clothes factors human eyes perceive in term of clothing similarity. Then these factors are automatically recognized in clothes images by learning classifiers over the extracted low-level image features. This clothes recognition system achieves high performance on a simulated test dataset; it also successfully enables social clothes retrieval and comparison in physical stores. Our research not only explores the application of image features and learning algorithms in intelligent computer-human interaction, but also benefits future research on the automated visual recognition of other social objects.

1.3 Dissertation Outline

The dissertation starts with a literature review (Chapter 2) of state-of-the-art interest region detectors and region descriptors, visual codebook learning algorithms and recognition methods based on discriminative image feature selection. Chapter 3 presents the new PCBR detector and recognition architectures developed for the stonefly recognition problem. Chapter 4 describes the new unsupervised and supervised visual codebook learning algorithms and their application in real-world object recognition tasks. Then this dissertation moves to the socially-driven clothes recognition problem in Chapter 5 and presents an efficient clothes recognition system for intelligent clothes recommendation in fitting room. This dissertation concludes in Chapter 6 with proposals for future research directions.

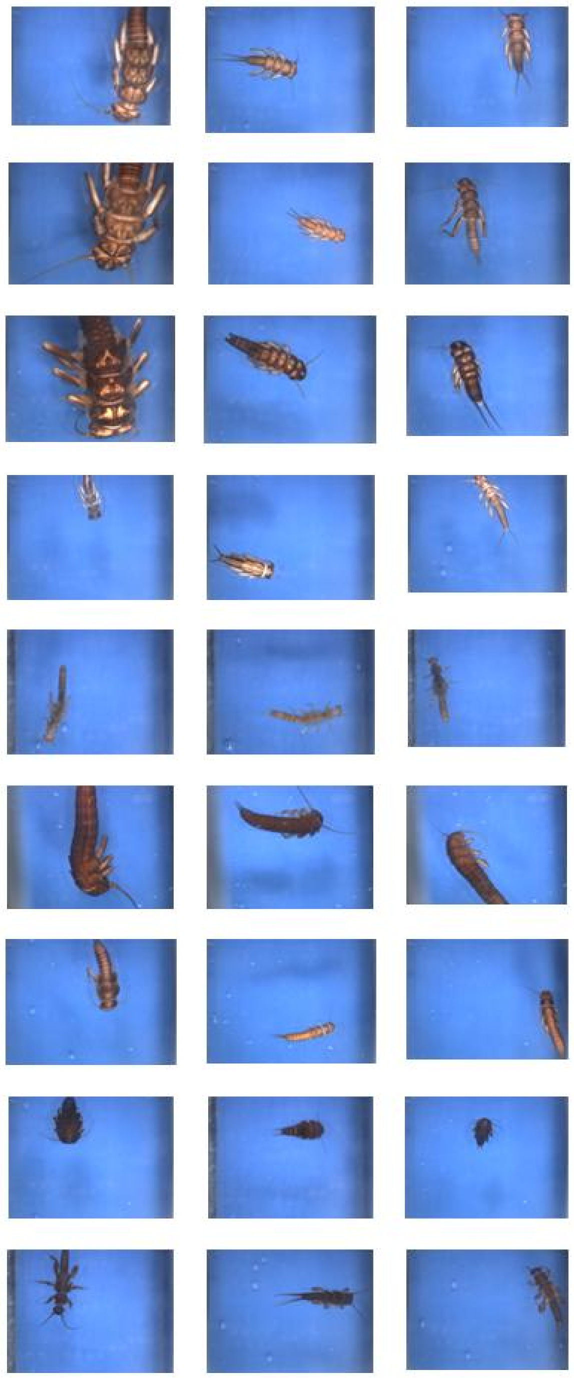


Figure 1. Example images in stonefly larvae, with rows corresponding to the nine categories: Cal, Dor, Hes, Iso, Mos, Pte, Swe, Yor and Zap.

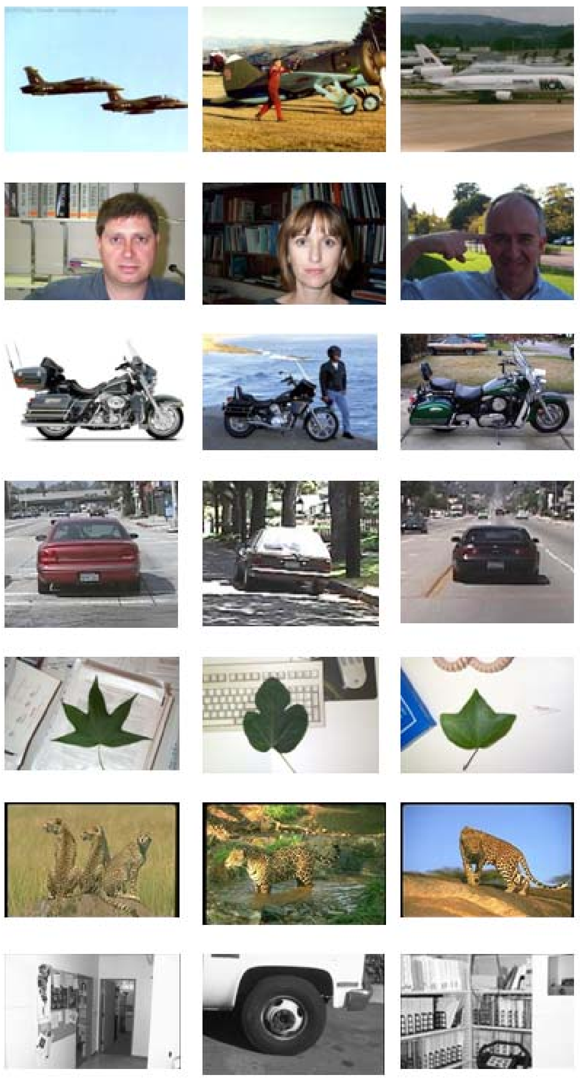


Figure 2. Example images in Caltech dataset with rows corresponding to different categories: airplanes, faces, motorbikes, cars, leaves, leopards, and background scenes.

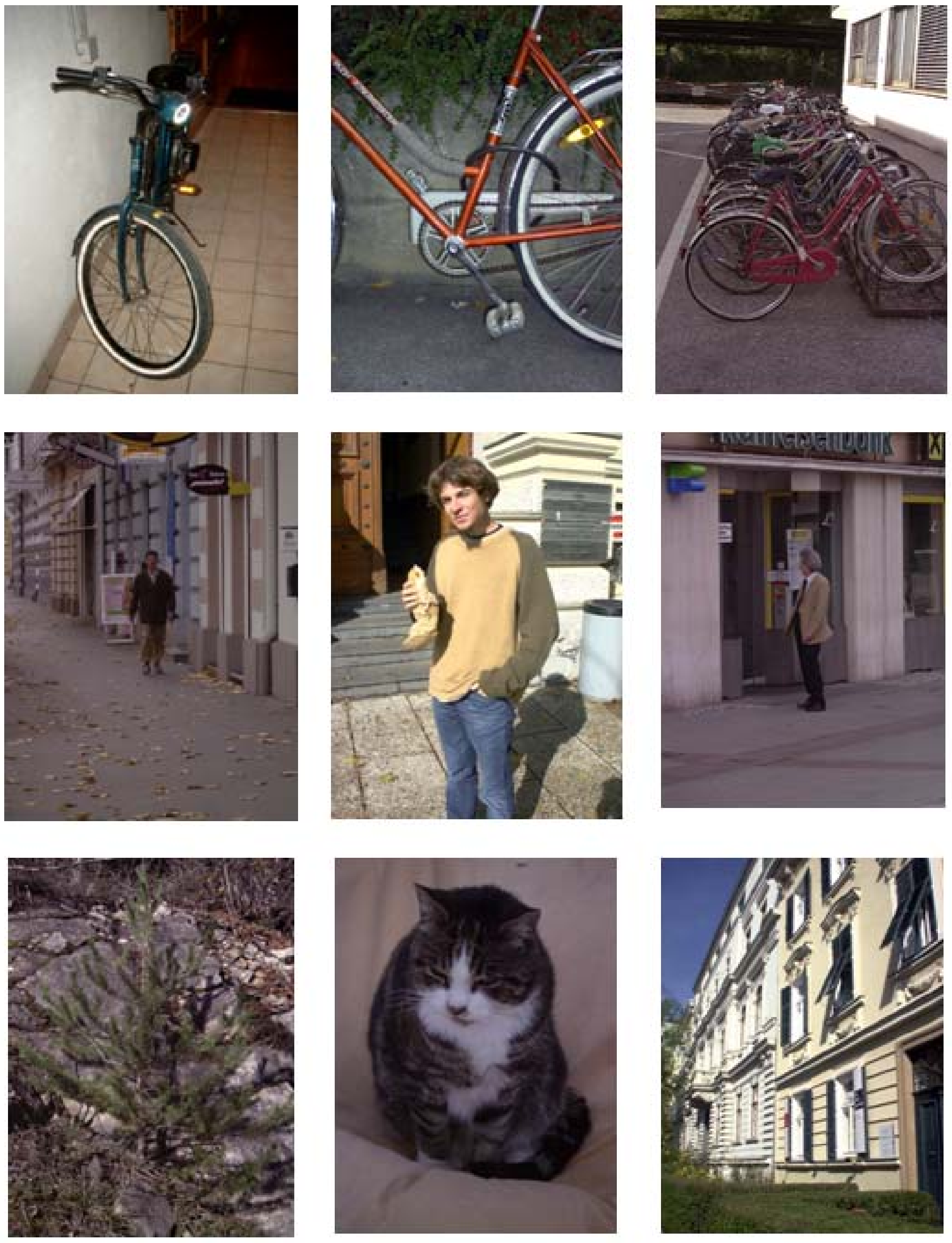


Figure 3. Example images in GRAZ dataset with rows corresponding to bikes, persons, and background categories.

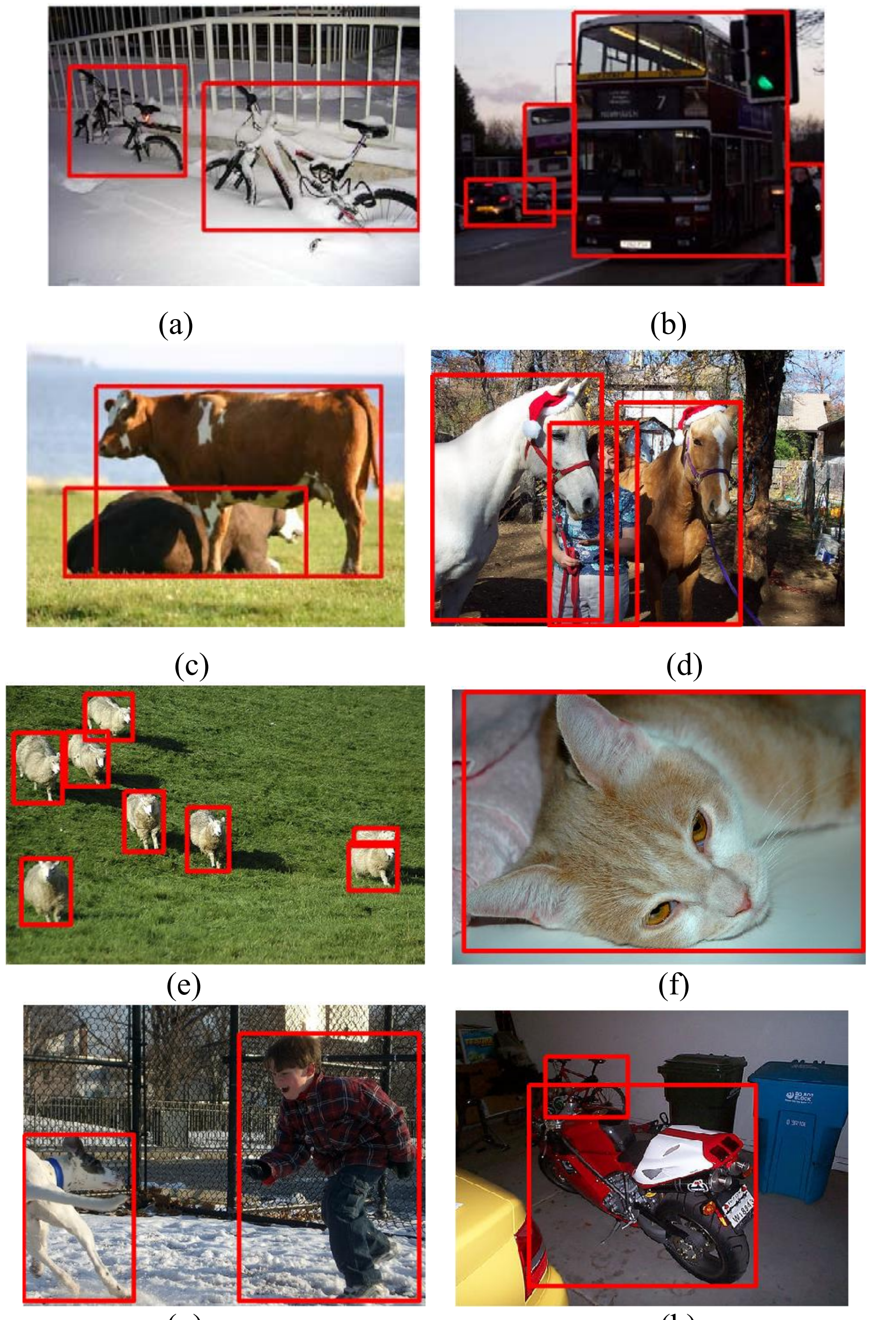


Figure 4. Example images in PASCAL 2006 dataset containing objects from different categories: (a) bikes, (b) cars, buses and person, (c) cows, (d) horses and person, (e) sheeps, (f) cat, (g) dog and person, (h) bike and motorbike.

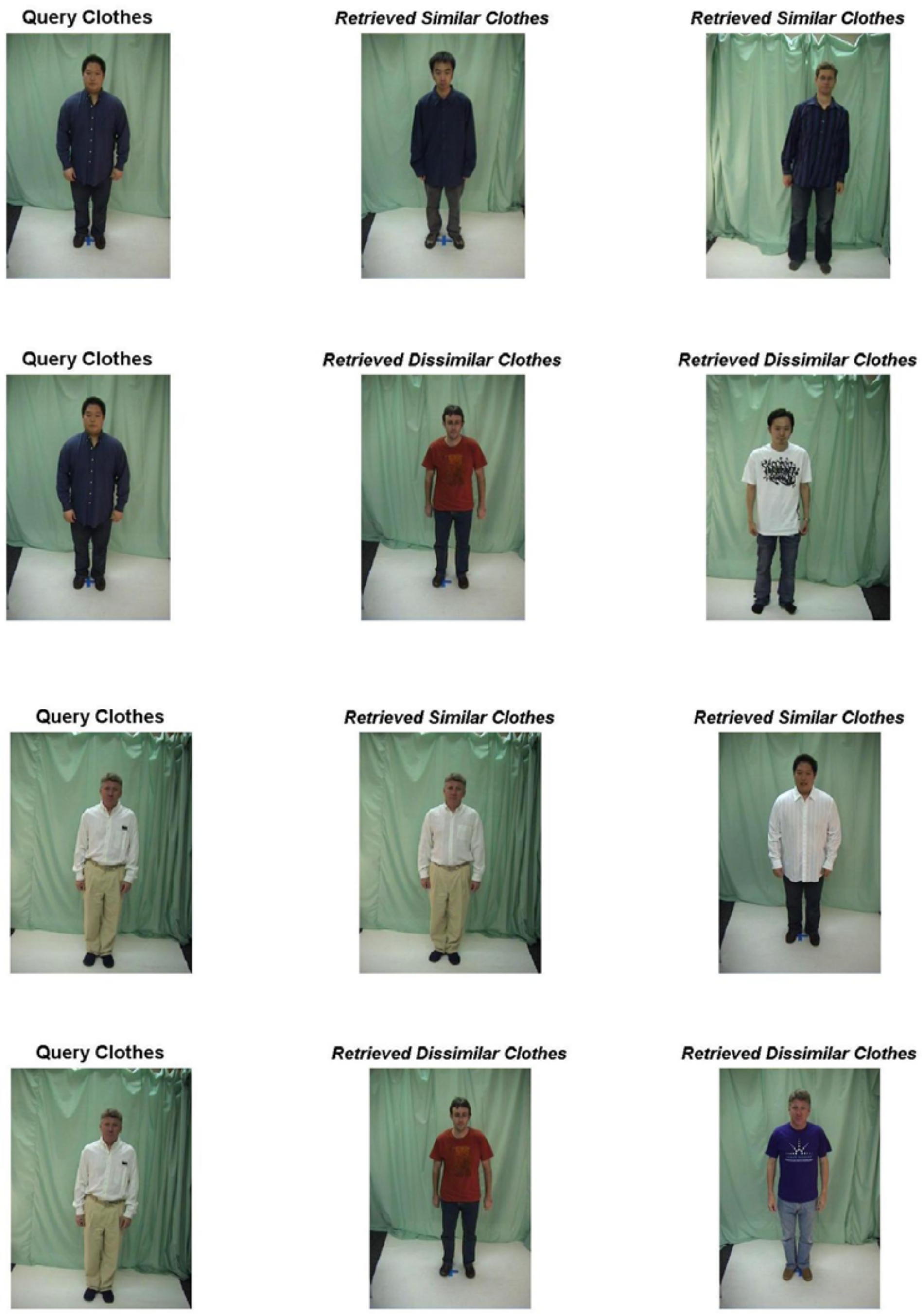

Query Clothes
Retrieved Similar Clothes
Retrieved Similar Clothes
Query Clothes
Retrieved Dissimilar Clothes
Retrieved Dissimilar Clothes
Query Clothes
Retrieved Similar Clothes
Retrieved Similar Clothes
Query Clothes
Retrieved Dissimilar Clothes
Retrieved Dissimilar Clothes

Figure 5. Illustration of clothes image retrieval using socially-driven clothes recognition.

2 Literature Review

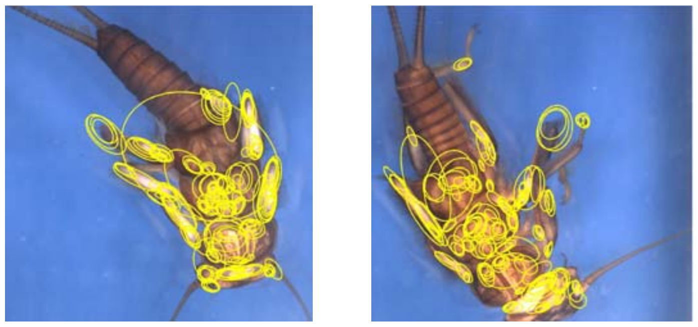
2.1 Overview

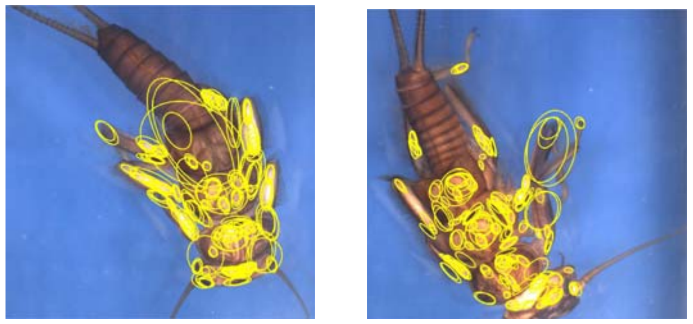
As introduced in Chapter 1, object recognition has long been an active area in computer vision and machine learning. A complete object recognition system is usually composed of a low-level image feature extractor, a high-level image classifier, and often an intermediate level of image representation.

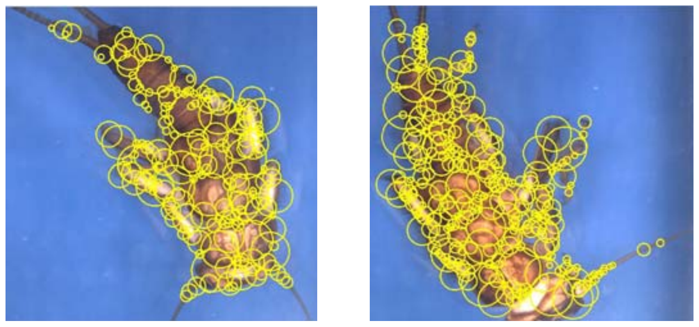
Various image features, image representations and image classifiers have been proposed for the recognition of biological, generic and social objects. This section briefly reviews and summarizes the related literature to give the background and motivation of the approaches presented in this dissertation. Section 2.2 describes the state-of-the-art low-level image feature extractors − interest region detectors and region descriptors − along with their performance evaluations. Section 2.3 presents the object recognition framework using a specific mid-level representation – the visual codebook. Both mid-level image representations (Sections 2.3.2, 2.3.3 and 2.3.4) and high-level image classification algorithms (Section 2.3.5) based on the image representations are introduced. Section 2.4 presents several methods that select a small number of most discriminative low-level image features in a new image for recognition. These methods are highly related to the classification algorithms introduced in Chapter 3 and Chapter 4.

2.2 Interest Region Detectors, Region Descriptors and Performance Evaluation

2.2.1 Interest Region Detectors


(a)


(b)


(c)

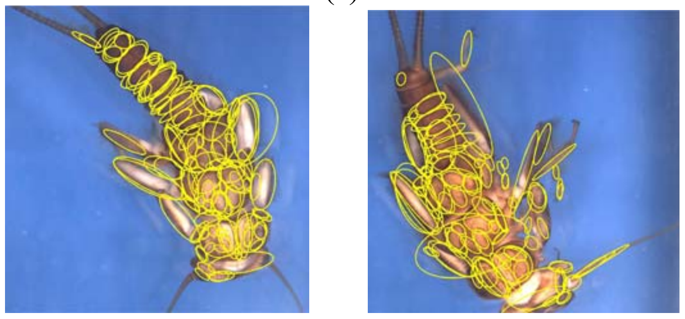

(d)

Figure 6. Examples of regions detected on stonefly images by: (a) Harris-Affine, (b) Hessian-Affine, (c) Kadir's salient detectors and (d) PCBR.

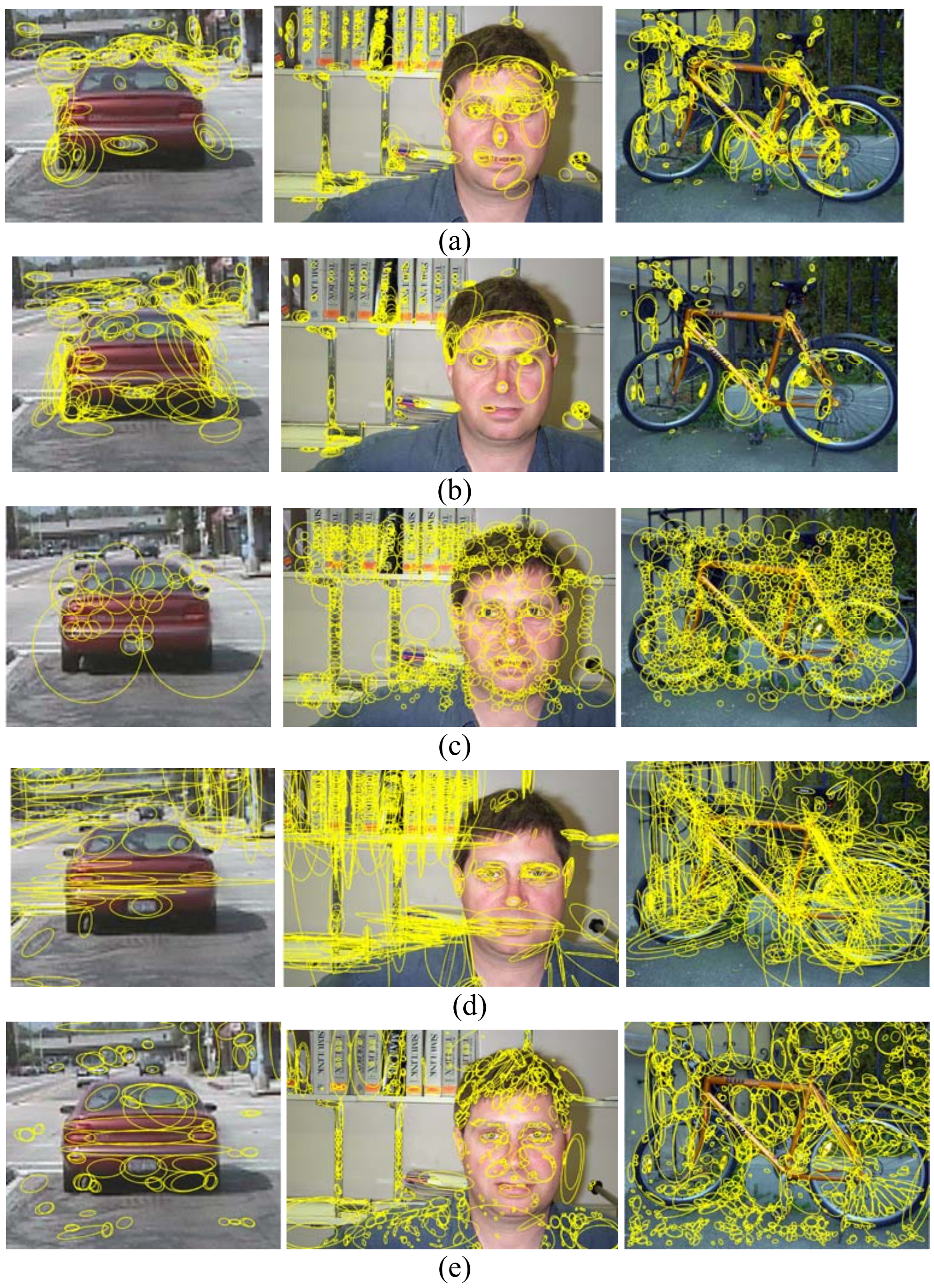


Figure 7. Examples of regions detected on generic objects by: (a) Harris-Affine, (b) Hessian-Affine, (c) Kadir's salient, (d) MSER detectors and (e) PCBR.

Real-world object recognition datasets usually contain significant appearance variation in images due to different pose, visual transformation, occlusion, noise signals, and so on, as shown in Figure 1−5. In order to obtain relatively invariant and compact representations of objects, various interest region detectors have been applied to images to extract "salient" patterns that are distinctive and repeatable. Some region detectors identify specific visual attributes based on image intensity, such as corners or blobs, or areas exhibiting a significant amount of complexity. Some other detectors identify regions based on the shape or structure information in images. Examples of regions detected on stoneflies and generic objects are shown in Figure 6 and Figure 7.

Intensity-based region detectors

Mikolajczyk and Schmid (2004) proposed two related detectors which detect interest points in scale-space, and then determine an elliptical region for each point. Interest points are either detected with the Harris detector (Harris-Affine) or with a detector based on the Hessian matrix (Hessian-Affine). The Harris detector usually finds corner points, while the Hessian detector focuses on blobs and ridges. In both cases scale selection is based on the Laplacian, and the shape of the elliptical region is determined with the second moment matrix of the intensity gradient. Examples of Harris-Affine and Hessian-Affine detections are shown in Figure 6 (a) (b) and Figure 7 (a) (b).

Lowe (2004) proposed the Difference-of-Gaussians (DoG) detector: The image is first convolved with Gaussian filters at different scales, and then the difference of successive Gaussian-blurred images is computed. Keypoints are then taken as maxima/minima of the Difference of Gaussians (DoG) that occur at multiple scales. The DoG detector also detects blob and ridge regions.

Instead of detecting specific visual patterns, the salient region detector in Kadir et al. (2004) identifies the regions exhibiting a significant amount of complexity based on the probability distribution function of the intensity values computed over the region. Examples of Kadir's salient regions are shown in Figure 6 (c) and Figure 7 (c).

The Intensity Extrema Region (IBR) detector by Tuytelaars and Van Gool (2004) starts with intensity extrema at multiple scales and defines the surrounding region as the extrema in intensity changes along rays emanating from the detected point. The Maximally Stable Extremal Region (MSER) by Matas et al. (2002) is another intensity-based detector commonly-used for image matching and recognition. An MSER region is a connected component of an appropriately thresholded image. The word 'extremal' refers to the property that all pixels inside the MSER have either higher (bright extremal regions) or lower (dark extremal regions) intensity than all the pixels on its outer boundary. Examples of MSER regions are shown in Figure 7 (d).

Structure-based region detectors

Intensity-based region detectors prefer to detect patterns like corners and blobs that are often too local and lack discriminative power. Several structure-based detectors have been proposed that rely on the edge or shape information for detection. The edge-based region (EBR) detector by Tuytelaars and Van Gool (2004) fits a parallelogram defined by a Harris corner point and points on two adjacent edge contours (extracted by the Canny edge detector). Scale-invariant shape features (SISF) (Jurie and Schmid 2004) detects circles at different locations and scales by evaluating salient convex arrangements of Canny edges based on a measure of how well a circle is supported by surrounding edges.

In this dissertation, we present a new structure-based region detector – the PCBR detector – that detects semi-local watershed regions within the multi-scale principal curvature images. PCBR regions are robust to image transformations and informative for object categorization. Examples of PCBR detections are shown in Figure 6 (d) and Figure 7 (e). Details of the PCBR detector will be introduced in Section 3.2.

2.2.2 Region Descriptors

As introduced in Chapter 1, for each region detected by the interest region detector, a descriptor vector is affiliated that describes the pixel intensities, color, texture, or edge information within the region. There are many possible descriptors that emphasize different image properties.

Agarwal et al. (2004) uses 13×13 patches extracted around each interest point as the region descriptor. Viola and Jones (2001) employed rectangular Haar-like features for face detection and proposed a very efficient algorithm to compute them using the integral image.

Differential descriptors have also been employed to represent interest regions due to their representational power and computational efficiency. Freeman and Adelson (1991) developed steerable filters, which use derivatives computed by convolution with Gaussian derivatives of $\sigma = 6.7$ for an image patch of size 41. The filter bank descriptor in Shotton et al. (2008) convolves the image with a 17-dimensional filter-bank at scale κ.

The most successful region descriptor up to now is the scale invariant feature transform (SIFT) descriptor proposed by Lowe (2004), which is based on the gradient distribution in the detected regions. The descriptor is represented by a 3D histogram of gradient locations and orientations. The contribution to the location and orientation bins is weighted by the gradient magnitude. The quantization of gradient locations and orientations makes the descriptor robust to small geometric distortions and small errors in the region detection. SIFT descriptors are employed in most of the experiments in this dissertation. Several extensions to SIFT descriptor have been proposed including the gradient location and orientation histogram (GLOH) descriptor by Mikolajczyk and Schmid (2005), the PCA-SIFT descriptor by Ke and Sukthankar (2004), and the Color-SIFT descriptors in Van de Sande et al. (2008). Belongie et al. (2002) employed the SIFT idea and proposed the shape context descriptor that computes a histogram describing the edge distribution in a region. These SIFT-style descriptors have been successfully applied to various image matching and object recognition applications.

2.2.3 Performance Evaluation

With the popularity of interest region detectors and descriptors, it is widely agreed that the evaluation of detectors and descriptors is important. Early work on the evaluation of region detectors was based on ground-truth verification, visual inspection, localization accuracy, theoretical analysis and performance on specific tasks. More general evaluation work was done by the INRIA group. In Schmid et al. (2000), two criteria for evaluating interest region detectors were proposed: repeatability and information content. Repeatability explicitly measures the geometrical stability of the detected interest points between different images of a given scene taken under varying viewing conditions. Information content measures the distinctiveness of the interest regions based on the entropy of the descriptors computed at the regions. Similar criteria were also used in Mikolajczyk et al. (2005a) to evaluate the robustness of affine covariant region detectors to changes in viewpoint, scale, illumination, defocus, and image compression. According to the evaluation results in Mikolajczyk et al. (2005a), MSER obtains the highest score in many tests, followed by Hessian-Affine. Recently, Haja et al. (2008) evaluated state-of-the-art region detectors with regard to their localization accuracy in position and region shape.

Very little work has been done on the evaluation of region descriptors in the context of matching and recognition. The most complete evaluation work up to now is Mikolajczyk and Schmid (2005). The evaluation criterion is recall-precision, i.e. the number of correct and false matches between two images. State-of-the-art region descriptors are evaluated with respect to their robustness to affine transformations, scale changes, image rotation, image blur, JPEG compression and illumination changes. In most of the tests, the GLOH descriptor obtains the best results, closely followed by SIFT.

All the evaluation criteria introduced above were proposed mainly for matching or image registration; they are not appropriate in the context of object recognition. For example, the highly evaluated detectors do not necessarily detect regions valuable for image classification. Some work has been done on the evaluation of low-level image features for object recognition. Dorko and Schmid (2004) evaluate the discriminative power of the clusters of low-level image features using the classification likelihood and mutual information criteria. In Mikolajczyk et al. (2005b), the region features output by different combinations of detectors and descriptors are compared in the context of object class recognition. The clusters of low-level image features are evaluated based on their average cluster precision, location distributions and complementarity. The GLOH descriptor vectors (Mikolajczyk and Schmid 2005) computed on Hessian-Laplace (Mikolajczyk and Schmid 2004) regions perform best according to these evaluation criteria. The second-best score is obtained by Kadir's salient regions (2004). These two detectors also have been shown to provide complementary image features for image classification. Motivated by previous work and the idea of visual codebook approaches, we present the Maximum Mutual Information (MMI) evaluation criterion to measure the discriminative power of generative codebooks built from different combinations of detectors and descriptors. This evaluation method will be introduced in Section 4.3.

2.3 Learning Visual Codebooks

2.3.1 General Framework

As described in Chapter 1, the application of region detector and descriptor transforms the images into bags of region descriptor vectors. Visual codebooks (or vocabularies, dictionaries) are proposed to relate new descriptors found in testing images to the descriptors previously seen in training images. The idea of visual codebooks is motivated by the bag-of-words approach used in text categorization (Salton and McGill 1983). But unlike the text domain, where the codebook is

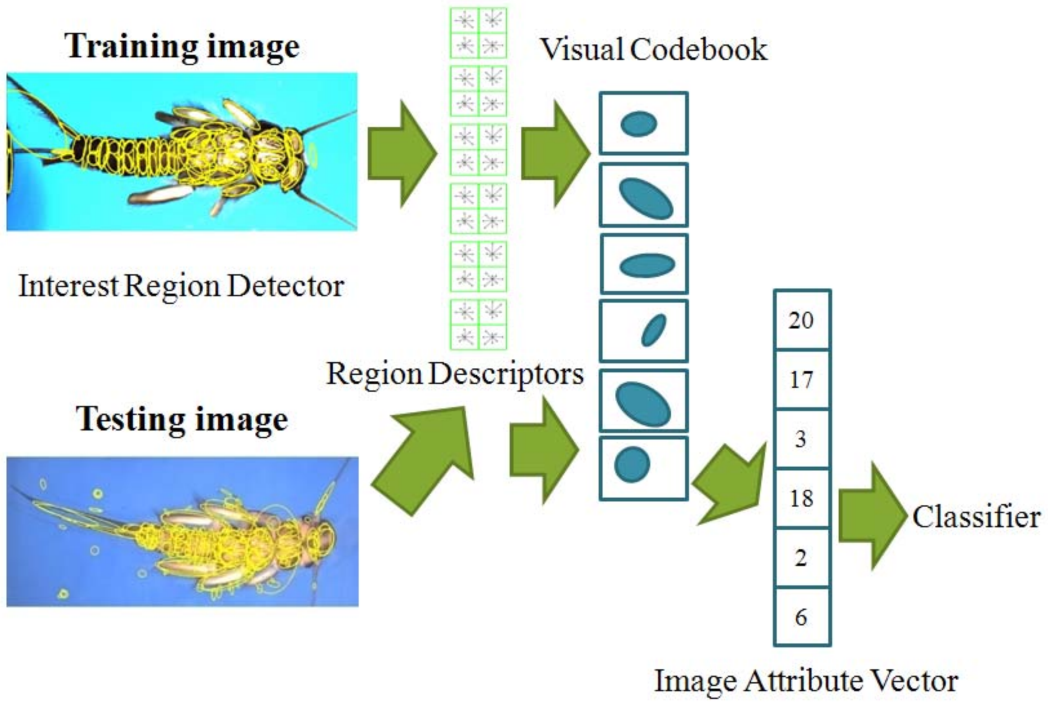


Figure 8. Illustration of the general framework of visual codebook learning approaches.

already given, an object recognition system has to automatically learn the visual codebook from the training data.

The general framework of visual codebook learning approaches is shown in Figure 8. The steps involved in training the system are:

- Detection and description of interest regions, as described in Section 2.2.
- Construction of the visual codebook from the region descriptors extracted from the training images. Both unsupervised learning and supervised learning have been applied, as presented in Sections 2.3.2 and 2.3.3.

- Construction of a fixed-length attribute vector for each training image based on the learned visual codebook. The length of the attribute vector is equal to the size of the codebook. The commonly used construction methods are presented in Section 2.3.4.
- Learning of an image classifier from the attribute vectors that represent the training images. The classification algorithms employed in previous work are summarized in Section 2.3.5.

In the testing stage, a testing image is classified according to the following steps:

- Detection and description of interest regions, same as above.
- Construction of the attribute vector for the testing image based on the visual codebook.
- Application of the image classifier to the attribute vector of the testing image to predict its category.

To handle the significant intra-class variations in the benchmark object recognition datasets, multiple region detectors and region descriptors have been applied on the images to extract different types of visual information for classification. It is natural to fuse visual information using visual codebooks. The standard fusing strategy is to build a separate visual codebook from each detector/descriptor combination; and then either concatenate these visual codebooks into a larger codebook or select the optimal combination of codebooks for the specific problem. For the sake of clarity, this section only describes the methodology of building a visual codebook from a single detector/descriptor combination.

The following sections summarize the most influential visual codebook learning approaches. The focus is on the principle of visual codebook learning and its application to generic object recognition problems.

2.3.2 Unsupervised Visual Codebook Learning

Many previous methods construct a visual codebook by pooling all the region descriptors from the training images and then applying an unsupervised clustering algorithm such as *k*-means (Csurka et al. 2004; Sivic and Zisserman 2004; Zhang et al. 2007), Gaussian mixture models (GMM) (Dorko and Schmid 2004; Larios et al. 2008; Perronnin 2008), on-line clustering with mean-shift (Jurie and Triggs 2005), and hierarchical clustering (Agarwal et al. 2004). Each cluster is treated as a codeword in the codebook. A new descriptor is assigned to the nearest codeword in the codebook according to its Euclidean distance or Mahalanobis distance to the centers of the clusters. The generative visual codebook's advantage is its simplicity, its lower risk of overfitting, its computational efficiency and its invariance to image transformations. The proposed approaches have achieved good performance on various object recognition applications. This approach is also employed in this dissertation for the difficult stonefly recognition problem (see Section 4.2).

2.3.3 Supervised Visual Codebook Learning

The unsupervised codebook learning approaches presented above construct large generative codebooks to capture relevant variation of object parts for all the object categories. But in real-world applications, it is common that the generative codebook is suboptimal and not discriminative enough for a specific problem. Therefore, some researchers have begun to introduce discriminative mechanisms into the visual codebook learning process. Winn et al. (2005) first apply *k*-means clustering with a large value *K* to construct a large codebook from manually segmented image patches. Then they employ the information bottleneck principle to merge the codewords whilst preserving their discriminative power. Perronnin (2008) also first learn a generative (universal) visual codebook on regularly sampled features and then employ MAP estimation to adapt the generative codebook to class-specific data; an SVM is learned to select among the generative and adapted clusters. Moosmann et al. (2007) learn multiple, independent randomized decision trees to partition the low-level feature space. Class label supervision is employed to search for discriminative split points during the construction of trees. The leaves of the decision trees define the codewords. More recently, Yang et al. (2008) proposed a codebook learning framework that is integrated with classifier learning. In their work, visual codebooks have a restricted form as a sequence of visual bits. Each visual bit is a linear classifier that maps the low-level features to a binary bit for classification. The learning is performed in a sequential manner, where the performance of the classifier using previous visual bits is used to extract the next set of visual bits.

Discriminative visual codebook learning approaches employ the class labels during the construction of codewords. They usually show higher performance in real-world object recognition applications. In this dissertation, two novel discriminative visual codebook learning algorithms (Zhang and Dietterich 2008; Zhang et al. 2009) are proposed (see Sections 4.4 and 4.5). Both algorithms achieve high performance on object recognition tasks.

2.3.4 Building Image Attribute Vectors from Visual Codebooks

Given the learned visual codebook, there are different ways to produce the image attribute vector from the low-level image features. The most common mapping method is the "histogram of occurrences" strategy (Csurka et al. 2004; Jurie and Triggs 2005; Moosmann et al. 2007; Zhang et al. 2007; Larios et al. 2008). This method assigns each descriptor vector extracted from an image to the closest codeword; then accumulates the occurrences of each codeword to form a histogram attribute vector. Agarwal et al. (2004), Jurie and Triggs (2005) and Winn et al. (2005) employ a "binary indicator" vector, which is a binarized version of the histogram of occurrences vector. Each codeword's feature is 1 if there is at least one descriptor that is assigned to the codeword occurs in an image, and 0 otherwise. A real-valued version of the histogram of occurrences strategy is employed in Perronnin (2008). Each of the descriptor vectors extracted from an image is assigned to the codewords in a probabilistic manner; and the probabilities are accumulated to form the image attribute vector for classification. Term frequency - inverse document frequency (*tf-idf*) strategy is used in Sivic and Zisserman (2004) motivated by the common application of weighting words in text retrieval. The *tf* term weights codewords occurring often in a particular image. It is exactly the histogram of occurrences as described previously. The *idf* term down-weights the codewords that appear often in the entire image set.

In this dissertation, both histogram of occurrences and tf-idf methods are employed in the new visual codebook learning methods. They will be introduced in more detail in Chapter 4.

2.3.5 Image Classification Based on Image Attribute Vectors

Standard learning algorithms, including nearest neighbor classifiers (Sivic and Zisserman 2004; Dorko and Schmid 2004; Winn et al. 2005), linear classifiers (Yang et al. 2008), Naïve Bayes classifiers (Csurka et al. 2004), and Support Vector Machines (Csurka et al. 2004; Jurie and Triggs 2005; Moosmann et al. 2007) have been applied to the image attribute vectors to train the final classifier. Zhang et al. (2007) also employs a SVM as the image classifier, but its kernel function is based on the earth mover's distance or χ^2 distance for the specific "signature" based image representations. Some other classifiers were also employed in previous visual codebook approaches. Larios et al. (2008) employs an ensemble learning method − bagged logistic model trees − to classify stonefly images based on concatenated attribute vectors (see Section 4.2.2). Perronnin (2008) uses both sparse logistic regression (SLR) and a linear SVM as the image classifier. Agarwal et al. (2004) employs a Sparse Network of Winnows (SNoW) learning algorithm as the classifier to take advantage of the sparseness properties of the image attribute vectors.

In this dissertation, we study the discriminative properties of the image attribute vector; and employ ensembles of weak classifiers (Dietterich 2000) to classify the images. Both bagged decision lists (Section 4.4) and bagged decision trees (Section 4.5) are applied to efficiently select and combine the most discriminative attributes for image classification. These approaches are motivated by previous work on classifying images based on discriminative image features, as introduced in next section.

2.4 Object Recognition Approaches Based on Discriminative Feature Selection

Discriminative feature selection has been extensively explored in computer vision for object detection and recognition. The motivation is that oftentimes only a small number of the most discriminative low-level image features (region descriptors) are enough for high-accuracy classification. The difficulty is how to identify these features from the large image feature pool. This section introduces previous work in this direction that is most related to our approaches.

As introduced in Chapter 1, the application of interest region detectors and descriptors transform the original images into bags of region descriptor vectors. The class labels are attached to the bags instead of to the descriptor vector instances. Chen et al. (2006) proposes the Multiple-Instance Learning via Embedded Instance Selection (MILES) algorithm for this multiple-instance learning problem (Dietterich et al. 1997). MILES maps each bag into a fixed-length attribute vector by measuring the similarities between the instances in the bag and all the instances in the training bags. A 1-norm SVM is employed to select discriminative image features and construct classifiers simultaneously.

This feature mapping strategy is also used in Opelt et al. (2006). But instead of using an SVM to select the discriminative features, a modified AdaBoost (Freund and Schapire 1996) algorithm is employed to select multiple image features and learn the corresponding weak classifiers sequentially. The pseudo-code of this boosting learning framework is shown in Figure 9. AdaBoost assigns weights to training images. Each iteration of boosting calls the *Weak-Hypotheses-Finder* function to select the most discriminative image attribute and learn its corresponding weak hypothesis (decision stump) according to the image weights. Then the error of the weak hypothesis (classifier) is calculated and the weights of the images are updated according to the correctness of the predictions made by the weak classifier. The weights of the correctly classified images are decreased and the weights of the incorrectly classified images are increased. Then the learning algorithm iterates to learn the next weak classifier using the updated weights. This boosting learning framework and its extensions are employed in our hierarchical recognition system based on multi-scale PCBR regions (Section 3.3.2) and in the non-redundant visual codebook learning method (Section 4.5).

The *Weak-Hypotheses-Finder* function in Opelt et al. (2006) evaluates the discriminative power of all the image attributes based on the current image weights and selects the best one to learn the weak classifier. The pseudo-code of this algorithm is given in Figure 10. In the dissertation, this algorithm is modified and applied to calculate the MMI score (see Section 4.3) and learn a decision list classifier (see Section 4.4). The details of these algorithms will be introduced later.

As presented in previous sections, Dorko and Schmid (2004) apply Gaussian mixture models (GMM) to build the visual codebook from region descriptors. Each codeword (cluster) is treated as a *part classifier* that assigns a new descriptor vector to the corresponding object part. Since most of the codewords are probably noisy, feature selection algorithms are applied to rank the part classifiers according to their discriminative abilities. One of the selection algorithms is based on mutual information (MI) criterion, which ranks the part classifiers based their information content for separating the background from the object-class. The mutual information of part classifier C_k and object-class O is

$$R_I(C_k) = P(\overline{C}_k,\overline{O})\log\frac{P(\overline{C}_k,\overline{O})}{P(\overline{C}_k)P(\overline{O})} + P(C_k,\overline{O})\log\frac{P(C_k,\overline{O})}{P(C_k)P(\overline{O})} +$$

$$P(\overline{C}_k,O)\log\frac{P(\overline{C}_k,O)}{P(\overline{C}_k)P(O)} + P(C_k,O)\log\frac{P(C_k,O)}{P(C_k)P(O)} \quad (1)$$

The part classifiers of highest rank are applied to classify a new image by simply counting the number of descriptor vectors that are positively labeled by these classifiers. In this dissertation, we modify the MI criterion in Equation (1) and apply it to evaluate the discriminative properties of the generative visual codebooks in Section 4.3.

Inputs: Training images $(I_1, l_1), (I_2, l_2), \ldots, (I_m, l_m)$,

($l_i = +1$ if image I_i contains at least one instance of the relevant object and $l_i = -1$ if image I_i contains no relevant object).

Initialization: Set the weights $w_1 = w_2 = \ldots = w_m = 1$.

For $t = 1, \ldots, T$

1) Get a weak hypotheses h_t according to weights $w_1, w_2, \ldots w_m$ from the *Weak-Hypotheses-Finder* (shown in Figure 10).

2) Calculate the weighted error of the weak hypothesis:
$$\varepsilon = \frac{\sum_{i=1, h_t(I_i) \neq l_i}^{m} w_i}{\sum_{i=1}^{m} w_i}$$

3) Choose:
$$\beta_t = \begin{cases} \sqrt{\dfrac{1-\varepsilon}{\varepsilon}} \cdot \eta & \text{if} \quad l_i = +1 \quad \text{and} \quad l_i \neq h_t(I_i) \\ \sqrt{\dfrac{1-\varepsilon}{\varepsilon}} & \text{else} \end{cases}$$

where η is an additional weight factor to control the update of false classified positive examples.

4) Update the weight: $w_i \leftarrow w_i \cdot \beta^{-l_i \cdot h_t(I_i)} \quad$ for $\quad i = 1, \ldots, m$

Output: The final hypotheses (image classifier):

$$H(I) = \begin{cases} +1 & \text{if} \quad \sum_{t=1}^{T} (\ln \beta_t) h_t(I) \geq th_{Ada}, \\ -1 & \text{else.} \end{cases}$$

Figure 9. Modified AdaBoost algorithm in Opelt et al. (2006).

Inputs: Training image attribute vectors: $\boldsymbol{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{M,1} & a_{M,2} & \cdots & a_{M,m} \end{bmatrix}$,

and the class labels of the training images: $\boldsymbol{L} = (l_1, l_2, \dots, l_m)$.

(M is the total number of descriptor vectors extracted from the training images)

Sorting: For each dimension (row) k of training attribute vectors $\boldsymbol{A}$, sort the m images in non-decreasing order of the attribute values, that is, find a permutation $\pi_k(1)$, $\pi_k(2)$, …., $\pi_k(m)$, such that:

$$a_{k,\pi_k(1)} \le a_{k,\pi_k(2)} \le \cdots \le a_{k,\pi_k(m)}.$$

Select the best weak hypothesis (Scanline): For all the attributes (a_k) calculate over all images I_i

$$\max_s \sum_{i=1}^{s} w_{\pi_k(i)} l_{\pi_k(i)}$$

and select the attribute a_k where the maximum is achieved.

Compute classification threshold: at the position s where the Scanline reached a maximum sum the threshold θ is set to:

$$\theta = \frac{a_{k,\pi_k(s)} + a_{k,\pi_k(s+1)}}{2}$$

Figure 10. Pseudo-code for the Weak-Hypotheses-Finder algorithm in Opelt et al. (2006).

3 Image Features and Learning Algorithms for Stonefly Recognition

3.1 Overview

As introduced in Chapter 1, the recognition of stoneflies is very challenging due to large intra-class variations in the images and small differences between categories. Figure 1 shows the examples of images belonging to different categories of stoneflies. The properties of stonefly larvae require *high-precision* recognition approach that is robust to deformations and variations. Motivated by the successful application of interest region detectors and region descriptors (see Section 2.2) in object recognition problems, we develop a similar bag-of-keypoints approach for the automated recognition of stoneflies.

In order to handle the unique challenges posed by this biological object recognition problem, we design a new interest region detector called the PCBR detector (Zhang et al. 2006; Deng et al. 2007) which outperforms other state-of-the-art detectors on the stonefly recognition problem. This new detector will be introduced in Section 3.2. We also design various object recognition architectures based on the bag-of-keypoints image representations. All these methods integrate different types of low-level image features to recognize the stoneflies with high accuracy. They will be described in Sections 3.3, 3.4 and 3.5.

3.2 Principal Curvature-Based Region (PCBR) Detector

3.2.1 Motivation

As described previously, images of the stoneflies exhibit large intra-class variation. This variation causes considerable variation in local intensity. Therefore, local intensity properties are no longer stable detection cues for the identification and localization of interest regions. As shown in Figure 6, the state-of-the-art Harris-Affine and Hessian-Affine (Mikolajczyk and Schmid 2004) detectors and salient region detector (Kadir et al. 2004) all fail in detecting the most characteristic regions in stonefly images. Their detections (corners or blobs) focus on the very local patterns in the images, which are suitable for image matching applications, but usually lack the discriminative power that is needed for object recognition approaches.

The limitations of existing local detectors inspired our idea of developing a new interest region detector that captures semi-local structural cues, which tend to be more robust to intensity, color and pose variations. We present the Principal Curvature-Based Region (PCBR) detector (Zhang et al. 2006; Deng et al. 2007) which exploits the salient curvilinear structures to reliably detect the interest regions. Curvilinear structures are lines (either curved or straight) such as roads in aerial or satellite images or blood vessels in medical scans. We choose curvilinear structure as detection cues for three reasons. First, by generating a single response for both lines and edges, curvilinear structure produces a clear structural sketch of the object, which is termed as principal curvature image. It provides the basis for non-local detections. Second, it is a fairly stable feature that can be detected over a range of viewpoints, scales, and appearance changes. Third, it usually produces semi-local regions that are more characteristic for object classes compared with local patterns, such as corners and blobs. PCBR applies the watershed transformation to the principal curvature image across scale space to form the interest regions that have coherent interiors and that are surrounded by distinctive curvilinear structure. As shown in Figure 6, PCBR is able to reliably detect the regions corresponding to the most salient and characteristic patterns on stoneflies. More examples of PCBR detections on stoneflies are shown in Figure 11.

The principles of the PCBR detector are introduced in Section 3.2.2. The approaches we used to improve the robustness of PCBR are described in Section 3.2.3. The performance of the PCBR detector is extensively evaluated in Section 3.2.4 in the context of image matching and object recognition.

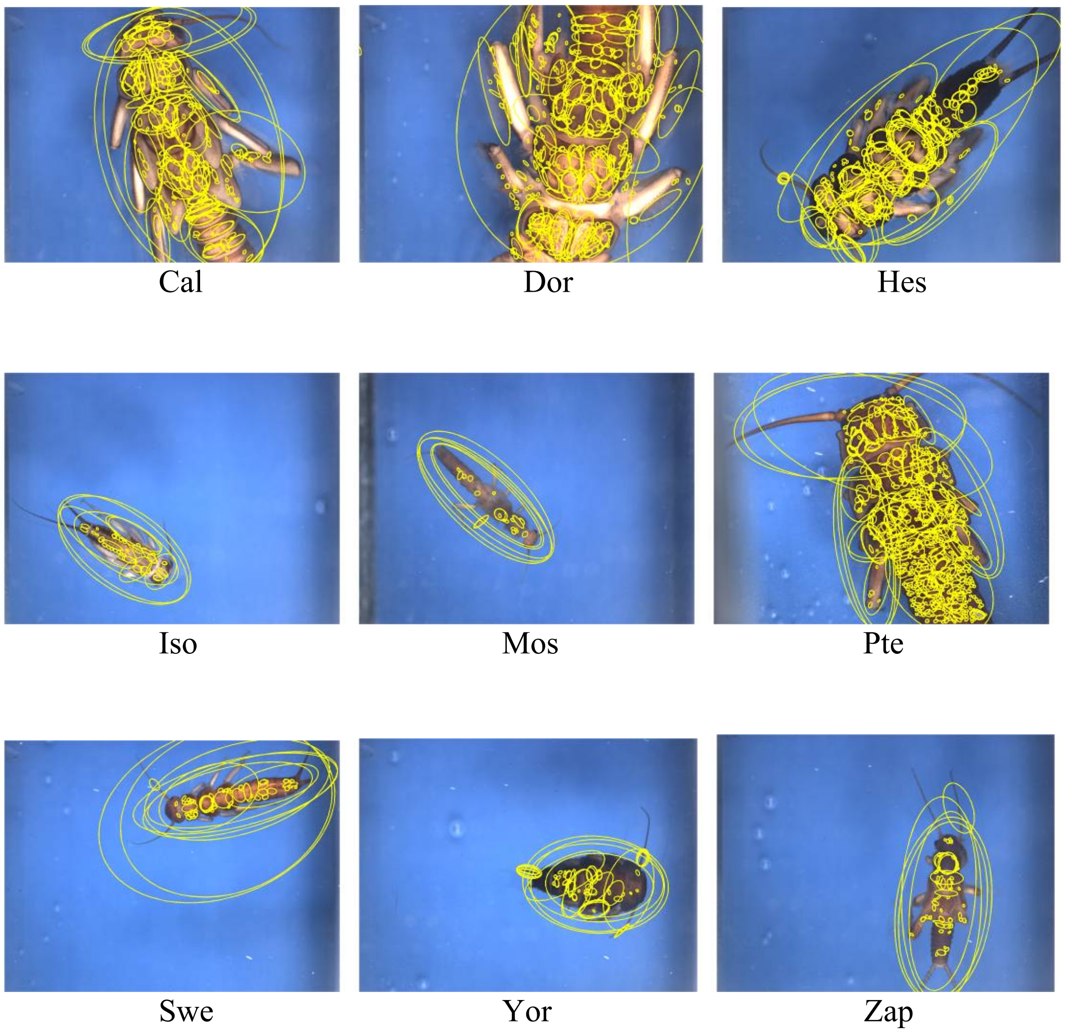


Figure 11. Examples of PCBR detections on nine categories of stoneflies.

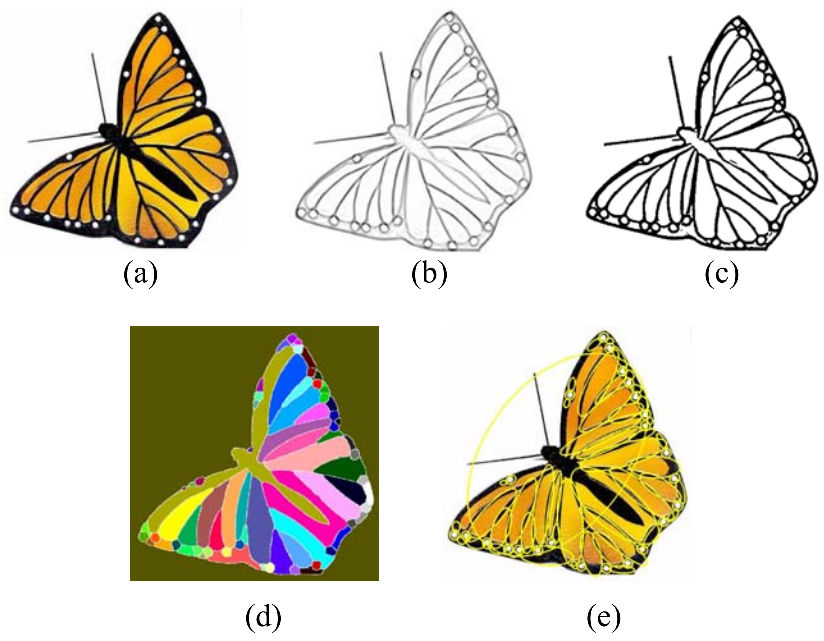


Figure 12. The detection procedure of PCBR detector: (a) Original image. (b) Principal curvature image. (c) Cleaned binary image. (d) Watershed regions. (e) Detected regions represented by ellipses.

3.2.2 PCBR Detection Procedure

The basic detection procedure of PCBR is illustrated in Figure 12. The PCBR detector first employs Steger's curvilinear detector algorithm (Steger 1998) to compute the structural sketch of the image, as shown in Figure 12 (b). We call it the principal curvature image, as it approximates the principal curvature of the image intensity surface. Steger's algorithm computes the eigenvalues of the Hessian matrix at each pixel and then forms an image that is composed of one of the two eigenvalues. The Hessian matrix at a particular point $\boldsymbol{x}$ is,

$$\boldsymbol{H}(\boldsymbol{x},\sigma_D) = \begin{bmatrix} I_{xx}(\boldsymbol{x},\sigma_D) & I_{xy}(\boldsymbol{x},\sigma_D) \\ I_{xy}(\boldsymbol{x},\sigma_D) & I_{yy}(\boldsymbol{x},\sigma_D) \end{bmatrix} \tag{2}$$

where $I_{xx}(\boldsymbol{x},\sigma_D)$, $I_{xy}(\boldsymbol{x},\sigma_D)$ and $I_{yy}(\boldsymbol{x},\sigma_D)$ are the second-order derivatives of the image as computed by convolving the image with the appropriate second-derivative of a Gaussian with scale σ_D. For each pixel $\boldsymbol{x}$ in an image, we compute the eigenvalues $\lambda_1(\boldsymbol{x})$ and $\lambda_1(\boldsymbol{x})$ of Equation (2). The principal curvature response is given by either

$$P(\boldsymbol{x}) = \max(\lambda_1(\boldsymbol{x}),0) \tag{3}$$

or

$$P(\boldsymbol{x}) = \min(\lambda_2(\boldsymbol{x}),0) \tag{4}$$

Equation (3) provides a high response only for dark lines on a light background (or on the dark side of edges); while Equation (4) detects light lines against a dark background. The principal curvature image is formed by compute the principal curvature response at each pixel, as shown in Figure 12 (b).

As described in Section 2.2.1, the Hessian matrix has been applied in several interest region detectors (Harris-Affine, Hessian-Affine: Mikolajczyk and Schmid 2004) to find the salient patterns in the objects. However, the PCBR detector is different and complementary to these detectors. Rather than finding extremal "points", PCBR applies the watershed transformation to ridges, valleys and cliffs of the image principal curvature surface to find semi-local "regions". These regions are usually more robust to image deformations and more characteristic for object classes. The watershed algorithm provides a more efficient mechanism for defining structural regions than previous methods of fitting circles, ellipses, and parallelograms. However, the watershed algorithm is sensitive to noise and other small image perturbations. To improve robustness to noise, we "clean" the principal curvature image with a grayscale morphological close operation followed by a new hysteresis thresholding method based on local eigenvector flow, which will be introduced in the following section. The resulting binary image is usually clear single-line sketch of the image, as shown in Figure 12 (c). The watershed transform is applied to the cleaned principal curvature image to segment the image into watershed regions (i.e., the catchment basins), as shown in Figure 12 (d). Finally, each watershed region is approximated with an ellipse having the same second moment to produce the PCBR detection, as shown in Figure 12 (e).

3.2.3 Approaches to Improve the Robustness of PCBR

Various methods have been applied to the PCBR detection procedure in order to improve its robustness to image deformations. These methods include principal curvature image pyramid in scale space, enhanced watershed transform, and searching for stable PCBR regions in scale space. These will be introduced in the following sections.

Principal curvature image pyramid in scale space

In order to improve the robustness of PCBR detector to image deformations, we compute the principal curvature images in scale space in a fashion similar to that of SIFT (Lowe 2004). We first double the size of the original image to produce our initial image, $\boldsymbol{I}_{11}$, and then produce increasingly Gaussian smoothed images, $\boldsymbol{I}_{1j}$, with scales of $\sigma = k^{j-1}$ where $k = 2^{1/3}$ and $j = 2, 3, \ldots, 6$. This set of images spans the first octave consisting of 6 images, $\boldsymbol{I}_{11}$ to $\boldsymbol{I}_{16}$. Image $\boldsymbol{I}_{14}$ is downsampled to half its size to produce image $\boldsymbol{I}_{21}$, which becomes the first image in the second octave. We apply the same smoothing process to build the second octave, and continue to create a total of $n = log2(min(w, h)) - 3$ octaves, where w and h are the width and height of the doubled image, respectively. Finally, we calculate a principal curvature image, $\boldsymbol{P}_{ij}$, for each smoothed image by computing the maximum eigenvalue (see Equation (3) and (4)) of the second derivative Hessian matrix at each pixel (Equation (1)). For computational efficiency, each smoothed image and its corresponding Hessian image are computed from the previous smoothed image using an incremental Gaussian scale.

The result of the computation is the principal curvature image pyramid in scale space. We then calculate the maximum curvature over every three consecutive principal curvature images to form the following set of 4 images in each of the *n* octaves:

$$\begin{array}{cccc} MP_{12} & MP_{13} & MP_{14} & MP_{15} \\ MP_{22} & MP_{23} & MP_{24} & MP_{25} \\ \ldots & & & \\ MP_{n2} & MP_{n3} & MP_{n4} & MP_{n5} \end{array} \tag{5}$$

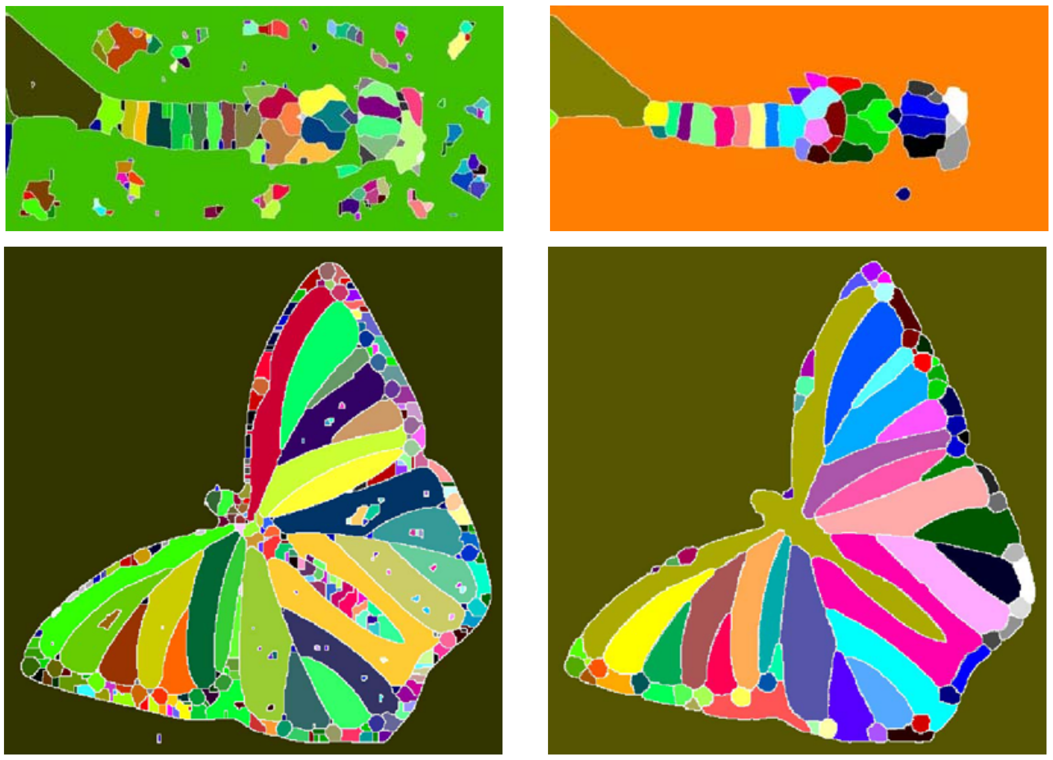
where $MP_{ij} = \max(P_{ij-1}, P_{ij}, P_{ij+1})$.

A maximum curvature image *MP* is calculated by maximizing the principal curvature at each pixel over three consecutive principal curvature images. These maximum principal curvature images provide a more stable structural sketch of the images for the detection of PCBR regions.

Enhanced watershed transform

The watershed transform is an efficient technique that is widely employed for image segmentation. We apply the watershed transform to the principal curvature image to detect interest regions. However, the watershed transform is sensitive to noise (and other small perturbations) in the intensity image. A consequence of this is that the small image variations form local minima that result in many, small watershed regions. Figure 13 (a) shows the oversegmentation results when the watershed algorithm is applied directly to the principal curvature image. To achieve more stable watershed segmentation, we apply a grayscale morphological closing followed by hysteresis thresholding to obtain “cleaner” binary images for segmentation.

The grayscale morphological closing operation removes small “potholes” in the principal curvature terrain, thus eliminating many local minima that result from noise and that would otherwise produce watershed catchment basins. But other than the small noisy basins, there are other variations that have larger zones of influence and cannot be removed by the morphological closing. In order to further eliminate spurious or unstable watershed regions, we threshold the principal



(a) (b)

Figure 13. (a) Watershed transform applied to original principal curvature images. (b) Watershed transform applied to the "cleaned" principal curvature images.

curvature image to create a clean binarized principal curvature image. However, the principal curvature response can become weak due to the low contrast of an edge or curvilinear structure. These low contrast segments may potentially cause gaps in the thresholded principal curvature image, which in turn causes watershed regions to merge that should otherwise be separate. Fortunately, the directions of the eigenvectors provide a strong indication of where curvilinear structures appear and they are more robust to these intensity perturbations than the eigenvalue magnitude. Therefore, we apply a more robust eigenvector-guided hysteresis thresholding to help link structural cues and remove perturbations.

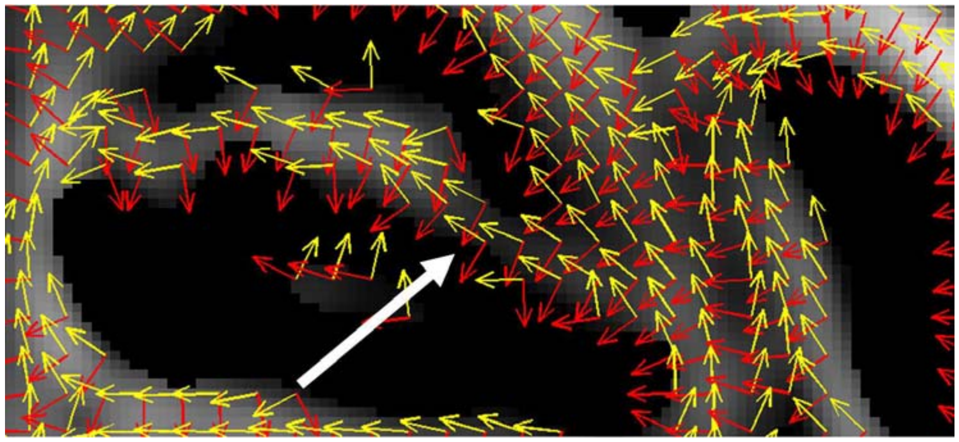


Figure 14. Illustration of how the eigenvector flow helps in connecting the weak principal curvature responses.

Eigenvector-flow hysteresis thresholding works with a higher threshold and a lower threshold the same as traditional hysteresis thresholding. Pixels with a strong response act as seeds that expand out to include connected pixels that are above the lower threshold. Unlike traditional hysteresis thresholding, our low threshold is a function of the support that each pixel's major eigenvector receives from neighboring pixels. The lower threshold is set on every pixel by comparing the direction of the major (or minor) eigenvector to the direction of the adjacent pixels' major (or minor) eigenvectors. This can be done by taking the absolute value of the inner product of a pixel's normalized eigenvector with that of each neighbor. If the average dot product over all neighbors is high enough, we set the low-to-high threshold ratio to 0.2; otherwise the low-to-high ratio is set to 0.7. These ratios were chosen empirically.

Figure 14 illustrates how the eigenvector flow helps in the connection of weak curvilinear structure. The red arrows are the major eigenvectors, and the yellow arrows are the minor eigenvectors. At the point indicated by the large white arrow, we see that the principal curvature response is small and the ridge is almost invisible. Nonetheless, the directions of the eigenvectors are quite uniform. This eigenvector-based active thresholding process yields better performance in building continuous ridges and in handling perturbations, which results in more stable regions as shown in Figure 13 (b).

Select stable regions across scales

To further improve the robustness of PCBR detections, we employ the key idea of the MSER detector (Matas et al. 2002) and keep only those regions that can be detected in at least three consecutive scales, as shown in Figure 15. Similar to the process of selecting stable regions via continuous thresholding as in MSER, we select the regions that are stable across scale changes. To achieve this, we compute the overlap errors of the detected regions across each triplet of consecutive scales in every octave. The overlap error is calculated in the same way as in (Mikolajczyk et al. 2005a). Overlapping regions that are detected at different scales normally exhibit some variation, especially for non-rigid biological objects. This variation is favorable for object recognition tasks, because it provides multiple descriptions of the same pattern, and these reduce the effect of the non-perfect localization of interest regions in objects (Haja et al. 2008). Therefore, we keep the overlapping regions and extract descriptor vectors for each region to provide more informative representations of local image features.

To determine a threshold value for the permitted amount of overlap, we performed a sensitivity analysis of the SIFT descriptor. Three transformations (translation, rotation and minor axis enlargement) were applied to the detected regions in the INRIA image matching dataset (*INRIA*). Overlap errors and similarities of SIFT descriptors between the transformed regions and the originals are calculated. To keep regions that can be detected over local scales, only regions with overlap error less than 30% are chosen. However, as indicated in Figure 16, SIFT similarity decreases to 70% for regions that have an overlap error of 30%. Consequently, we keep all stable regions with an overlap error greater than 30% to maintain more descriptions for similar regions. We also notice that the similarity

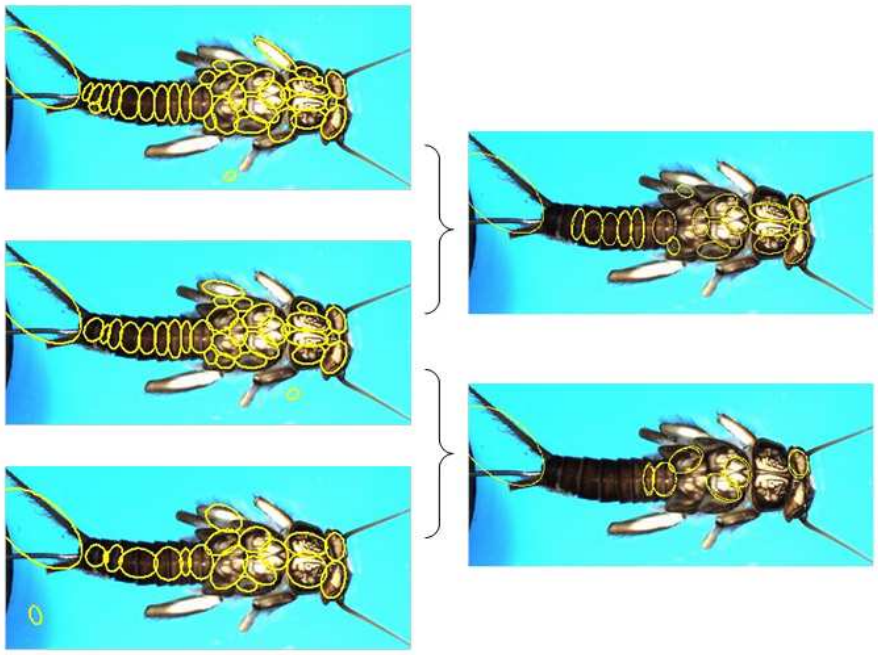


Figure 15. Illustration of selecting stable PCBR regions across scales.

of the SIFT descriptors is above 90% when the overlap error is less than 10%. These very similar regions are merged into a single region.

3.2.4 Performance Evaluation of PCBR

As introduced in Section 2.2, the performance of an interest region detector is evaluated using different criteria for different tasks. To evaluate the PCBR detector for both image matching and object recognition problems, we evaluate in three ways: 1) visual inspection, 2) repeatability measurement, and 3) maximum mutual information scores and curves.

Visual Inspection

Figure 11 in Section 3.2.1 shows the PCBR detections on stonefly images. We can see that rather than detecting regions clustered around a few distinctive

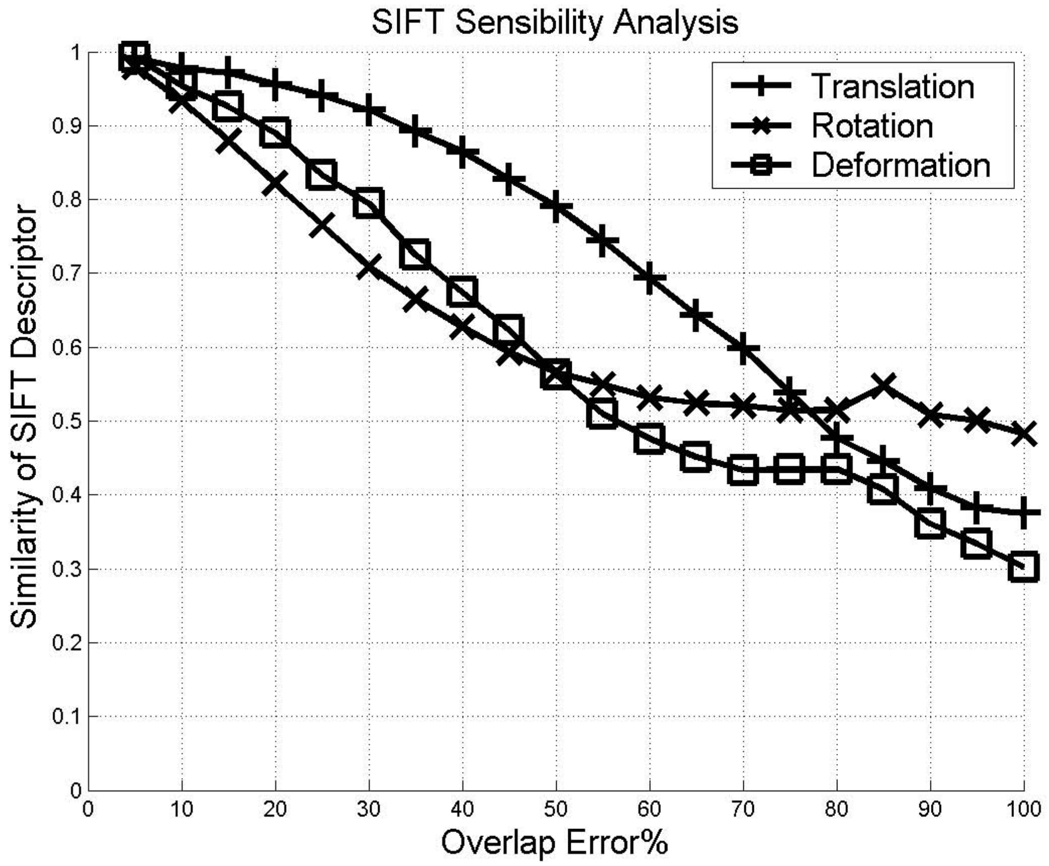
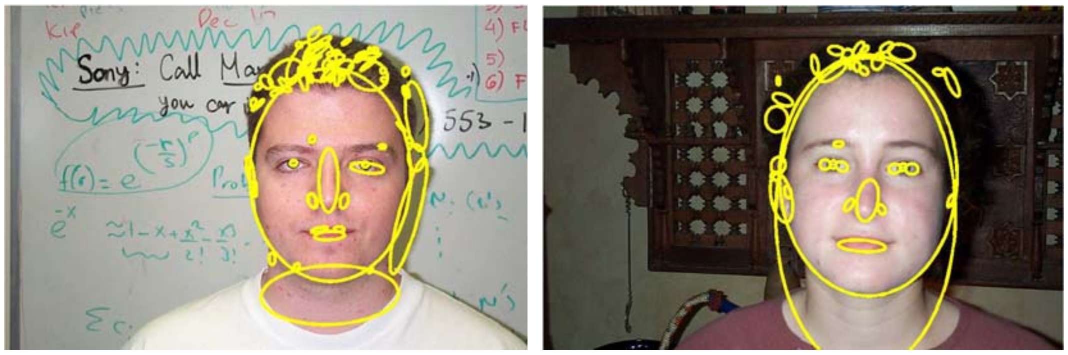
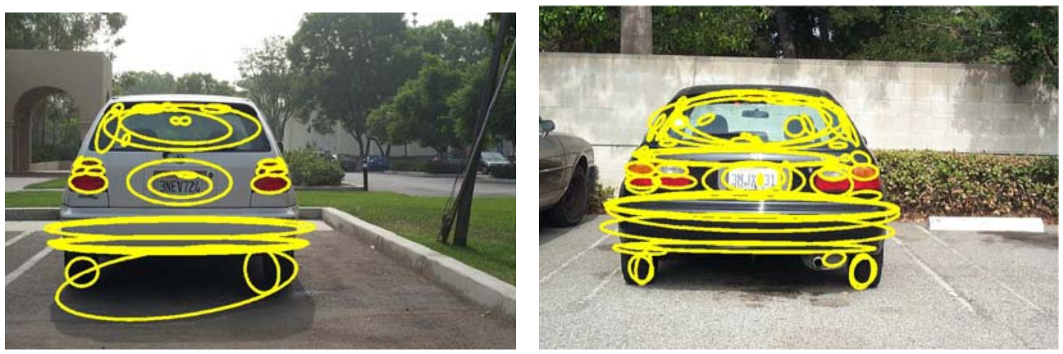


Figure 16. Sensitivity analysis of SIFT descriptors.

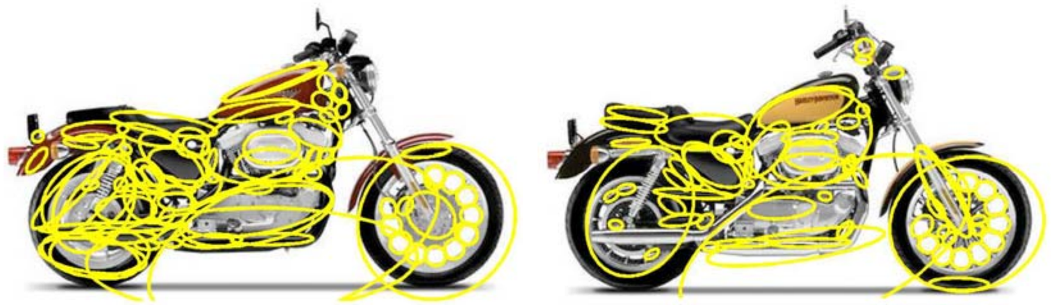
areas in the image, PCBR usually outputs evenly distributed detections that are more characteristic and more robust to occlusion and local image perturbations. Figure 17 shows the PCBR detections on human faces, cars and motorbikes from the Caltech object categorization dataset (*Caltech*). Figure 18 presents the PCBR detections on two graffiti images from the INRIA dataset (*INRIA*). From these figures, it can be observed that PCBR regions are quite consistent across different views of the scene and robust to the variations of the objects.



(a)



(b)



(c)

Figure 17. Examples of interest regions detected by PCBR on generic objects: (a) human faces, (b) cars, and (c) motorbikes from Caltech dataset.

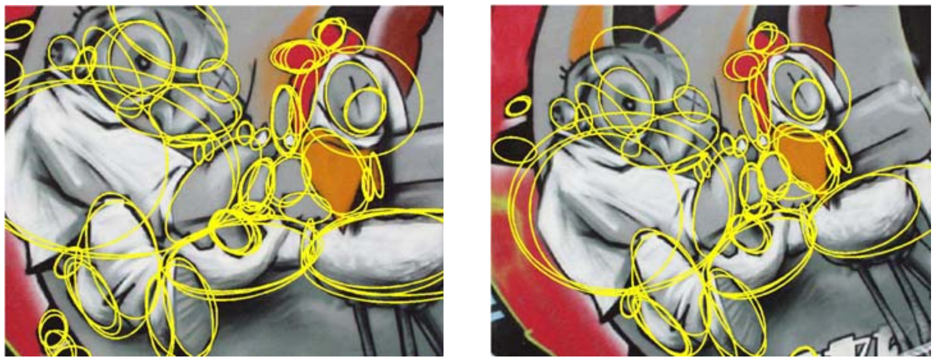


Figure 18. Examples of PCBR detections on graffiti images from INRIA dataset.

Repeatability Evaluation

The PCBR detector is designed for biological and generic object recognition rather than wide baseline matching (Shapiro and Stockman 2001), but we still evaluate its repeatability (see Section 2.2.3) and compare it to other detectors using the method introduced by Mikolajczyk et al. (2005a). The *INRIA* dataset and code are used for our experiments. Table 1 provides the average repeatability of the PCBR detector compared to other detectors. Average repeatability is determined from the repeatability vs. transformation curves as output by the INRIA evaluation code (with the overlap error parameter set to 20%). As we can see, the PCBR detector is comparable to other detectors in terms of repeatability.

Evaluation via Maximum Mutual Information (MMI) Scores

As introduced in Section 2.2, Repeatability evaluation criteria (Mikolajczyk et al. 2005a) and other similar methods are designed mainly for matching or image registration; they are not well-defined in the context of object class recognition. In this dissertation, we evaluate a detector directly based on the discriminative properties of the detections.

Table 1. Average repeatability of the PCBR detector and other detectors on the INRIA dataset.

Images	*PCBR*	*Hessian-Affine*	*Harris-Affine*	*MSER*	*IBR*	*EBR*	*Kadir's Salient*
Bikes	30.2	48.2	32.8	33.6	26.5	37.5	15.3
Trees	10.7	20.4	9.8	11.5	9.6	3.9	2.1
Boats	16.2	29.7	22.3	27.5	12.8	19.8	0.3
Leuven	37.6	40.0	32.0	66.7	34.4	30.1	17.6
Graffiti	35.5	17.7	13.0	51.7	19.7	16.7	2.1
Walls	16.6	24.5	17.3	31.4	14.7	11.1	0

Our evaluation method is motivated by the successful application of "visual codebook" approaches (see Section 2.3) for object recognition. We present the Maximum Mutual Information (MMI) criterion, which efficiently measures the discriminative power of visual codebook entries for specific problems. The highest MMI evaluation is given to the detector that can consistently detect the most discriminative structural or textural patterns (e.g. two eyes in human faces and rear lights of cars) in object images. The detailed description of the MMI criterion is given in Section 4.3.

The PCBR detector is evaluated on benchmark object categorization datasets and compared with other state-of-the-art detectors using the MMI criterion. The PCBR detector achieves MMI evaluation above average on all the object classes. It works especially well on highly structured objects, such as leaves and cars. PCBR is usually able to find several highly distinctive and discriminative patterns, e.g. on leopards and stoneflies. The MMI curves of PCBR and other detectors on stoneflies are shown in Figure 19. More results are given in Section 4.3.

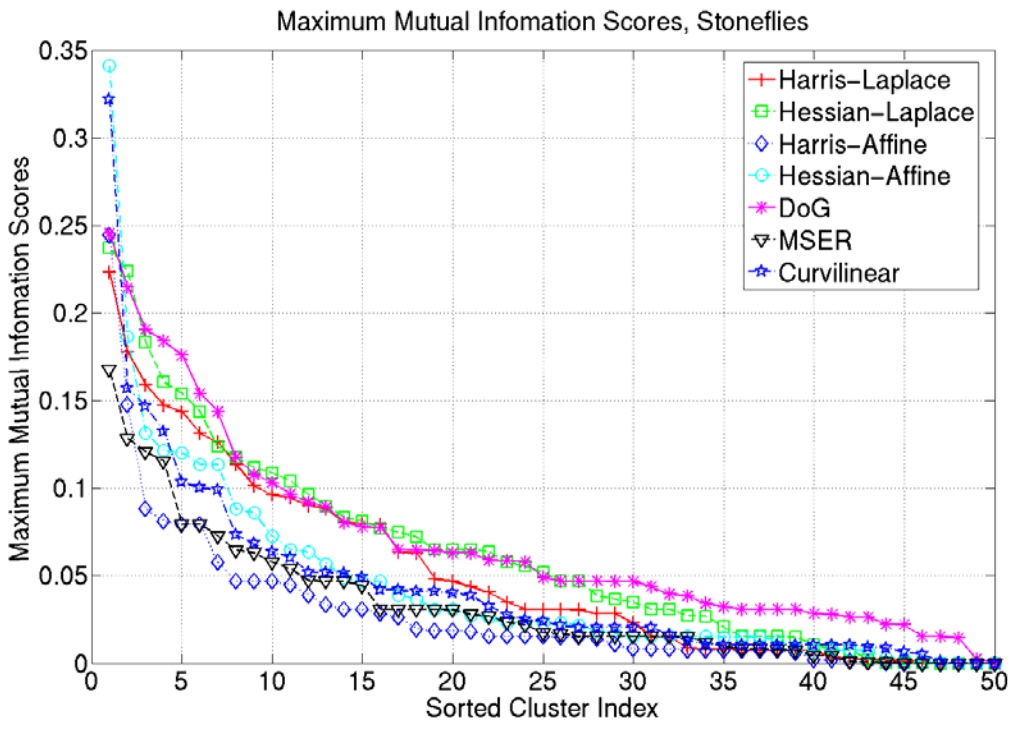


Figure 19. MMI curves of PCBR (Curvilinear) detector and other state-of-the-art detectors on stonefly dataset.

3.3 A Hierarchical Recognition System based on Multi-Scale PCBR Regions

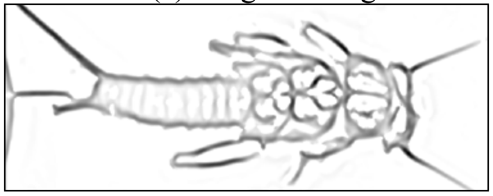
3.3.1 Multi-scale PCBR Regions and Descriptions

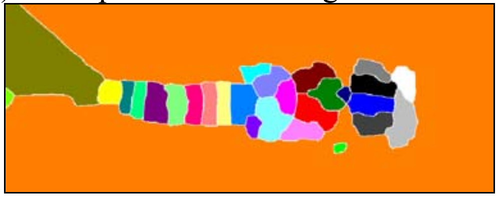
For a lot of object recognition tasks, coarse-to-fine multi-scale descriptions of objects are usually more informative than single-scale description. This is biologically motivated — the human visual system selects and combines both coarse (global) and detailed (local) object features for recognition. Shokoufandeh et al. (1999) use saliency map graphs to capture the salient image structure using multi-scale wavelet transforms. Epshtein and Ullman (2005) proposes feature hierarchies based on mutual information feature selection and parameter adaptation. The work of Bouchard and Triggs (2005) models each object as a hierarchy of parts and subparts with partial transformations (translation and scale transformations) that softly relate the parts and sub-trees to their parents. But there is a common weakness existing in these hierarchical object descriptions: all these descriptions are highly concrete models (trees or graphs). Applying these types of descriptions to classification requires graph matching or model instantiation algorithms.

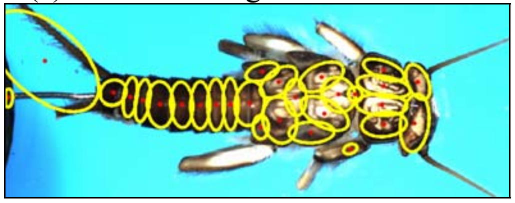
This section introduces a general coarse-to-fine object description based on multi-scale PCBR regions (Zhang et al. 2006). In Equation (2), the parameter σ_D represents the scale of the second-derivative of a Gaussian used to compute the Hessian matrix. To extract the coarse-to-fine object description, we apply the PCBR region detection algorithm at different scales, i.e., using different values of σ_D. Figure 20 shows the multi-scale PCBR region detections on a stonefly image. We can see that a larger scale produces fewer and coarser regions while a smaller scale results in more detailed and local image features.

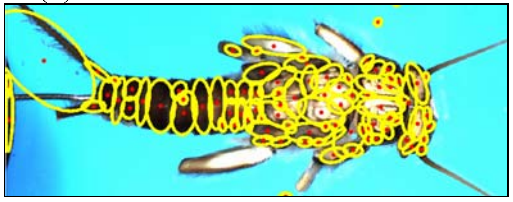
We employ both statistical measurements of region intensities and PCA-SIFT (Ke and Sukthankar 2004) descriptors as the descriptions of the multi-scale PCBR regions. The statistical feature is composed of coefficient of variation, skewness, kurtosis, and moment invariants. The PCA-SIFT descriptors (see Section 2.2.2) are 36-dimensional and have been demonstrated to be more compact and distinctive than SIFT according to Ke and Sukthankar (2004). In addition, we describe the spatial configuration of the multi-scale PCBR regions with bins-based cluster index distribution histograms, inspired by the idea of shape context of Belongie et al. (2002). Basically, the construction of spatial relation descriptions takes three steps. First, cluster the PCA-SIFT descriptors from the positive training images using E-M to determine a GMM with $K = 16$ clusters (Duda et al. 2001). Second, mark the cluster index of each region in training and testing images using maximum likelihood. And third, discretize the distances and directions between

(a) Original image


(b) Principal curvature image at scale $\sigma_D = 4.0$


(c) Watershed regions at scale $\sigma_D = 4.0$


(d) PCBR detections at scale $\sigma_D = 4.0$


(e) PCBR detections at scale $\sigma_D = 2.0$

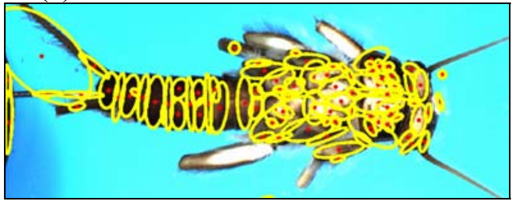

(f) PCBR detections at scale $\sigma_D = 1.0$

Figure 20. Multi-scale PCBR detections on a stonefly image.

regions into M = 36 bins with 12 distinct directions and 3 distance ranges. The sizes of the bins are fixed relative to the image sizes. Thus, the spatial configuration of regions in each image is described by a histogram R composed of $D = K \times M \times K = 16 \times 36 \times 16 = 9216$ elements. An element $R_{i,m,j}$ in R records the number of times a region with cluster index j falls into bin m with center region index i.

3.3.2 Hierarchical Object Recognition System

Based on the multi-scale PCBR regions and their descriptions, we designed a hierarchical object recognition system (Zhang et al. 2006) that uses the coarse-to-fine image analysis to do classification. This system is illustrated in Figure 21. From the top layer to the bottom, we train layer classifiers $L_1, \ldots, L_n$ based on the region descriptors obtained at scales $s_1, \ldots, s_n$, which are in decreasing order (global to local). We then combine the outputs of the layer classifiers to predict the class labels of new images.

Layer classifier

According to the experiments, usually only a small portion of the PCBR regions are useful for classification. So we employ and improve the boosting feature selection algorithm proposed by Opelt et al. (2006), as introduced in Section 2.4. The layer classifiers are learned using the AdaBoost algorithm (Freund and Schapire 1996). Each iteration of AdaBoost evaluates all the unselected descriptors (intensity statistical descriptors, PCA-SIFT, and spatial relation descriptors) of the training images based on the current image weights to find the most discriminative dimension.

We evaluate the statistical intensity descriptors and the PCA-SIFT descriptors in the same way as in Opelt et al. (2006) (see Figure 10). The stability and discriminating power of a descriptor vector $\boldsymbol{v}_j$ is evaluated in three steps. First, calculate the distances from $\boldsymbol{v}_j$ to all the training images. This is done by finding the minimum distance $a_{j,i}$ from $\boldsymbol{v}_j$ to all the descriptor vector in each training image

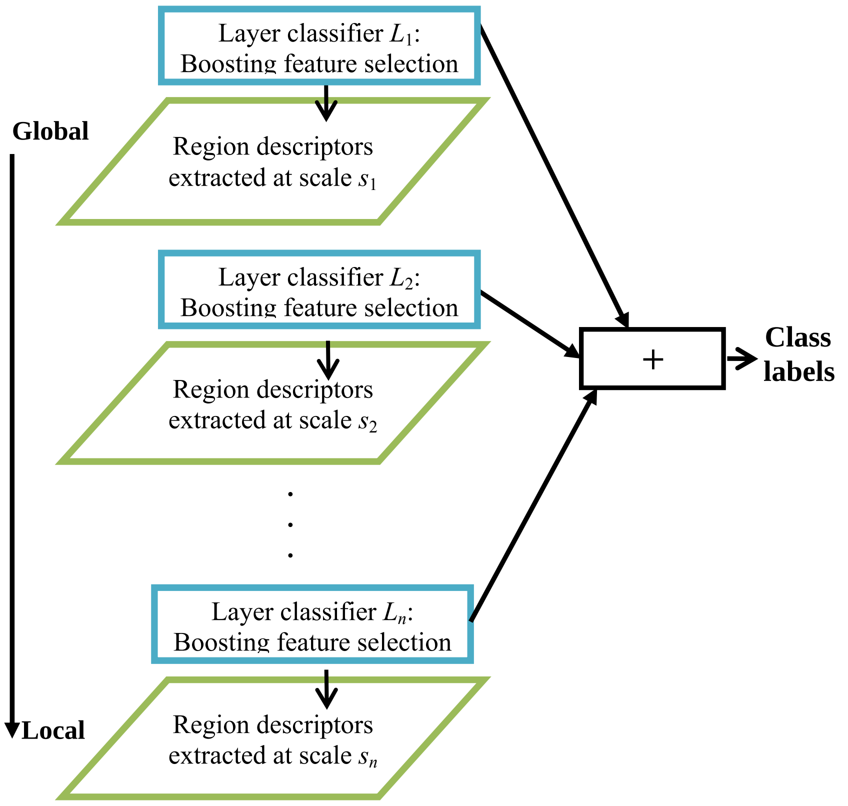

Layer classifier L_1:
Boosting feature selection
Region descriptors extracted at scale s_1
Layer classifier L_2:
Boosting feature selection
Region descriptors extracted at scale s_2
Layer classifier L_n:
Boosting feature selection
Region descriptors extracted at scale s_n
Global
Local
+
Class labels

Figure 21. Illustration of hierarchical object recognition system based on multi-scale PCBR regions and descriptions

I_i. We employ the Mahalanobis distance metric for the statistical intensity descriptor and Euclidean distance for PCA-SIFT. Second, sort the training images into ascending order according to their distance. Third, we apply the *scanline* algorithm to the sorted array of $\boldsymbol{v}_j$ to calculate the maximal weighted sum, which is used as the evaluation of $\boldsymbol{v}_j$.

Evaluating the spatial relation descriptors is simpler because there is no need to calculate the descriptor-to-image distances. The training images are directly sorted according to their spatial relation descriptor values.

A perfect descriptor should have all the positive images (+1) sorted before all the negative images (–1) so that the descriptor vector gives a weak classifier that is perfectly discriminative. The descriptor and threshold $\{\boldsymbol{v}^*, \theta^*\}$ which has maximal score among all the available descriptor vectors is selected as the weak classifier for iteration *t*. We construct *T* weak classifiers at each layer. All these *T* weak classifiers are then combined into a strong classifier (called the layer classifier) using standard AdaBoost, as shown in Figure 9. For presence/absence 2-class object recognition problems, it is not plausible to use negative descriptors to recognize positive examples. So we modify the original algorithm in Opelt et al. (2006) to select only among the features from positive images.

Final classifier

The final result of the hierarchical system is simply the sign of the sum of the outputs of layer classifiers, which is given by:

$$Y = sign(\sum_{i=1}^{n} y_i) \tag{6}$$

In our initial experiments, we also tried to set weights for layer classifiers, and employ the Voted Perceptron algorithm (Freund and Schapire, 1999) to adapt the weights to minimize the classification error on training images, but it overfits the data and the performance degrades.

3.3.3 Experiment Results

We experiment on biological and generic object recognition datasets in order to test the performance of the multi-scale PCBR regions and the hierarchical recognition system. We employed a four-layer system with scales {4.0, 3.0, 2.0, 1.0} and the number of boosting iterations T = 100. The system is tested on six object classes in the Caltech dataset (*Caltech*): airplanes (1074), cars (rear) (526), cars (side) (123), faces (450), leaves (186) and motorbikes (826). The background set in Caltech contains 451 images. We also tested on a subset of the stonefly larva (see Section 3.1) set containing 70 Doroneuria (Dor) images (positive) and 57 Hesperoperla (Hes) images (negative). Examples of the Caltech and stoneflies

Table 2. (a) ROC equal error rates of hierarchical recognition system and other approaches; (b) ROC equal error rates of the full hierarchical system compared to single-layer with spatial relation and 4-layer without spatial relation.

Dataset	*Hierarchical*	*Fergus*	*Opelt*
Airplanes	**90.6**	90.2	88.9
Cars (rear)	**94.3**	90.3	/
Cars (side)	83.6	**88.5**	83.0
Faces	**98.8**	96.4	93.5
Leaves	**97.5**	/	/
Motorbikes	**94.3**	92.5	92.2
Stoneflies	**88.6**	/	/

(a)

Dataset	*4-layer with spatial*	*1-layer*	*4-layer without spatial*
Airplanes	**90.6**	89.0	90.0
Cars (rear)	**94.3**	91.0	89.2
Cars (side)	**83.6**	81.6	80.3
Faces	**98.8**	97.2	**98.8**
Leaves	**97.5**	96.0	97.3
Motorbikes	**94.3**	92.0	93.5
Stoneflies	**88.6**	80.0	82.9

(b)

images are shown in Figure 1 and Figure 2. Half of the images in each set are held out for testing. Recognition performance is evaluated by ROC equal error rates.

The hierarchical system based on multi-scale PCBR region descriptions is tested on these datasets and compared with the constellation model of Fergus et al. (2003) and the boosting feature selection approach by Opelt et al. (2006). The results are summarized in Table 2 (a). The comparison indicates that our hierarchical object recognition system outperforms the other methods on most of the datasets.

In order to test the value of the multi-scale object descriptions, we compared the equal error rates of the whole 4-layer system (denoted as 4-layer with spatial) to the best single layer classifier (1-layer). The results are summarized in the first two columns of Table 2 (b). In the third columns of Table 2 (b), we show the performance of the 4-layer system without spatial relation features (4-layer without spatial) to test the utility of the spatial configuration descriptor. We noticed that on all these datasets, there are significant gaps between the performance of the multi-layer system and that of the best one-layer classifier. This demonstrates that coarse-to-fine object description is more generic and informative for object classes than single scale description. On most of the datasets, spatial relation descriptors improve the performance of the system, thus supporting our assumption that the spatial configurations of the detected regions are also valuable cues for recognition.

3.4 Stonefly Recognition Using Stacked Decision Tree Ensembles

3.4.1 Stacked Decision Tree Ensembles

As described in Chapter 1, the images of the stoneflies have large intra-class variations and small inter-class differences, so that the recognition of their categories is a very challenging task. It is even more challenging when we test on multi-way classification of all nine categories of stoneflies: Cal, Dor, Hes, Iso, Mos, Pte, Swe, Yor and Zap, as shown in Figure 1. This problem is referred as the STONEFLY9 problem. It is different from most prior work that focuses on addressing general object categories that exhibit relatively large inter-category differences in terms of their intrinsic and contextual properties, as exemplified in benchmark datasets, e.g., *Caltech* (see Figure 2) and *GRAZ* (see Figure 3) datasets. Despite PCBR's high performance on stoneflies, especially its multi-scale version (see Section 3.3), extracting only one type of image features is not sufficient to capture the subtle differences between the classes in STONEFLY9. Therefore, we extract different types of low-level image features and use an efficient algorithm to fuse visual information to tackle this difficult problem.

Our method employs stacked decision tree ensembles (Martínez-Muñoz et al. 2009). It involves two major steps: First, a number of diverse image features, including PCBR regions, are extracted from each stonefly image. Second, the low-level image features are directly input to an ensemble of decision trees. The decision trees are capable of fusing disparate types of features without the need for their normalization which is usually a cause of many artifacts. Random forests are trained directly on each kind of feature. Each leaf node in the forest stores a histogram of the class labels of the training image features that reached this leaf. When a new image is encountered, the low-level features extracted from the new image are "dropped" through the learned forest until they reach leaf nodes. The class histograms at all such leaves are summed to produce a single class histogram. This histogram is then used as a feature vector in the second level of our stacked classifier — namely, in a boosted ensemble of decision trees — that makes the final prediction.

3.4.2 Experimental Results

The stacked decision tree ensembles recognition system is evaluated on the STONEFLY9 dataset to test its performance on this difficult recognition task. Four

Table 3. Classification accuracies of stacked decision tree ensembles recognition system on STONEFLY9 dataset.

Dataset	*Our method*	*visual codebook*
Hessian	84.5	66.0
Kadir	88.9	76.1
PCBR	88.8	71.9
Edges	63.7	-
Hessian + edges	88.6	-
Kadir + edges	90.2	-
PCBR + edges	90.8	-
Hessian + Kadir	91.9	81.0
Kadir + PCBR	92.2	80.8
PCBR + Hessian	92.2	79.8
All keypoints	93.6	83.9
Edges + all keypoints	94.4	-

types of features are extracted from stonefly images: Hessian-Affine (Mikolajczyk and Schmid 2004), salient regions (Kadir et al. 2004) and PCBR interest regions (see Section 3.2) described by SIFT descriptors (Lowe 2004), and edge features (Martínez-Muñoz et al. 2009). The experimental setup for STONEFLY9 consists of a stratified 3-fold cross validation using two folds for training and one for testing. The overall classification accuracies corresponding to different combinations of features are summarized in Table 3. As we can see, our decision-tree based image classification method using raw features outperforms categorization methods that use standard visual codebook learning (see Section 2.3) on the STONEFLY9 dataset.

3.5 Stonefly Recognition Using Visual Codebooks

Visual codebooks (see Section 2.3) have been successfully applied to object categorization tasks with bag-of-keypoints image representations. Based on extensive studies on visual codebook learning methods, we designed generative visual codebook (Larios et al. 2008) and discriminative visual codebook learning algorithms (Zhang et al. 2009) for the recognition of stonefly larvae. These algorithms achieve excellent performances on this difficult dataset. This section gives the brief introduction of the algorithms and the experimental results on stonefly dataset. The detailed description of the visual codebook learning algorithms and their application in object categorization will be presented in Chapter 4.

3.5.1 Learning Generative Visual Codebooks

The general framework of employing visual codebook learning for object recognition is illustrated in Figure 8 in Section 2.3. Following the standard unsupervised visual codebook learning approaches (see Section 2.3.2), we construct a separate generative codebook for each region detector and each class by fitting a Gaussian mixture model (GMM) to the feature pool via the Expectation-Maximization (EM) algorithm (Larios et al. 2008). The image attribute vectors are computed via the histogram of occurrences strategy (see Section 2.3.4). The attribute vectors obtained from different codebooks are concatenated to form the final image attribute vectors. These vectors are used to learn bagged logistic model trees to classify the images.

We apply this approach to two stonefly recognition problems: STONEFLY2 (Cal, Dor) and STONEFLY4 (Cal, Dor, Hes, Yor). The classification accuracy of the approach with different combinations of detectors is summarized in Table 4. As we can see, our approach achieves promising performance on these difficult problems. PCBR is shown to be a good complementary detector to employ

Table 4. Classification accuracies of the generative codebook learning method on the stonefly dataset when applied with different combinations of detectors. A √ indicates that the corresponding detector was used.

Hessian	*Kadir*	*PCBR*	Accuracy (%)	
			STONEFLY4	STONEFLY2
√			73.14	70.10
	√		70.64	70.34
		√	71.69	79.03
√	√		78.14	74.16
√		√	80.48	78.68
	√	√	78.31	68.83
√	√	√	82.42	79.73

informative regions on the stoneflies. More details of this approach will be presented in Section 4.2.

3.5.2 Learning Non-Redundant Visual Codebooks

Most of the previous methods (see Section 2.3) construct a single codebook to describe the distribution of the low-level image features. But in real-world applications, data can be represented in many different ways; and oftentimes a single codebook is not enough to fully describe the different structures of the data. So we present methods to learn multiple codebooks that are non-redundant in discriminative power, motivated by the idea of boosting and multi-view clustering. The details of this method will be presented in Section 4.5. This non-redundant visual codebook learning algorithm is applied to the STONEFLY2 and STONEFLY4 problems, and also to the STONEFLY9 problem which involves the

Table 5. Classification accuracies of the non-redundant codebook learning method compared with other methods on stonefly recognition datasets.

Dataset	*Boost*	*Larios08*	*Opelt06*
STONEFLY2	**97.85**	79.37	70.10
STONEFLY4	**98.21**	82.42	/
STONEFLY9	**95.09**	/	/

multi-way classification of all the nine categories of stoneflies. The performance of this algorithm (*Boost*) is compared with other methods: Larios et al. (2008) and Opelt et al. (2006). The experiment uses the same 3-fold cross validation setting as in Larios et al. (2008). For a fair comparison, we also employ the same interest region detectors—Hessian-Affine regions (Mikolajczyk and Schmid 2004), salient regions (Kadir et al. 2004) and PCBR regions (see Section 3.2) — described by SIFT descriptors (Lowe 2004). The results are summarized in Table 5.

From Table 5, we can see that our non-redundant codebook learning algorithm performs much better that the previous work (Larios et al. 2008; Opelt et al. 2006). Figure 22 shows the performance of our algorithm on the stonefly datasets versus the number of boosting iterations. We can see that by comparing the starting points of the curves (using a single codebook of size K = 100) that the addition of non-redundant codebooks significantly improves the discriminative power of the recognition system. For all three tasks, the learning converged within 25 iterations and showed no sign of overfitting.

In summary, the properties of the stonefly dataset pose the following requirements for the recognition system. (1) Due to the high complexity of the objects, a single image feature extractor is usually not able to extract sufficient visual information for classification. The recognition system needs to employ different types of feature extractors and combine the information explored by these

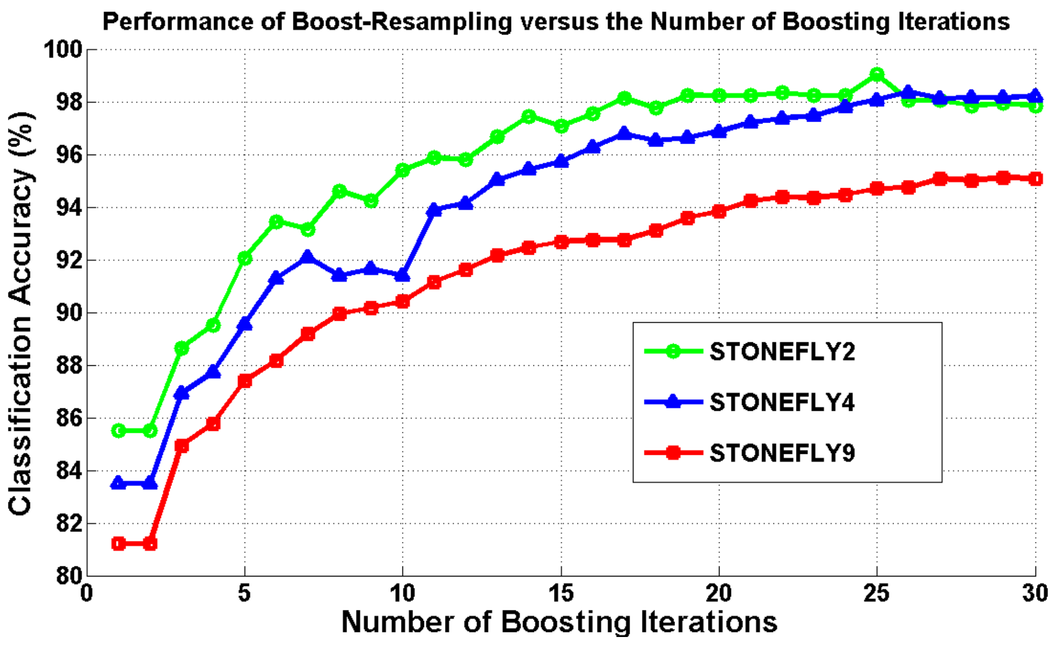


Figure 22. The performance of Boost-Resample non-redundant codebook learning algorithm on stonefly dataset versus the number of boosting iterations.

extractors to achieve high performance. According to our experiments, the combination of the structure-based PCBR regions and intensity-based interest regions (Hessian-Affine, salient regions) works quite well. (2) Visual codebooks can provide compact and informative image representations for stoneflies. Their performance can be significantly improved by introducing non-redundancy in the construction process (see Section 4.5). (3) Due to the small between-class differences of the stoneflies, the image classifiers based on discriminative feature selection is preferred because they rely on a small number of discriminative low-level image features or mid-level image attributes for high-precision classification. Ensemble learning techniques can be employed to reduce the variance of the classifier. For example, the boosting decision stumps (see Section 3.3), bagged decision lists (see Section 4.4) and the bagged decision trees (see Section 4.5) classifiers all achieved high performance on the stonefly recognition problem.

4 Visual Codebooks for Object Recognition

4.1 Overview

As discussed in Chapter 2, interest region detectors and descriptors have been computed to robustly represent images that are subject to noise and deformations. Each image is thus represented as a bag of interest region descriptors. In such tasks, the object recognition problem reduces to the problem of classifying a bag of low-level features (interest region descriptors) into one or a subset of the possible object classes. Visual codebooks provide a way of generalizing a bag of low-level features (descriptors) to a fixed-length attribute vector, to which standard classifiers can be directly applied. A visual codebook can be constructed either by unsupervised clustering (see Section 2.3.2) or by discriminative learning (see Section 2.3.3).

Generative visual codebooks have shown good performance on various visual object categorization tasks. We employ this unsupervised codebook learning method (Larios et al. 2008) to the difficult stonefly recognition problem. This method will be introduced in Section 4.2. Despite the high performance of these generative visual codebooks for object recognition, their discriminative characteristics are not well-studied by previous researchers (see Section 2.2). So we present the Maximum Mutual Information (MMI) evaluation criterion (Zhang and Deng 2008) to measure the discriminative power of codebook entries quantitatively. This evaluation method and evaluation results are introduced in Section 4.3.

An advantage of unsupervised codebook learning is that there is little or no risk of overfitting the training images. But the generative codebook is not designed to discriminate between the different object categories in the specific problem; discriminative learning is mainly restricted to the final image classifier based on the codebook. Therefore, we present an Iterative Discriminative Clustering (IDC) algorithm (Zhang and Dietterich 2008) to introduce discriminative mechanisms into the visual codebook learning process. This algorithm is combined with a bagged decision lists classifier which searches for a small number of codewords so that recognition is very efficient. These methods are introduced in Section 4.4.

All the codebook learning methods introduced in Section 2.3 build one codebook by performing a single clustering on the low-level features. Instead we design the learning algorithm to build multiple visual codebooks that are non-redundant in discriminative power (Zhang et al. 2009). Instantiation of this framework is successfully applied to stonefly recognition tasks. This method is presented in Section 4.5.

4.2 Generative Visual Codebook Learning for Stonefly Recognition

Our approach of generative visual codebook learning (Larios et al. 2008) follows the unsupervised learning approaches introduced in Section 2.3.2, but with several modifications and extensions. The key component of the system is the generative visual codebook learning algorithm based on Gaussian Mixture Model (GMM). It will be introduced in the following section.

4.2.1 Generative Visual Codebook Learning

Suppose there are F different combinations of detector/descriptor applied to the images. We construct a separate codebook for each detector/descriptor combination f and each class c. Let $\boldsymbol{V}_{f,c}$ be the low-level features extracted from detector/descriptor combination f in all cluster-set images from class c. We fit a Gaussian mixture model (GMM) to $\boldsymbol{V}_{f,c}$ via the Expectation-Maximization (EM) algorithm (Duda et al. 2001). A GMM with K components has the form

$$p(\boldsymbol{v}) = \sum_{k=1}^{K} C_{f,c,k}(\boldsymbol{v} \mid \boldsymbol{\mu}_{f,c,k}, \boldsymbol{\Sigma}_{f,c,k}) \qquad (7)$$

where $\boldsymbol{v}$ denotes a descriptor vector, the component probability distribution $C_{f,c,k}$ is a multivariate Gaussian density function with mean $\boldsymbol{\mu}_{f,c,k}$ and covariance matrix $\boldsymbol{\Sigma}_{f,c,k}$ (constrained to be diagonal). Each fitted component of the GMM defines one of K codewords. Given a new descriptor vector $\boldsymbol{v}$, we compute the corresponding codeword $k = key_{f,c}(\boldsymbol{v})$ by finding the k that maximizes $p(\boldsymbol{v} \mid \boldsymbol{\mu}_{f,c,k}, \boldsymbol{\Sigma}_{f,c,k})$. Note that we disregard the mixture probabilities $P(k)$. This is equivalent to mapping $\boldsymbol{v}$ to the nearest cluster center $\boldsymbol{\mu}_k$ under the Mahalobis distance defined by $\boldsymbol{\Sigma}_k$.

We initialize EM by fitting each GMM component to each cluster obtained by the *k*-means algorithm (Csurka et al. 2004; Duda et al. 2001). The *k*-means algorithm is initialized by picking random elements. The EM algorithm iterates until the change in the fitted GMM error from the previous iteration is less than 0.05% or until a defined number of iterations is reached.

After building the codebook, we next construct a set of training examples by applying the F detector/descriptor combinations to each training image. Then we follow the "histogram of occurrence" mapping method described in Section 2.3.4. We map each feature extracted from detector/descriptor combination f to the nearest codeword (as describe above) for each class c using $key_{f,c}(\boldsymbol{v})$. We accumulate the codewords to form a histogram $\boldsymbol{H}_{f,c}$ and concatenate these histograms to produce the final attribute vector. With F detectors, C classes, and K mixture components, the number of attributes A in the final attribute vector (i.e., the concatenated histogram) is $F \times C \times K$.

4.2.2 Experimental Results

As presented in Section 3.5.1, we apply our generative visual codebook learning algorithm to the challenging STONEFLY2 and STONEFLY4 problems. Three region detectors are employed: Hessian-Affine regions (Mikolajczyk and Schmid 2004), Kadir's salient regions (Kadir et al. 2004) and PCBR regions (see

Table 6. Comparison of the performance of the generative codebook learning method (Concatenate Histograms Feature & Logistic Model Trees (CHF&LMT)) and Opelt's method (2006) on STONEFLY2 problem.

Hessian	*Kadir*	*PCBR*	Accuracy (%)	
			Opelt	CHF&LMT
√			60.59	70.10
	√		62.63	70.34
		√	67.86	79.03
√	√	√	70.10	79.37

Section 3.2). All detected regions are described by SIFT descriptors (Lowe 2004). The concatenate histograms feature vectors are used to learn bagged logistic model trees (Landwehr et al. 2005) as the image classifier. This recognition system achieves good performance on the stonefly recognition problems, as shown in Table 4 in Section 3.5.1. It also outperforms state-of-the-art recognition method (Opelt et al. 2006) on the STONEFLY2 problem, as summarized in Table 6.

4.3 Understanding Generative Visual Codebooks using MMI Curves

4.3.1 Motivation

As introduced in Section 2.3, visual codebooks have been successfully applied to object recognition as a mid-level process. This approach employs clusters of region descriptors as the codewords in the visual codebook. The recognition task is then accomplished by manipulating the codewords and selecting the most discriminative ones to build the final image classifier. Different combinations of interest region detectors and region descriptors will produce different pools of low-level features to build the codebooks. Ideally, all the codewords should be consistent and informative to make classification a trivial task. The choice of low-level feature extractor has significant impact on the performance of recognition approaches (Dorko and Schmid 2004; Opelt et al. 2006; Mikolajczyk et al. 2005b). But it is not always obvious which detector and descriptor is the best for a given problem. Usually, the choice is made purely empirically. Given an object recognition problem, it would be much more rational to experiment only with low-level features that are promising for the problem, rather than trying every possibility.

Different evaluation criteria (Dorko and Schmid 2004; Mikolajczyk et al. 2005b) have been proposed to measure the discriminative ability of descriptor clusters, as introduced in Section 2.2.3. Motivated by previous work and the successful discriminative feature selection algorithm in Opelt et al. (2006), we introduce the Maximum Mutual Information (MMI) evaluation criterion (Zhang and Deng 2008), which measures the discriminative power of codewords quantitatively. This method is introduced in Section 4.3.2. MMI evaluation is closely related to the classification of image instances in the recognition task. It can be performed on any object recognition dataset efficiently without the requirement for prior knowledge of homographies. The MMI curves can clearly reveal the characteristics of codebooks for the specific object recognition problem. Additionally, comparison results are valuable guidelines for the design of the image classifier. Visual codebooks built from state-of-the-art interest region detectors are evaluated on benchmark datasets. The results are summarized in Section 4.3.3. The results can help future researchers to select suitable detectors for similar object recognition problems.

4.3.2 MMI Scores and MMI Curves

Generative codebook learned by GMM algorithm

Given a binary object recognition dataset composed of object (positive) images and background (negative) images, the positive images are partitioned into two disjoint sets, one is called the clustering set, denoted as I_C. The other positive set is combined with all negative images to form the evaluation set I_E. Then a specific interest region detector is applied to all the images. For each detected region, a SIFT descriptor (Lowe 2004) is computed to produce the clustering descriptor vectors F_C and evaluation descriptor vectors F_E.

Like generative codebook learning in Section 4.2, our method first fits a Gaussian Mixture Model (GMM) to the descriptor vectors in F_C. Each cluster C_k is described by a d-dimensional mean vector $\boldsymbol{\mu}_k$ and a $d\times d$ diagonal covariance matrix $\boldsymbol{\Sigma}_k$. In our experiments, the number of clusters K is set to 50.

MMI Score

Given: a cluster C_k: $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, the *evaluation* set I_E of m images, the class labels of evaluation images $L_E = (l_1, \ldots, l_i, \ldots , l_m)$, with $l_i \in \{+1,-1\}$, and the SIFT vectors of evaluation images $F_E = (F_1, \ldots, F_i, \ldots , F_m)$, we would like to evaluate the discriminative power of cluster C_k. In other words, that is, how well does the cluster reveal the categories of evaluation images. This is done by employing cluster C_k to classify the evaluation images and searching for the maximum mutual information (MMI) (MacKay 2003) between the classification results and the true class labels. The MMI score is treated as the evaluation score for C_k. In order to classify the evaluation images using cluster C_k, we employ an approach similar to the *Weak-Hypotheses-Finder* algorithm in Opelt et al. (2006). First, we calculate the distance from C_k to all the evaluation images. For cluster C_k: $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ and image I_i, the distance between them is:

$$a_{k,i} = \min(d(F_i, C_k)) = \min_j((\boldsymbol{v}_{ij} - \boldsymbol{\mu}_k)^t \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{v}_{ij} - \boldsymbol{\mu}_k)) \tag{8}$$

where $\boldsymbol{v}_{ij} \in F_i$ is the SIFT descriptor vector computed from region $j = 1, \ldots, N_i$. N_i is the total number of detected regions in image I_i.

Given the distances between C_k and evaluation images ($a_{k,1}, \ldots, a_{k,i}, \ldots, a_{k,m}$), all the evaluation images can be sorted based on these distances. That is, we find a permutation $\boldsymbol{\pi} = (\pi(1), \ldots, \pi(i), \ldots, \pi(m))$ such that

$$a_{k,\pi(1)} \leq \ldots \leq a_{k,\pi(i)} \leq \ldots \leq a_{k,\pi(m)} \tag{9}$$

Then the sorted class labels are

$$\boldsymbol{l}_{k,\boldsymbol{\pi}} = (l_{k,\pi(1)}, \ldots, l_{k,\pi(i)}, \ldots, l_{k,\pi(m)}),\ l_{k,\pi(i)} \in \{+1,-1\} \tag{10}$$

The sorted label array illustrates the discriminative power of cluster C_k. A perfect cluster should have all the positive images (+1) ranked first followed by all the negative images (−1); while a poor cluster can never discriminate between them, so gives randomly ordered labels. From the view of Information Theory (MacKay 2003), the sorted label array $\boldsymbol{l}_{k,\boldsymbol{\pi}}$ indicates how much information the distance-based sorting can tell about the class labels. This is quantitatively measured by the maximum mutual information (MMI) (Dorko and Schmid 2004) between the sorted array and the true class labels:

$$MMI_k = \max_s (MI(\boldsymbol{l}_{k,\boldsymbol{\pi}}, s)). \tag{11}$$

$MI(\boldsymbol{l}_{k,\boldsymbol{\pi}}, s)$ calculates the mutual information between the classification results $C_{k,s}$ and the true image class L_E. $C_{k,s}$ is given by a decision stump (Opelt et al. 2006) which sets the threshold at position s ($1 \leq s \leq m$) in the sorted label array and classifies the images before s to be positives and those after s to be negatives. The search for the maximum in Equation (11) can be sped up by exploiting the fact that the maximum can only possibly be obtained between +1 followed by a −1 in the sorted label array. Using the sorted label array $\boldsymbol{l}_{k,\boldsymbol{\pi}}$, the mutual information Equation (11) can be calculated as in Dorko and Schmid (2004) (See Equation (1) in Section 2.4). MMI scores measure the discriminative power of codewords. A perfectly discriminative codeword is assigned a full MMI score of 1.0, while a non-discriminative codeword will have a score near 0.

The MMI Curve

For each detector, we sort the codewords into decreasing order of their MMI scores and plot the sorted scores as a function of their position in the ordering. We call such a plot an MMI curve (see Figure 23 − Figure 25). A detector's performance and suitability for object recognition can be measured by the Area-Under-Curve (AUC) and the shape of the MMI curve. The MMI curve for a perfect detector is a horizontal line at full score (this also gives maximum AUC). If a detector produces an MMI curve that is above average but relatively flat, such as the MMI curve for DoG (Lowe 2004) on the cars dataset in Figure 23, it indicates that most of the detected regions are fairly distinctive and discriminative and only a few of the detections are very noisy. Under this situation, classifiers that assume equal contributions from all features, such as Nearest Neighbor and Neural Networks, are probably able to tolerate the noise and give high recognition performance. On the other hand, consider a detector that generates a curve that has very high scores for the top ranking clusters but relatively low scores for the following clusters. For example, see the MMI curve for PCBR detector (noted as curvilinear) on stoneflies in Figure 19. This curve shows that the detector can only find a few highly distinctive and discriminative regions while at the same time producing many uninformative detections. In this case, the classifiers mentioned above will probably fail. Instead, this detector may work well with algorithms that perform discriminative feature selection (see Section 2.4). These algorithms are able to achieve high classification accuracy using only a small part of the relevant features. In summary, MMI curves are valuable guidelines for the selection of detectors and the design of the image classifiers.

4.3.3 Evaluation Results and Conclusions

We evaluate the performance of visual codebooks built from several state-of-art interest region detectors: (1) Harris-Laplace, Hessian-Laplace, and their affine-invariant versions: Harris-Affine and Hessian-Affine from Mikolajczyk and

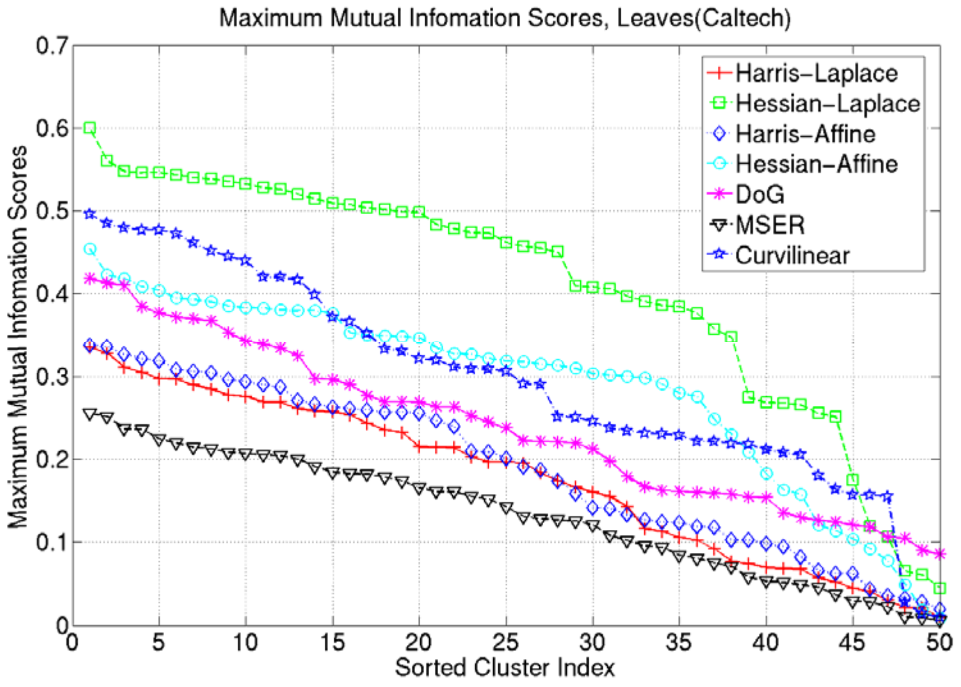
Schmid (2004); (2) Difference-of-Gaussian (DoG) detector from Lowe (2004) ; (3) Maximally Stable Extremal Regions (MSER) from Matas et al. (2002); and (4) PCBR Regions (denoted as Curvilinear in figures) detector (see Section 3.2). These detectors have been described in previous sections.

We experimented with four benchmark object recognition datasets: Caltech (*Caltech*), Caltech 101 (*Caltech101*), GRAZ (*GRAZ*) and Stonefly larvae (*Bug-ID*) (see Chapter 3). Some objects are highly textured (e.g. leopards), some are structured (e.g. leaves); and these datasets differ greatly in their complexity. Each experiment is repeated 10 times with random selection of the clustering and evaluation sets, and the results are the average of 10 iterations. The MMI curves are shown in Figure 23 − Figure 25.

We can see that all the detectors work fairly well on simple objects, such as leaves. More than half of the codewords have mutual information above 0.15, and some codewords achieve very high mutual information scores (> 0.5). But for relatively complex problems, such as leopards and stoneflies, the performance of detectors inevitably degrades a lot. Most of the codewords have mutual information scores below 0.05.

Different detectors exhibit different characteristics and performance for different object classes, which are illustrated by the shapes of their MMI curves.

DoG has the best overall performance for most of the datasets (except leaves set). This demonstrates DoG’s ability to detect discriminative regions in natural scenes, and its robustness to various planar transformations and limited view changes. In Figure 24, we can see that on the leopards set, DoG performs far better than other detectors. On the leaves set, DoG is outperformed by the Hessian detectors and the PCBR detector.



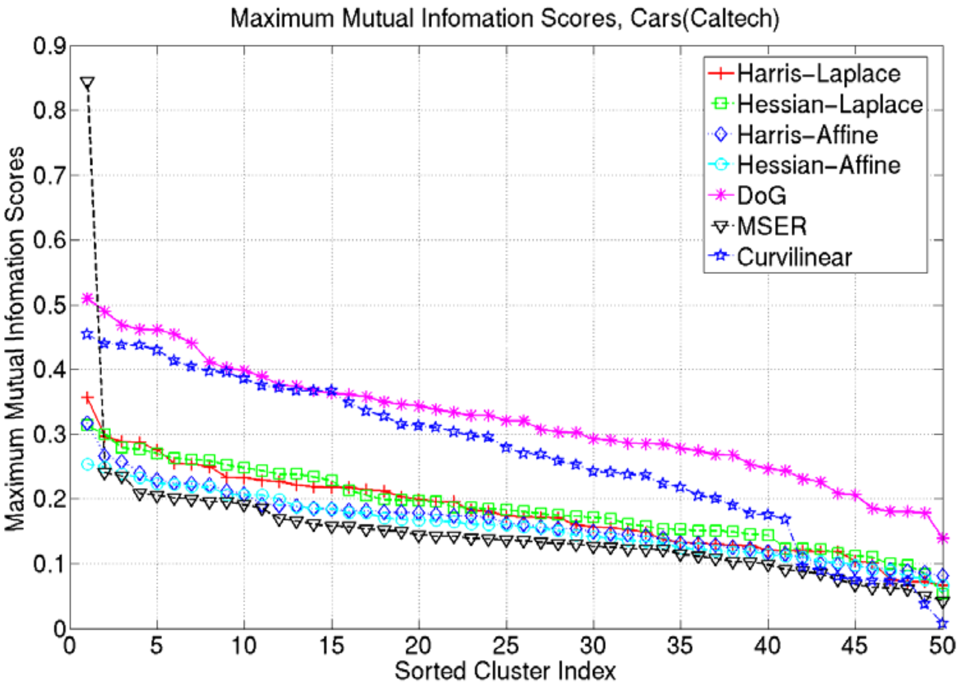
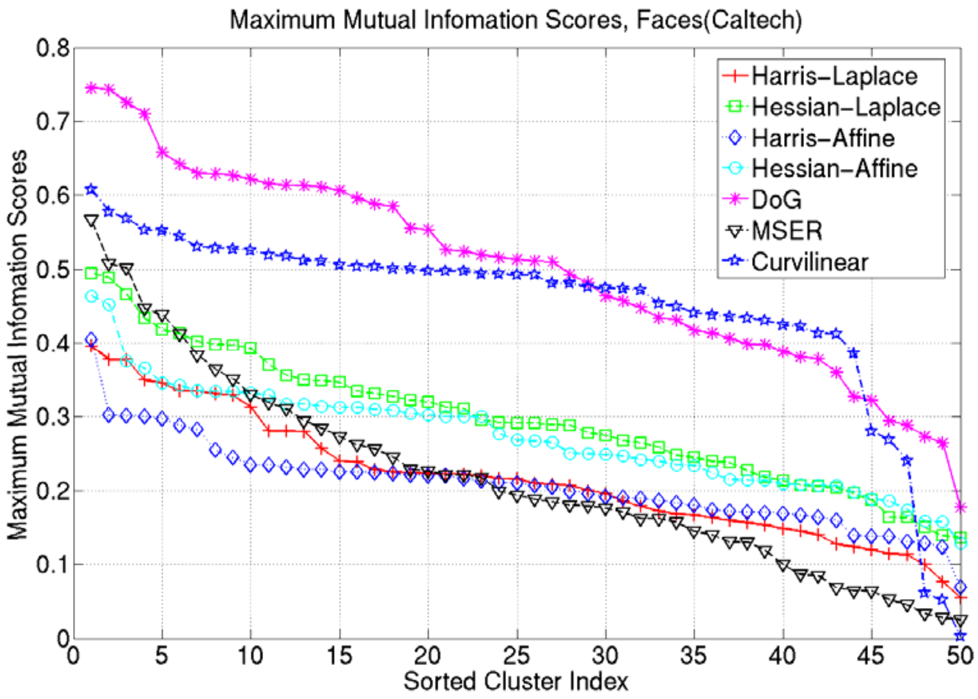


Figure 23. MMI curves of state-of-the-art detectors on the leaves and cars datasets.



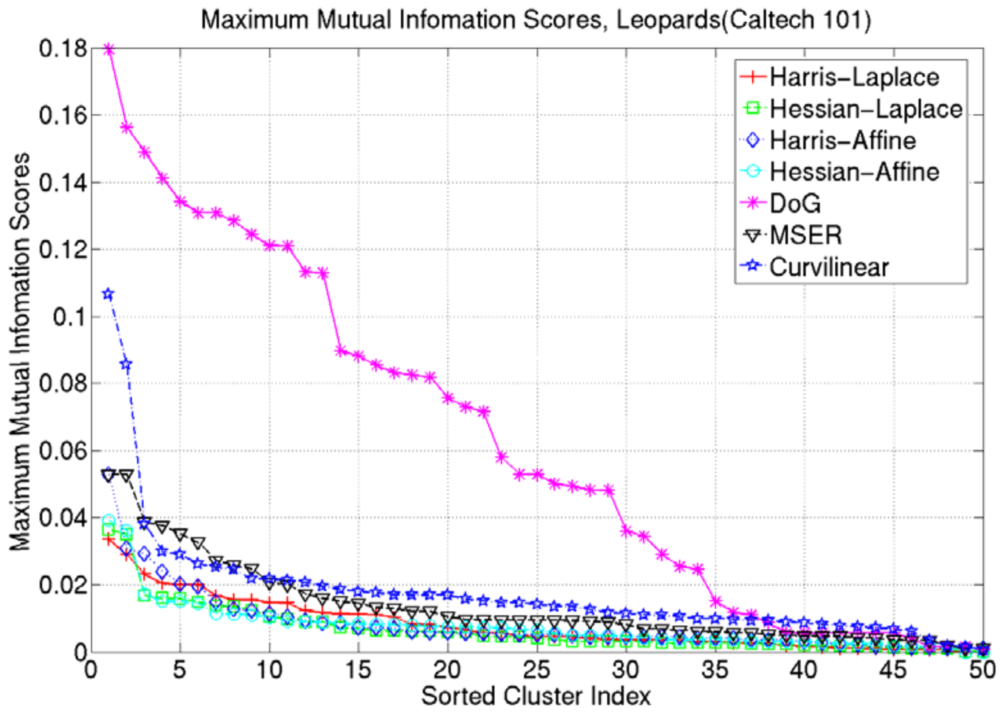


Figure 24. MMI curves of state-of-the-art detectors on the faces and leopards datasets.

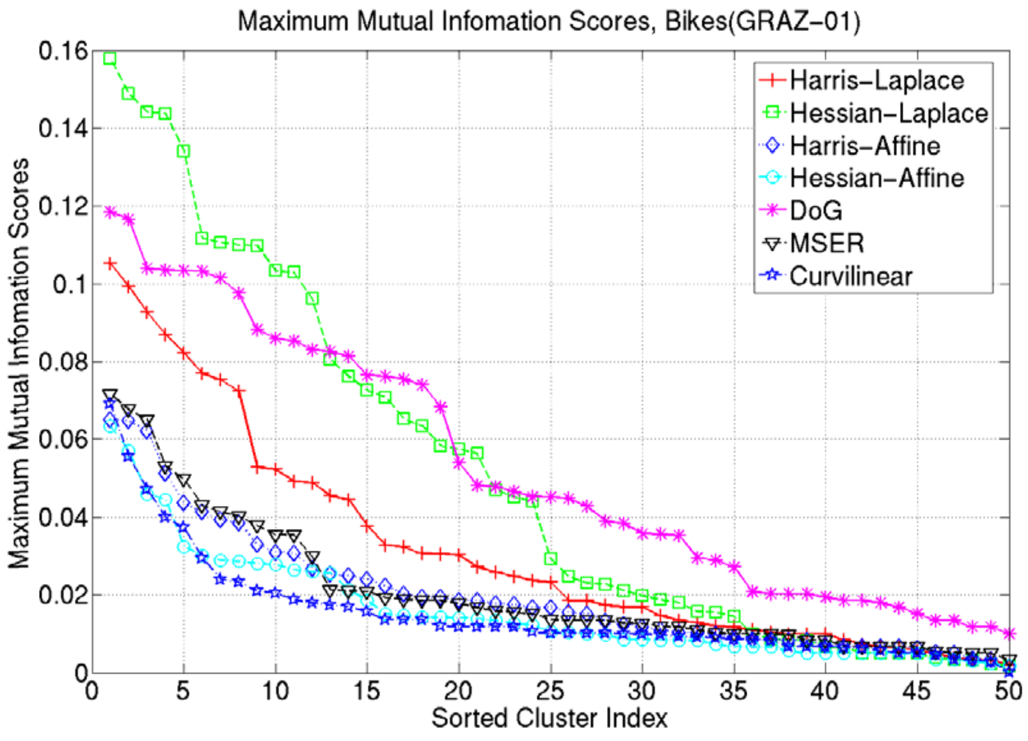


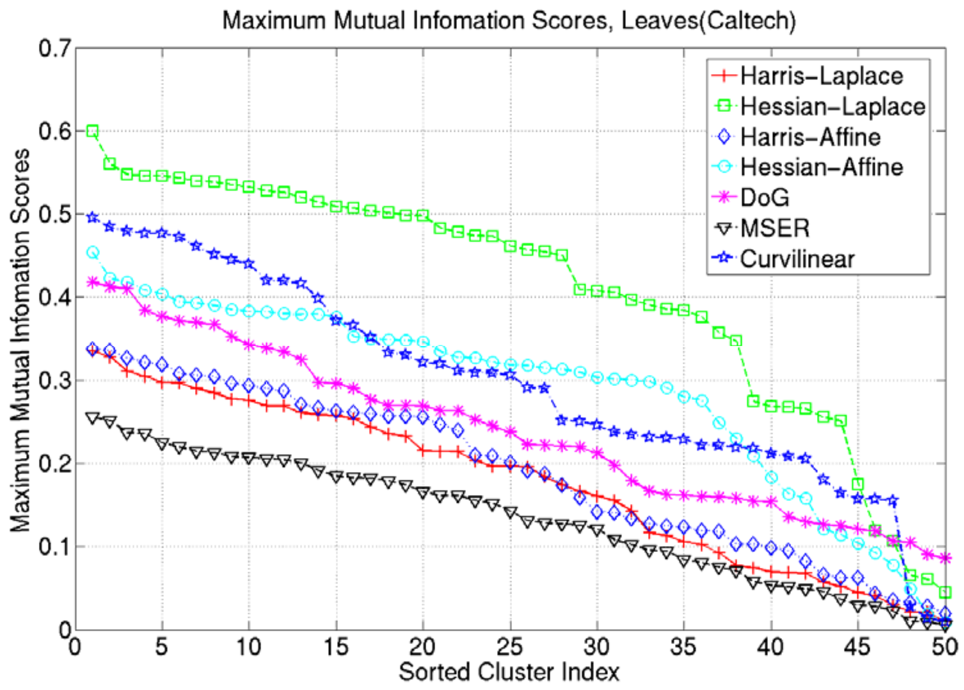
Figure 25. MMI curves of state-of-the-art detectors on the bikes dataset.

The PCBR detector has evaluation scores above average on all the object classes. It works especially well on highly structured objects, such as leaves and cars. PCBR is usually able to find several highly distinctive and discriminative patterns, e.g. on the leopards and stoneflies sets. This implies its utility for feature selection methods. Its high-performance in combination with other intensity-based local detectors has been demonstrated on the stonefly recognition dataset, as presented in Chapter 3.

On most of the datasets, Harris-Laplace and Hessian-Laplace have similar MMI curves; also do Harris-Affine and Hessian-Affine. This can be explained by their similar local intensity-based detecting principles. But on leaves set in Figure 23, the Hessian detectors score much higher than the corresponding Harris detectors. Harris-Laplace works much better than all the other detectors in this family.

In addition to evaluating the relative performance of detectors, MMI curves also reveal the intrinsic characteristics of the visual codebooks. MMI curves for DoG are fairly good and quite flat; this indicates that most of the codewords are informative, so they can be recommended for use with the classifiers that assume equal contribution from all features; PCBR has a similar MMI curve on cars set, as do the Hessian detectors on the leaves set. On the other hand, we also notice that some other detectors produce quite different MMI curves on some datasets. MSER is not stable on all the object classes in the sense that most of the codewords score relatively low, while it has the ability to extract a few highly distinctive and discriminative codewords in cars set. As shown in Figure 23, its MMI curve starts at 0.82, which is about 0.3 units higher than any other method. Similarly for the PCBR and Hessian-Affine detectors on the Stoneflies set. Their MMI curves all start with very high scores but then soon drop down to noise detections. For these codebooks, classifiers based on discriminative feature selection are more promising. So even if a detector fails to give stable detections (low repeatability), it is still possible that it can produce a small number of highly distinctive and discriminative codewords if it fits the object class. In summary, the characteristics of the visual codebooks generated by different interest region detectors can be explored directly by the shapes of their MMI curves.

We also extensively studied the robustness of our MMI evaluation criteria to several key factors: (1) the density of detection; (2) the size of regions and (3) the size of the codebook. The MMI criterion is robust to these factors in that the relative ranking of the detectors are mostly invariant to different settings. For example, we show the evaluation results on leaves set with K=100 and K=20 in Figure 26. Comparing the curves, we can see little difference between the rankings.



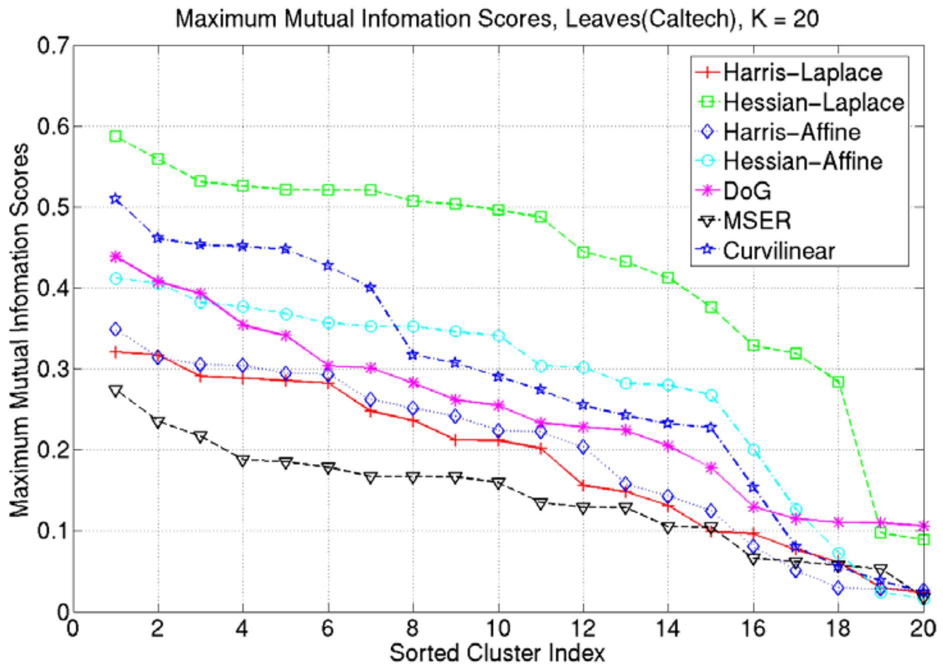


Figure 26. MMI curves of state-of-the-art detectors on the leaves dataset with codebooks of different sizes K: Top: K = 100; Bottom: K = 20.

Table 7. Classification accuracies of generative codebooks learned from different detector/descriptor combinations.

Class	*Har-Lap*	*Hes-Lap*	*Har-Aff*	*Hes-Aff*	*DoG*	*MSER*	*PCBR*
Leaves	98.9	**99.6**	98.5	99.3	99.3	92.7	**99.6**
Cars	96.4	96.4	96.4	94.1	**98.3**	97.2	93.6
Faces	97.05	97.9	96.8	98.8	**99.7**	99.1	**99.7**
Leopards	80.65	80.6	79.6	80.7	79.4	80.8	**82.1**
Bikes	72.05	**75.6**	71.6	67.3	70.3	69.4	61.3
Stoneflies	80.8	78.7	70.2	80.9	**83.0**	70.2	69.4

To validate the MMI evaluation results, we also employ the boosting feature selection classifier introduced in Section 2.4 (Opelt et al. 2006) to select the high-scoring codewords and test their combined classification accuracy on real-world problem. The *evaluation* set is divided into two non-overlapping sets. One set is used as a *training* set to train the image classifier; the other is used for *testing*. Then decision stumps are learned on the *training* set as described in Section 3.3.2. Each iteration of AdaBoost searches among the unused codewords and selects the one that has the highest MMI score. The boosted decision stumps are then applied to the testing images to evaluate the performance of the detector. The results are given in Table 7. We can see that the results are consistent to the comparison of the detectors using the MMI curves.

4.4 Generative/Discriminative Learning of Visual Codebooks

4.4.1 Motivation

There are two major challenges for any approach to object recognition: (a) Small training sets. In most benchmark datasets, it is usual to have fewer than 200 training examples of each object class. This is one reason that generative methods for codebook learning (see Section 2.3.2 and Section 4.2) have been popular: they are less prone to overfit the data. (b) Low signal-to-noise ratio. Due to the complexity of object recognition problems, even using state-of-the-art interest region detectors and region descriptors, only a small fraction of the extracted descriptors are discriminative, while the others are noisy (see the MMI curves in Figure 23 – Figure 25 in Section 4.3). This is the main motivation for developing discriminative visual codebook learning methods that exploit the class labels to identify discriminative features. The challenge is to do this without causing overfitting.

As introduced in Section 2.3.3, discriminative mechanisms have been employed to learn visual codebooks. In this dissertation, we introduce a new efficient codebook learning method called the Iterative Discriminative Clustering (IDC) algorithm (Zhang and Dietterich 2008) that combines the best of generative and discriminative learning to address the challenges in object recognition. This algorithm is presented in Section 4.4.2. The learned codebook benefits from generative initialization in its robustness to overfitting and obtains higher test set accuracy from discriminative learning. Unlike previous methods, which treat all elements of the descriptor vector as equally important, our method learns a cluster-specific full-rank distance metric that improves cluster generalization and discriminative power. In addition, we employ an efficient rule learning algorithm, decision lists, to create the final classifier. This algorithm is described in Section 4.4.3. This classifier can achieve high performance by identifying the logic conjunctions of small number of informative codewords from the highly noisy feature pool. Extensive experiments on three well-recognized object recognition benchmark datasets show performance that matches or exceeds state-of-the-art codebook and instance selection based object recognition approaches. The results are summarized in Section 4.4.4.

4.4.2 Generative Initialization and Discriminative Refinement of Codebooks

Generative initialization of the codebook

Similar to the generative visual codebook described in Section 4.2, a discriminative visual codebook *DVC* has the form

$$DVC = \{\boldsymbol{E}_1, \cdots, \boldsymbol{E}_k, \cdots, \boldsymbol{E}_K\} = \{<\boldsymbol{x}_1, \boldsymbol{W}_1>, \cdots, <\boldsymbol{x}_k, \boldsymbol{W}_k>, \cdots, <\boldsymbol{x}_K, \boldsymbol{W}_K>\}; \quad (12)$$

where $\boldsymbol{x}_k$ and $\boldsymbol{W}_k$ are the representative value and the distance metric, respectively, of codeword $\boldsymbol{E}_k$. A separate codebook $DVC_{f,c}$ is learned for each detector/descriptor combination f and each object class c. The codebook $DVC_{f,c}$ is initialized by k-means clustering on the region descriptor vectors of type f from the training images of class c. Full rank covariance matrices and the Mahalanobis distance are employed during clustering. Each representative value, $\boldsymbol{x}_k$, is initialized to the corresponding cluster center, and each distance metric, $\boldsymbol{W}_k$, is initialized to the inverse of the corresponding covariance matrix. So the distance from an initial codeword $\boldsymbol{E}_k$ to a descriptor vector $\boldsymbol{x}$ is measured by the Mahalanobis distance metric (Duda et al. 2001):

$$d(\boldsymbol{E}_k, \boldsymbol{x}) = ((\boldsymbol{x}_k - \boldsymbol{x})^t \boldsymbol{W}_k (\boldsymbol{x}_k - \boldsymbol{x}))^{1/2} \quad (13)$$

Iterative discriminative clustering (IDC) algorithm

The main contribution of our codebook learning method is to apply supervised learning directly to construct problem-specific discriminative codebooks for image classification. Our *Iterative Discriminative Clustering* (*IDC*) algorithm combines and adapts the idea of the *EM-DD* algorithm (Zhang &

Goldman 2001) for multiple-instance learning and *Relevant Component Analysis* (*RCA*) (Shental et al. 2002) for distance metric learning.

The IDC algorithm is applied separately to each codeword $\boldsymbol{E}_k = < \boldsymbol{x}_k , \boldsymbol{W}_k >$ in a codebook. Let c be the class of the codebook. Consider a training image i represented by its bag of region descriptor vectors $\boldsymbol{B}_i$. Let $\boldsymbol{p}$ be the descriptor vector in $\boldsymbol{B}_i$ that is closest to $\boldsymbol{E}_k$ according to $d(\boldsymbol{E}_k, \boldsymbol{B}_{ij})$; we will call $\boldsymbol{p}$ the nearest neighbor point from image i. Let $\{\boldsymbol{NN}^+\}_k$ be the set of all nearest neighbor points $\boldsymbol{p}$ for codeword k drawn from positive examples of class c, and let $\{\boldsymbol{NN}^-\}_k$ be the corresponding set drawn from negative training examples (i.e., examples of other classes $c' \neq c$). If $\{\boldsymbol{NN}^+\}_k$ is compact and well-separated from $\{\boldsymbol{NN}^-\}_k$, then $\boldsymbol{E}_k$ is a compact, discriminative entry, because it has consistent nearest neighbor points in images of class c, and it is far away from the images of other classes. Otherwise, if $\{\boldsymbol{NN}^+\}_k$ has high variance or $\{\boldsymbol{NN}^+\}_k$ and $\{\boldsymbol{NN}^-\}_k$ overlap, then $\boldsymbol{E}_k$ is suboptimal in term of discrimination, and we seek to improve its performance with supervised learning.

The idea of the learning algorithm is to locally adapt the representative value and the distance metric of entry $\boldsymbol{E}_k$ with the goal of making it discriminative. We limit the adaptation to the local neighborhood of entry $\boldsymbol{E}_k$ to avoid situations in which all codewords converge to the same global maximum and the learned codebook has low discriminative power. The pseudo-code for the IDC algorithm is given in Figure 27. IDC algorithm iterates between the following two steps: in the "nearest neighbors search" step, the nearest neighbor point sets $\{\boldsymbol{NN}^+\}_k$ and $\{\boldsymbol{NN}^-\}_k$ are computed for codeword $\boldsymbol{E}_k$ based on the representative value and distance metric from the previous iteration. In the following "entry updates" step, the representative value of $\boldsymbol{E}_k$ is updated to be the mean of the positive nearest neighbor points; and the Relevant Component Analysis (RCA) algorithm (Shental et al. 2002) is employed to learn a new distance metric to sphere and better separate the point sets $\{\boldsymbol{NN}^+\}_k$ and $\{\boldsymbol{NN}^-\}_k$.

Distance metric learning using RCA algorithm

RCA learns a linear transformation to assign large weights to the relevant dimensions and small weights to the irrelevant dimensions. Here the "relevant dimensions" are the dimensions that help to discriminate between the sets: $\{\boldsymbol{NN}^{+}\}_k$ and $\{\boldsymbol{NN}^{-}\}_k$. The learned entry $\boldsymbol{E}_k$ is analogous to the "weighted cluster" in Domeniconi et al. (2007), such that the data points in a cluster are tightly grouped according to the L_2 norm distance weighted by $\boldsymbol{W}_k$. The distance between the learned entry $\boldsymbol{E}_k$ and an image indicates the confidence for whether the image contains the object part corresponding to $\boldsymbol{E}_k$. This distance can be used as a feature value for image classification, as shown in Equation (17).

Convergence criteria

The adaptation of a codeword is limited to its local neighborhood using early-stopping conditions. The two steps of the IDC algorithm iterate until one of the following two stopping conditions is satisfied: 1. it reaches the maximal number of iterations: *MaxI*; we set it small to encourage local maxima. 2. When the updated codeword is not more discriminative than the original one. The discriminative power is measured by the difference between the average distances from the codeword to the nearest neighbor points in $\{\boldsymbol{NN}^{+}\}_k$ and $\{\boldsymbol{NN}^{-}\}_k$.

After the algorithm stops, we also verify the relative distance from the new representative value, $\boldsymbol{x}_k^*$, to the original one, $\boldsymbol{x}_k$, and only keep the new entry if it is within a certain neighborhood of $\boldsymbol{x}_k$:

$$\frac{\left\|\boldsymbol{x}_k^* - \boldsymbol{x}_k\right\|^2}{\left\|\boldsymbol{x}_k\right\|^2} \le T_L \tag{14}$$

Codebook compression

The IDC algorithm described above is applied to each codeword in the codebook. It is usual that this causes several codewords to converge to the same point. In this case, only one of them is kept. Learned codewords that are close to each other are merged if the following two conditions are met:

```
Input: Bags of descriptor vectors of m training images:
  $\boldsymbol{D} = \{<\boldsymbol{B}_1, l_1>, \ldots, <\boldsymbol{B}_i, l_i>, \ldots, <\boldsymbol{B}_m, l_m>\}$;
  Initial codebook of class $c$: $DVC =$
  $\{<\boldsymbol{x}_1, \boldsymbol{W}_1>, \ldots, <\boldsymbol{x}_k, \boldsymbol{W}_k>, \ldots, <\boldsymbol{x}_K, \boldsymbol{W}_K>\}$;
Learning:
for ($k = 1$; $k \le K$; $k$ ++)
  $\boldsymbol{E}_k = <\boldsymbol{x}_k, \boldsymbol{W}_k>$;     // initial codeword
  while (not converged)
    $\{\boldsymbol{NN}^+\}_k = \{\}$;  $\{\boldsymbol{NN}^-\}_k = \{\}$;
    for (each bag $\boldsymbol{B}_i \in \boldsymbol{D}$)
      // nearest neighbors search
      $\boldsymbol{p} = \arg\min_{\boldsymbol{B}_{ij} \in \boldsymbol{B}_i} d(\boldsymbol{E}_k, \boldsymbol{B}_{ij})$;
      if ($l_i == c$) then Add $\boldsymbol{p}$ to $\{\boldsymbol{NN}^+\}_k$;
      else Add $\boldsymbol{p}$ to $\{\boldsymbol{NN}^-\}_k$;
    $\boldsymbol{x}_k$ = Mean ($\{\boldsymbol{NN}^+\}_k$);      // entry updates
    $\boldsymbol{W}_k$ = RCA ($\{\boldsymbol{NN}^+\}_k$, $\{\boldsymbol{NN}^-\}_k$);
    $\boldsymbol{E}_k = <\boldsymbol{x}_k, \boldsymbol{W}_k>$;
```

Figure 27. Pseudo-code for the Iterative Discriminative Clustering algorithm.

$$\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2}{\|\boldsymbol{x}_1\|^2 + \|\boldsymbol{x}_2\|^2} \le T_{s_1} \tag{15}$$

$$\frac{\mathrm{norm}(\boldsymbol{W}_1 - \boldsymbol{W}_2)}{\mathrm{norm}(\boldsymbol{W}_1) + \mathrm{norm}(\boldsymbol{W}_2)} \le T_{s_2} \tag{16}$$

where the norm is the Frobenius-norm. This process eliminates redundancy in the codebook and improves classification efficiency.

Parameter setting

We tested the performance of the recognition system for different settings of the parameters. The performance is quite robust. For example, changing *MaxI* = 3 to *MaxI* = 10 makes less than 0.5% point difference in the classification error rates on the Caltech motorbikes *vs*. general background problem. The size of the initial codebook *K*, which is the most sensitive parameter, also has little impact on the classification accuracy (see Figure 30). Thus we adopted a single set of parameter values for all experiments:

$K = 200$;

$MaxI = 5$;

$T_L = 50$ (reject fewer than 10% of new codewords);

$T_{s_1} = T_{s_2} = 0.01$ (merge 10% − 50% of the codewords).

Computational complexity

The IDC algorithm requires $O(K|\boldsymbol{D}|)$ per iteration, where K is the size of the initial codebook, which is usually set at several hundred, and $\boldsymbol{D}$ is the set of descriptor vectors extracted from training images. Typically, $K << |\boldsymbol{D}|$. So the IDC algorithm is much more efficient than the approach in Opelt et al. (2006) whose complexity is $O(|\boldsymbol{D}|^2)$. Furthermore, since we use sparsely detected interest regions, $|\boldsymbol{D}|$ is much smaller than for densely sampled regions. Thus, our codebook learning algorithm is also more efficient than the algorithms based on dense sampling (Jurie and Triggs 2005; Perronnin 2008). Our nonoptimized MATLAB implementation takes less than one hour to learn a 200-words codebook from ~100,000 regions detected from ~1,000 images.

Feature mapping based on the learned codebook

As in the recognition system described in Section 4.2, a separate codebook is learned for each object class (including the background class) and each detector/descriptor combination. All these separate codebooks are concatenated to construct the final codebook; suppose it has M codewords. Then a new image $\boldsymbol{B}_i = \{\boldsymbol{B}_{ij}: j=1,\ldots,N_i\}$ is mapped to an image attribute vector $\boldsymbol{A}_i$ according to its minimum distances to the codewords, that is:

$$\boldsymbol{A}_i = \begin{bmatrix} a_{1,i} \\ a_{2,i} \\ \vdots \\ a_{M,i} \end{bmatrix} = \begin{bmatrix} \min_{j=1,\cdots,N_i} d(\boldsymbol{E}_1, \boldsymbol{B}_{ij}) \\ \min_{j=1,\cdots,N_i} d(\boldsymbol{E}_2, \boldsymbol{B}_{ij}) \\ \vdots \\ \min_{j=1,\cdots,N_i} d(\boldsymbol{E}_M, \boldsymbol{B}_{ij}) \end{bmatrix} \tag{17}$$

Note that in Equation (17) and also in the IDC algorithm, we compute the minimum distance over a set of points to learn and access the codebook. This is a potential source of overfitting. We considered employing a low percentile cutoff rather than the minimum distance. But that approach risks ignoring the most informative detected regions. On balance, we chose to use the minimum distance approach, but then to introduce additional methods to reduce the risk of overfitting. We apply *unsupervised* codebook initialization followed by *local* generalization of codewords to avoid making overly strong use of the class labels. And we apply Bagging (Breiman 1996) to avoid overfitting in the image classifier. Our excellent test set performance shows no overfitting.

4.4.3 Bagged Decision Lists Classifier

Based on the learned codebook, we have chosen bagged decision lists as our classifier. This classsifier combines and adapts the boosting feature selection method from Opelt et al. (2006) and the idea of cascaded classifiers framework (Viola and Jones 2001). The decision list classifier is better suited to this problem than other standard classifiers for two reasons. First, decision lists favor situations in which the conjunctions of a small number of features are capable of discriminating the two classes. This matches our experience and the results of the previous work presented in Section 2.4. Second, decision lists are flexible classifiers — they can adjust their complexity as necessary to fit the data.

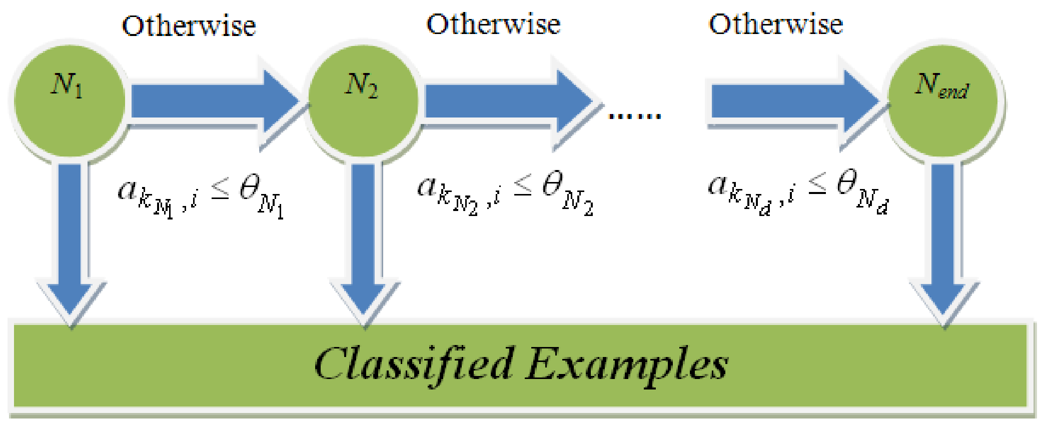

Otherwise
Otherwise
Otherwise
N_1
N_2
......
N_{end}
$a_{k_{N_1},i} \leq \theta_{N_1}$
$a_{k_{N_2},i} \leq \theta_{N_2}$
$a_{k_{N_d},i} \leq \theta_{N_d}$
Classified Examples

Figure 28. Illustration of the decision list classifier.

Decision list

A decision list is a variable-length sequence of decision nodes. As shown in Figure 28. Each node N is defined by an image attribute index k_N, a classification threshold θ_N, and a class label C_N for prediction. An image i is classified by node N into class C_N if

$$a_{k_N,i} \leq \theta_N; \tag{18}$$

where $a_{k_N,i}$ is the value of the image attribute vector $\boldsymbol{A}_i$ at dimension k_N. An example is classified by processing it against each node in the decision list until one of the nodes is able to classify the example.

Training

Given all the attribute vectors for training images, the decision list is grown by starting with the empty list and adding decision nodes one at a time until all these training examples are correctly classified. The detailed steps of the algorithm are as follows:

1. Find the best decision node: The algorithm calls the function "*NodeFinder*" to search for the image feature dimension and corresponding threshold and class label that has the highest overall performance. The pseudo-code for the *NodeFinder* algorithm is given in Figure 29. The *NodeFinder* function is similar to the "*Weak_Hypotheses_Finder*" algorithm Opelt et al. (2006) shown in Figure 10, but it differs in several ways:

First, instead of searching for the feature dimension k* and corresponding threshold θ* that maximize the classification accuracy, our algorithm searches for the k* and θ* that maximize the "coverage", that is, the number of training examples covered by this node under the constraint that all of them are correctly classified. Second, in order to improve the robustness of our algorithm to overfitting, a safety margin (σ=3 in all experiments) is employed when computing the threshold θ*. This heuristic ensures that a node can only make predictions when we have high confidence.

2. Split the current training set based on the selected node: the correctly classified examples are removed from the training set; all the unclassified or misclassified examples are passed to the next node.

3. Repeat steps 1 & 2, adding new decision nodes to the list until the training set is empty, i.e., all the training examples have been correctly classified.

Classification

Applying the learned decision list to a new image is similar to the training process. Its region descriptor vectors are first mapped to the image attribute vector according to Equation (17). The resulting attribute vector is passed down the decision list until the image is classified by a decision node. Classification is efficient because most of the examples are classified by the nodes that appear early in the decision list.

Inputs: Training image attribute vectors: $\boldsymbol{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{M,1} & a_{M,2} & \cdots & a_{M,m} \end{bmatrix}$,

the class labels of the training images: $\boldsymbol{L} = (l_1, l_2, \ldots, l_m)$.

(M is the number of image attributes, i.e. the size of the final codebook; and m is the number of training images).

Sorting: For each dimension (row) k of training attribute vectors $\boldsymbol{A}$, sort the m images in non-decreasing order of the attribute values, that is, find a permutation $\pi_k(1)$, $\pi_k(2)$, …., $\pi_k(m)$, such that:

$$a_{k,\pi_k(1)} \le a_{k,\pi_k(2)} \le \cdots \le a_{k,\pi_k(m)} .$$

Search for the maximal-coverage dimension:

For each image attribute k, compute the maximal coverage: $MC(k) = \max_p |\sum_{j=1}^{p} l_{\pi_k(j)}|$, under the constraint that the classification precision is 100%, i.e.,

$$l_{\pi_k(1)} == l_{\pi_k(2)} == \cdots == l_{\pi_k(p)} .$$

$$k^* = \arg\max_k MC(k) . \quad \text{// the maximal-coverage dimension}$$

If k^* classifies positive examples, $C^* = +1$; else, $C^* = -1$.

Compute classification threshold:

$$\theta^* = (a_{k^*,\pi_{k^*}(s-\sigma)} + a_{k^*,\pi_{k^*}(s+1-\sigma)})/2$$

with s be the position where k^* achieves highest MC value.

Output: Decision node: [k^*, θ^*, C^*].

Figure 29. Pseudo-code for the NodeFinder algorithm

Bagged decision lists (BDL)

Note that because the decision lists can grow to be arbitrarily long, they can have high variance, which can lead to overfitting and poor performance. So we perform 200-fold bagging. This is accomplished by drawing 200 bootstrap replicates (Breiman 1996) of the training images and learning a separate decision list for each replicate training data set. A new image is classified by each of the 200 decision lists, which then vote to determine the overall prediction.

Computational complexity

Learning a decision list has complexity $O(Km\log m)$, where K is the size of the codebook and m is the number of training examples. Since both K and m are on the order of several hundred, learning a decision list classifier is not expensive. In our experiments, the MATLAB implementation of the decision list learning algorithm takes several minutes to learn a decision list. Applying the learned codebook and classifier to a new image typically requires less than one minute.

4.4.4 Experimental Results

Our generative/discriminative codebook learning method is tested on three families of object recognition problems: the Caltech dataset (*Caltech*), the UIUC cars side dataset (*UIUCCar*), and the GRAZ-01 (*GRAZ*) dataset. All of the problems are binary *object present* versus *object absent* decision problems. Our approach could be easily applied to multi-class recognition problems by using divide-and-conquer techniques, e.g., the "One *vs.* the Rest" strategy or the Error Correcting Output Coding method in Dietterich and Bakiri (1995). Three interest region detectors are employed to extract the regions from object images: the Hessian-Affine detector (Mikolajczyk and Schmid 2004), Kadir's salient region detector (Kadir et al. 2004), and the PCBR detector (see Chapter 2 and Chapter 3). Each region is re-represented by steerable filters (Freeman and Adelson 1991) introduced in Section 2.2.2.

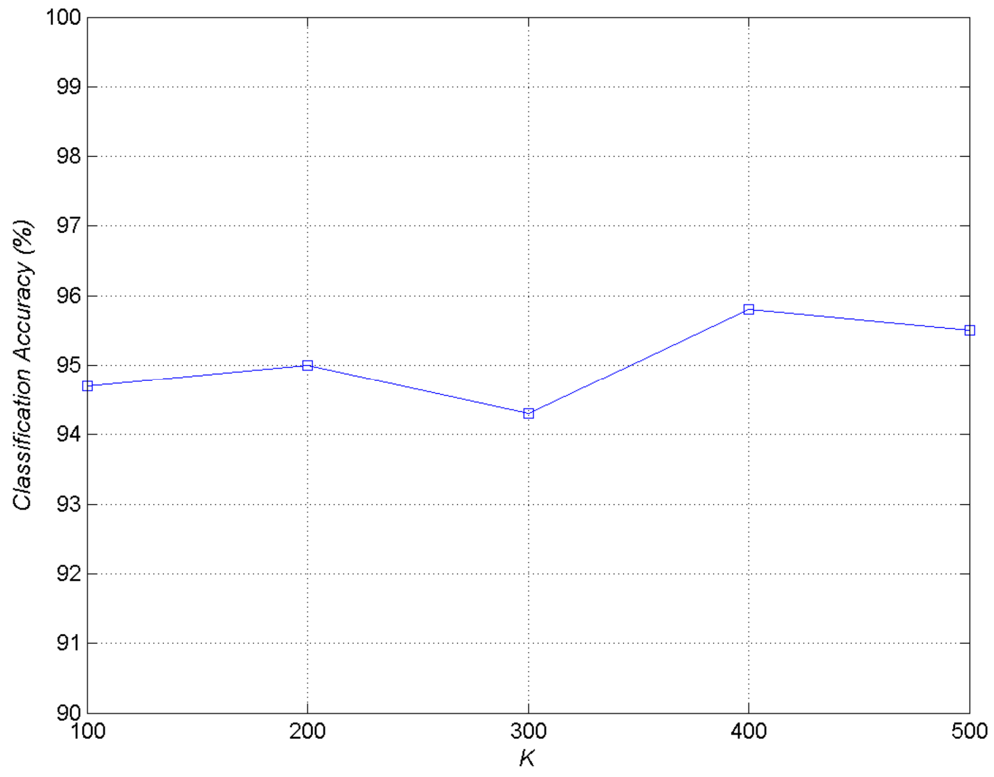


Figure 30. Recognition accuracy of IDC-BDL on Caltech cars vs. empty road scenes problem (Caltech) for different codebook sizes *K*.

On the test datasets, our method is compared to state-of-the-art generic object recognition approaches based on discriminative image feature selection (Chen et al. 2006; Dorko and Schmid 2004; Opelt et al. 2006), as introduced in Section 2.4; and to the constellation model in Fergus et al. (2007).

Before presenting the results, we first report the results of a sensitivity study for the most important parameter *K*, the size of the initial codebook. The experiment was performed on the Caltech cars (rear) *versus* empty road scenes problem. The results are shown in Figure 30. The relatively flat curve demonstrates that the performance of the learned codebook is very robust to the setting of K. This is probably due to the automatic codebook compression step in the IDC algorithm. Consequently, in all our experiments, we set $K = 200$. Similar robustness was observed for other parameters in our method. Thus, we use a single set of parameter values for all experiments.

Experiment settings are the same as in previous papers for fair comparison. The results are reported as the ROC-equal-error rates (i.e. $P(\text{True positive}) = 1 - P(\text{False positive})$), where the ROC curves are obtained by varying the number of votes required by the 200 decision lists to classify an example as positive. All the experiments are repeated with five random splits, and the average results are reported. The results are summarized in Table 8 – Table 10. We can see that our method, IDC-BDL (Iterative Discriminative Clustering + Bagged Decision Lists), gives superior performance on most of the problems. On all problems where we obtain improvements, the differences are statistically significant at a 95% level using an unpaired test for the difference between two proportions (Dietterich 1998). Even on object classes where our method is not the best, its performance is comparable to other methods. In summary, the overall recognition performance of our approach matches or exceeds the state of the art.

In order to analyze the contribution of the two major parts of our system, we also compare the whole system IDC-BDL to ablated versions on the Caltech problems. The two ablated configurations are (*i*) KM-BDL (the initial codebook obtained from unsupervised *k*-means clustering + Bagged Decision Lists classifier) and (*ii*) IDC-DL (IDC codebook learning + a single decision list classifier). The KM-BDL version is compared to test the value of discriminative learning of visual codebook. The IDC-DL version is compared to test the value of bagging ensembles for the learning of the image classifier based on the image attribute vectors. The results are summarized in Table 11.

Table 8. EERs of IDC-BDL approach and other approaches on Caltech dataset

Dataset	*IDC- BDL*	*Fergus*	*Opelt*	*Chen*
Airplanes	**99.2**	93.7	88.9	98.0
Faces	98.4	91.7	93.5	**99.5**
Motorbikes	**98.3**	96.7	92.2	96.7
Leopards	**98.0**	89.0	/	/
Cars (Rear)	**95.5**	91.2	91.1	94.5

Table 9. EERs of IDC-BDL approach and other approaches on UIUC cars side dataset

Dataset	*Average length [confidence interval]*	*IDC-BDL*	*Fergus*	*Opelt*
Cars (side)	27.3 [23.8, 31.5]	**92.7**	88.5	83.0

Table 10. EERs of IDC-BDL approach and other approach on GRAZ dataset

Dataset	*Average length [confidence interval]*	*IDC-BDL*	*Opelt*
Bikes	14.1 [12.3, 16.2]	**76.5**	73.5
Persons	16.2 [14.1, 18.7]	**71.7**	63.0

Table 11. Ablation experiment for IDC-BDL approach on Caltech dataset

Dataset	*Average length [confidence interval]*	*IDC-BDL*	*KM-BDL*	*IDC-DL*
Airp.	14.4 [12.4, 16.7]	**99.2**	97.5	92.4
Face	8.3 [7.1, 9.7]	98.4	**99.3**	87.9
Motor.	14.8 [12.8, 17.3]	**98.3**	95.8	89.2
Leopard	3.8 [3.3, 4.4]	**98.0**	96.2	94.4
Cars (R)	20.4 [17.7, 23.7]	**95.5**	93.0	91.0

IDC-BDL outperforms KM-BDL and IDC-DL on most of the object classes; and on all these object classes, the improvements are statistically significant at the 95% level. This shows that supervised codebook learning is valuable for the construction of high-performance problem-specific visual codebooks and that bagging helps a lot when the individual classifier cannot generalize well to new testing images (especially on Faces and Motorbikes problems). In addition, we also report statistics (the mean and a 95% confidence interval) on the length of the decision lists learned by IDC-BDL. The decision lists are usually fairly short. This shows that only a small number of most discriminative codewords are sufficient for accurate classification.

4.5 Learning Non-Redundant Visual Codebooks

4.5.1 Motivation

All the codebook learning methods introduced in previous sections build one codebook by performing a single clustering on the low-level features. But in real-world applications, data can be represented in many different ways; and oftentimes a single codebook is not enough to fully describe the different structures of the data. In this dissertation, we introduce a simple framework for learning multiple codebooks that are non-redundant in discriminative power (Zhang et al. 2009). This framework is introduced in Section 4.5.2. This framework seeks to advance the state-of-the-art of codebook learning in an orthogonal direction in the sense that any new advanced codebook learning algorithm can be employed in this framework to further improve its performance. The basic idea is to wrap the codebook construction process inside a boosting procedure. Each iteration of boosting begins by learning a codebook according to the weights assigned by the previous boosting iteration. The resulting codebook is then applied to encode the training examples; a new classifier is learned; and new weights are computed.

We apply the new non-redundant codebook learning framework on both visual object recognition and document classification tasks. This dissertation will only describe the application on object recognition (see Section 4.5.3). The resulting methods give very substantial performance gains over the baseline of learning a single codebook of equivalent size. For the difficult stonefly recognition problem (see Chapter 3), we obtain a 77% reduction in error on the challenging STONEFLY9 task.

This non-redundant codebook learning framework is inspired by the successful application of multi-view clustering algorithms (Cui et al. 2007; Jain et al. 2008) for exploratory data analysis. Non-redundant clustering is motivated by the fact that real-world data are complex and may contain more than one interesting form of structure. The goal of non-redundant clustering is to extract multiple structures from data that are different from one another; each is then presented to the user as a possible view of the underlying structure of the data. In principle, the non-redundant clustering techniques developed by Cui et al. (2007) and Jain et al. (2008) can be directly applied to learn multiple clusterings of the low-level features and create codebooks that are non-redundant with each other. However, such an approach is fundamentally unsupervised and will not necessarily lead to more accurate classifiers. In contrast, our framework learns codebooks that are non-redundant in the sense that they complement each other in their discriminative power.

There are a few recent developments in the vision community that are related to this framework, as presented in Section 2.3.3. Moosmann et al. (2007) learn multiple, independent randomized decision trees to partition the low-level feature space. However, they do not produce codebooks of complementary discriminative power. Yang et al. (2008) proposed a codebook learning framework that is integrated with classifier learning. Our framework shares the same basic boosting principle of sequential learning of codebooks and classifiers; yet our framework is much more general and can be applied with any form of visual codebook and classifier.

4.5.2 General Framework of Non-redundant Codebook Learning

The overall framework of our non-redundant codebook learning method is illustrated in Figure 31. Given a base codebook learner and a classifier, we iteratively learn one codebook at a time and stop when reaching the pre-defined *T* iterations. Each iteration consists of following steps:

1. Codebook learning: The inputs to the base codebook learner at iteration *t* are the training examples $\{B_i\}_{i=1}^m$ (where $B_i = \{\boldsymbol{x}_1^i, \ldots, \boldsymbol{x}_{N_i}^i\}$ is the bag of image features for training example *i*) and a set of weights $W^t = \{w_i^t\}_{i=1}^m$ specifying the importance of each example. The output is a codebook $D_t = (d_1^t, \ldots, d_K^t)$ where d_k^t is the *k*-th code word (cluster). In the first iteration, the weights are initialized to be uniform over the training examples.

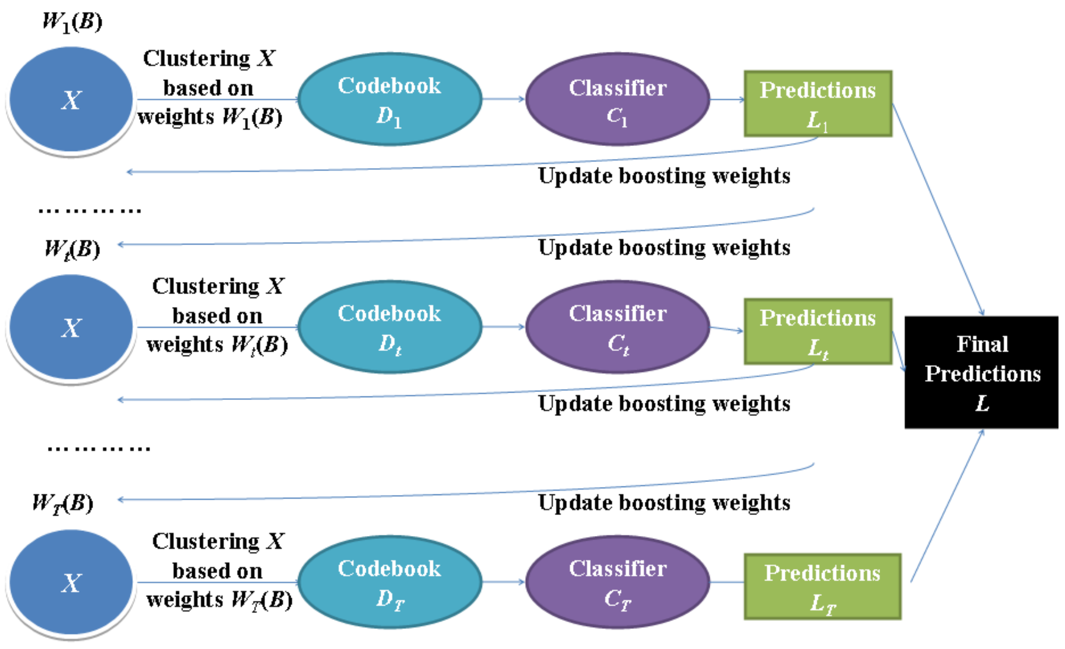


Figure 31. Illustration of non-redundant visual codebooks and classifiers learning framework.

2. Classifier learning: the training examples are mapped to fixed-length attribute vectors $A_t = \{a_i^t\}_{i=1}^m$ based on the codebook D_t such that the *j*-th attribute, $a_i^t(j)$, is related to the number of features in B_i mapped to code word d_j^t: $|\{x_n \in B_i : x_n \mapsto d_j^t\}|$. A classifier C_t is then learned from the attribute vectors A_t. The output of C_t are the class label predictions L_t.
3. Weight updating: the predictions L_t are used to update the training example weights as in AdaBoost (Freund and Schapire 1996) − the weights of the incorrectly-classified examples are increased and the weights of the correctly-classified examples are decreased. The updated weights, W^{t+1}, are provided to the base codebook learner for the next iteration.

To classify a new example B, it is mapped into T fixed-length attribute vectors $\{a^t\}_{t=1}^T$, and each a^t is then classified by C_t. The outputs of the T classifiers are combined to give the final class label prediction L via the weighted voting scheme of AdaBoost.

The above framework can be applied with any base codebook learner and any classifier. Some codebook learners can be easily modified to take the weights of the examples into consideration. Incorporating the weights into others may not be straightforward. For the latter case, as we show in Section 4.5.3, sampling can be employed to learn codebooks from weighted examples effectively. In the following sections, we will present the instantiation of our framework for the stonefly recognition application to demonstrate the high performance of our framework.

4.5.3 Boost-Resampling Algorithm and Experimental Results

We apply this non-redundant codebook learning framework to the difficult stonefly recognition problem described in Chapter 3. For this large scale data set, a practical challenge is computational efficiency. In particular, we often need to learn from thousands of images, and each image often contains hundreds or even thousands of low-level interest region descriptors in order to capture sufficient visual information for classification. The huge number of low-level features requires our non-redundant codebook learning algorithm to be highly efficient. Below we present the Boost-Resampling algorithm that applies the general framework described in Section 4.5.2 to learn a set of non-redundant visual codebooks efficiently.

Base Codebook learner

There are many options for base codebook learners for visual object categorization, including both unsupervised and supervised clustering methods, as introduced in Section 2.3. In order to reduce the risk of overfitting the training data, we choose to employ unsupervised *k*-means clustering as our base codebook learner which is also employed in Csurka et al. (2004), Sivic and Zisserman (2004) and Zhang et al. (2007). *k*-means clustering is also preferable because it is very efficient. The more complex GMM modeling algorithm (Duda et al. 2001) was also tested on the stonefly recognition task and failed to outperform the simpler *k*-means algorithm in our framework. The size of each codebook, *K*, is set to 100 empirically. According to our experiments, different values of *K* had little effect on the performance of the algorithm.

Feature mapping based on codebook

As presented in Section 2.3.4, a lot of previous work employs the histogram of occurrences method to map the low-level image features to fixed-length image attribute vectors. But this mapping method treats the codewords as equally important without considering their distribution over different examples. In this dissertation, we employ the *tf–idf* weight (*term frequency–inverse document frequency*) (Salton and Buckley 1988) developed for information retrieval and text mining. This mapping method was also employed in Sivic & Zisserman (2004). For image B_i, the j^{th} dimension of the *tf–idf* attribute vector a_i is given by:

$$f_i(j) = \left(\frac{|\{x_n \in B_i : x_n \mapsto d_j\}|}{N_i} \right) \times \log\left(\frac{m}{|\{B_l : \exists_{x \in B_l} x \mapsto d_j\}|} \right) \qquad (19)$$

As in the document domain, the first term (*tf*) measures the number of occurrences of the j^{th} codeword in the example divided by the total number of low-level features in the example. This term is exactly the histogram of occurrences measurement. The second term, *idf*, measures the (lack of) distinctiveness of the j^{th} codeword over different examples. A codeword that appears in more examples has a lower *idf* value; while a codeword that is only found in one example has the highest *idf* value. The *tf–idf* mapping method improves the robustness of learning algorithms, especially when the distribution of codewords is significantly unbalanced over different examples, which is the case in many object categorization problems. In our initial experiments, the *tf–idf* mapping

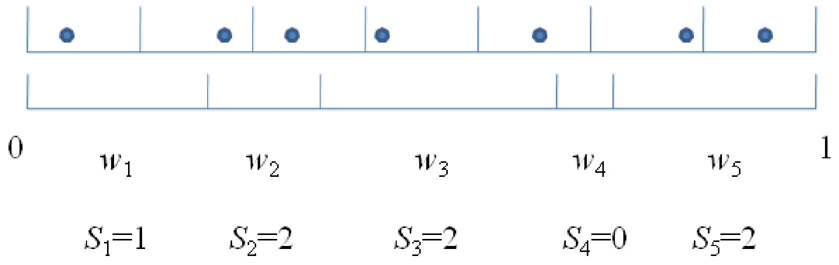


Figure. 32 Illustration of Quasi-Random Weighted Sampling (QWS) technique.

systematically outperforms the histogram of occurrences mapping. Hence, we adopt *tf–idf* as the feature mapping method.

Classifier learning

We employ an ensemble of 50 un-pruned C4.5 decision trees (Quinlan 1993) in each boosting iteration. The trees are generated via bagging (Breiman 1996). Learning more than 50 trees did not provide superior performance in our experiments.

Weighted Sampling

Note that instead of directly using the weights with *k*-means clustering, we adopt the Quasi-Random Weighted Sampling (QWS) (Kalal et al. 2008) approach to achieve improved efficiency. QWS creates a smaller "active set" of the training images based on the weights assigned to the examples such that the examples with larger weights have higher probability of being selected. As a result, the algorithm is more likely to select the training examples that have not been well-represented and classified by previous codebooks. Thus, the codebook learned on this pool is encouraged to be different to the previous codebooks.

The principle of QWS is illustrated in Figure 32. The weights are represented as intervals and arranged on the unit line segment. The line segment is split into *m* equal intervals, where *m* is the total number of training images. Within each interval, one random number (shown as a dot in the figure) is generated uniformly at random, whose position determines the index of the selected sample. S_i represents the number of occurrences of the i^{th} example in the sampled active set. As can be seen from the figure, an example with larger weight (e.g., w_3) is more likely to be sampled; and an example may be selected multiple times (e.g., S_3=2). More details of QWS method are described in Kalal et al. (2008). QWS reduces the variance of weighted random sampling and has worked well in previous work (Moosmann et al. 2007; Kalal et al. 2008).

The total number of codewords learned is $T \times K$ (the number of boosting iterations × the size of each codebook). This number grows to several thousands in our experiments. But at each iteration, we only sample 20% of the training data to form the active set. Therefore the learning of the codebooks is memory and time efficient because each clustering operation is performed on a small subset of the data. The *Boost-Resampling* algorithm can be directly applied to problems with continuous high-dimensional low-level features. Below we empirically evaluate it using the stonefly recognition dataset.

Experiment on the stonefly recognition dataset

In order to test the performance of our *Boost-Resampling* algorithm on complex object categorization problem, we evaluate it on the STONEFLY2, STONEFLY4 and STONEFLY9 stonefly recognition problems described in Chapter 3. The results have been presented in Section 3.5.2.

In order to test the value of building non-redundant codebooks and the value of weighted sampling, we compare our *Boost-Resampling* algorithm with two baseline algorithms. The first algorithm (referred as *Single*) learns only a single codebook for each detector/descriptor combination to represent the data. This codebook is built by *k*-means clustering on the pool of training image features. The size of the codebook is set to $T \times K$ (the number of boosting iterations × the size of each codebook) for fair comparison with non-redundant codebooks. The second baseline (called *Random*), replaces QWS sampling with uniform random sampling

Table 12. Classification accuracies (%) of Boost-Resampling algorithm and two baselines on STONEFLY2, STONEFLY4 and STONEFLY9 datasets.

Dataset	*Boost*	*Single*	*Random*
STONEFLY2	**97.85**	85.84	89.16
STONEFLY4	**98.21**	67.20	90.42
STONEFLY9	**95.09**	78.33	89.07

that neglects the boosting weights. This comparison experiment is performed on STONEFLY2, STONEFLY4 datasets and the complete STONEFLY9 dataset. The comparison results are summarized in Table 12. *Boost-Resampling* outperforms the two baselines on all the datasets, and the differences are statistically significant at a 95% level (Dietterich 1998). Comparing to a single codebook of equivalent size, the new method was able to achieve error reductions of 94.5%, 84.8% and 77.3% respectively.

In summary, this dissertation studied various unsupervised and supervised visual codebook learning approaches and their application to biological and generic object recognition tasks. The proposed approaches can be easily adapted to other similar recognition problems. The non-redundant codebook learning framework is the most promising and most general method that we have developed so far. It achieved high performance on the stonefly recognition and document classification problems. We will continue our research in this direction and extensively test this framework on more datasets.

5 Socially-Driven Clothes Recognition

5.1 Overview

This section presents my study on a socially-driven clothes recognition problem that arose as part of the Palo Alto Research Center (PARC) "Responsive Mirror" project. The "Responsive Mirror (RM)" (Zhang et al. 2008b; Begole et al. 2008) is a novel system for retail fitting rooms that enables online social fashion comparisons in physical stores based on multi-camera perception. It is an implicitly controlled interface that allows a shopper to directly compare a currently worn garment with images from previously worn garments. The orientation of images from past trials is matched to the shopper's pose as he moves. The system also allows comparison to clothes that other people in the shoppers' social network are wearing. Illustration of the RM system is shown in Figure 33 and Figure 34. The architecture and the major components of the RM system are presented in Section 5.2.

A key component in the RM system is the clothes recognition and matching engine (Zhang et al. 2008a; 2008b; 2008c) for automated social fashion information retrieval. Clothes recognition is termed as a "social" vision problem (see Section 5.3.1) that requires us to appreciate and understand how humans perceive objects that have social meaning. We solve this difficult problem by identifying the most important factors for clothes recognition from a user study and then employing various low-level features and learning algorithms to recognize these factors. These approaches are introduced in Sections 5.3.2 – 5.3.9. Conclusions are given in the end along with a discussion of the results.

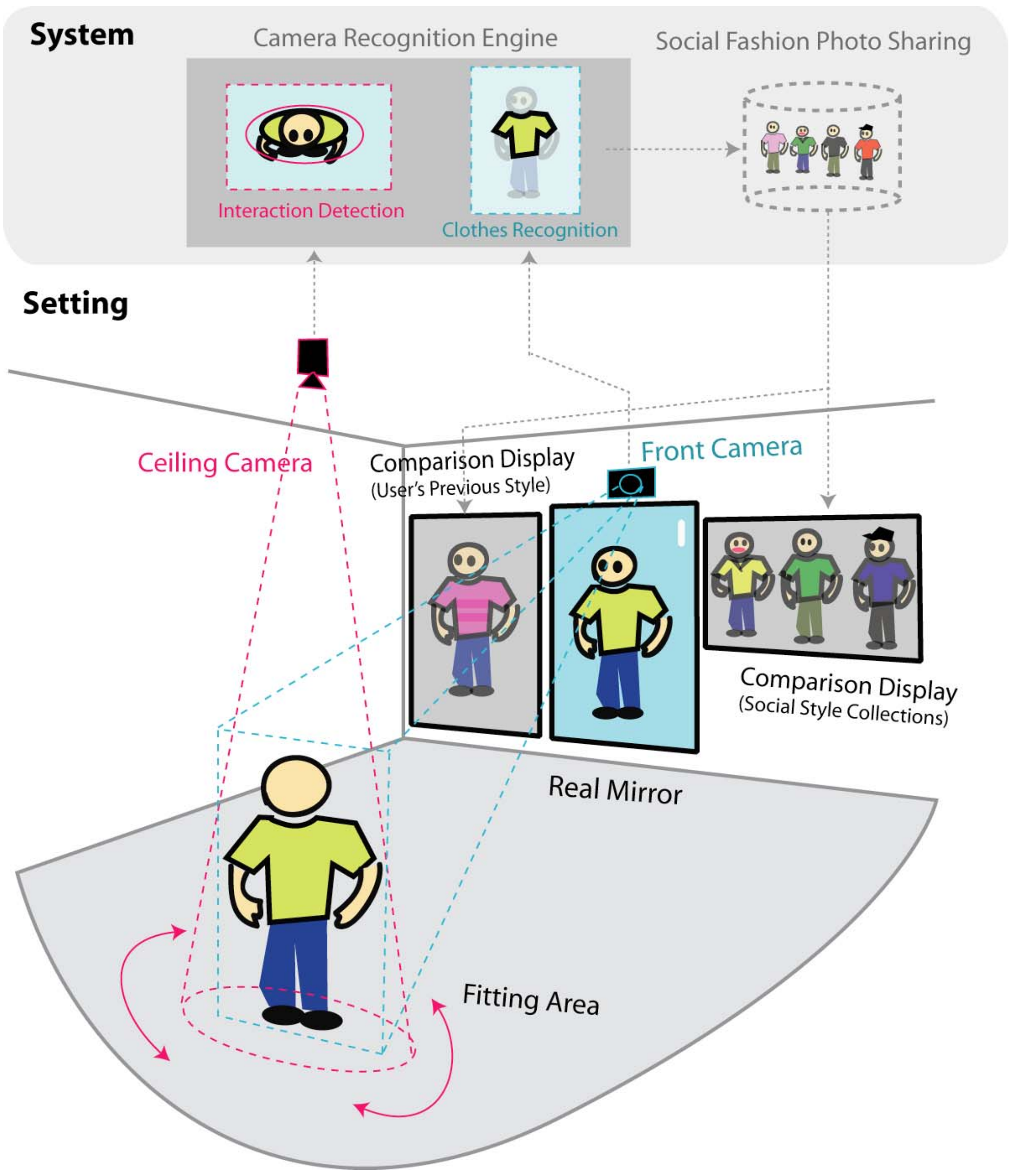


Figure 33. The concept of Responsive Mirror system.

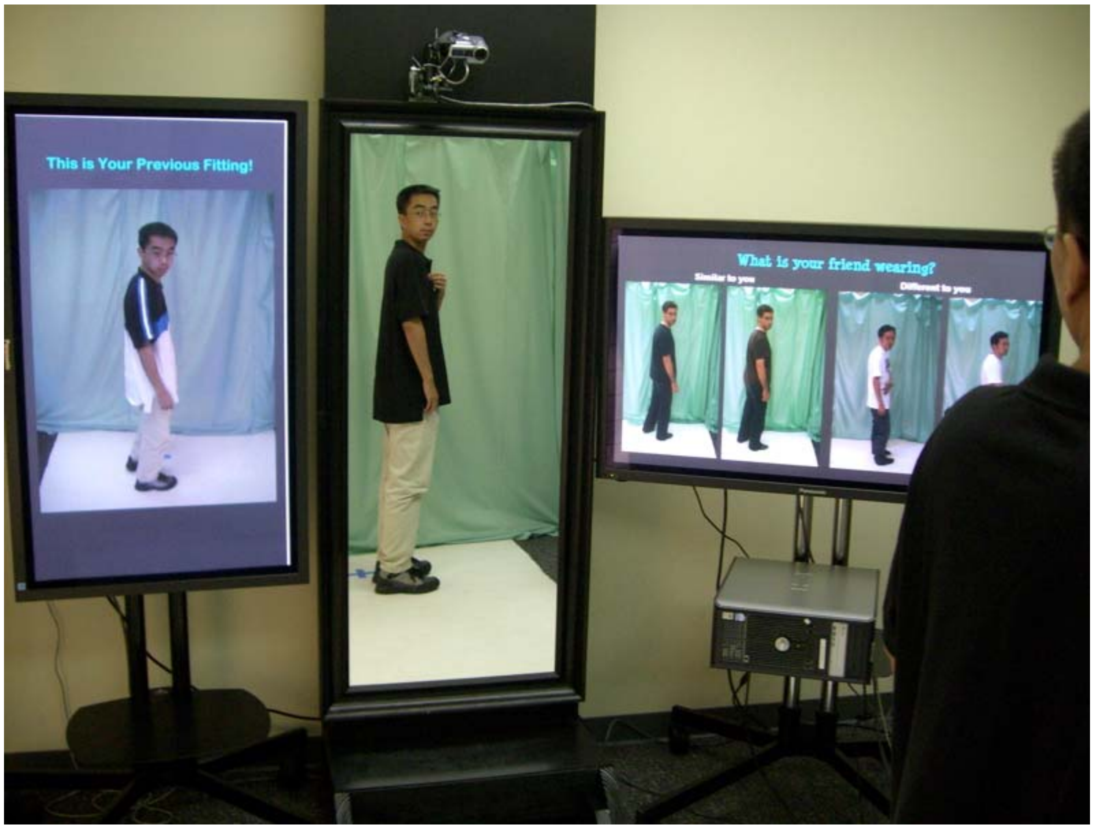


Figure 34. The Responsive Mirror prototype.

5.2 Responsive Mirror System

5.2.1 Concept and Architecture of Responsive Mirror System

Clothing shopping is an information seeking activity. Shoppers want information about availability, cost, size, colors, texture, feel, fit, style trends, and so on. Online shopping provides a great deal of this information and allows shoppers to search and compare alternative choices side by side. However, online shopping cannot provide tactile information, and shoppers still need to visit a physical store to examine certain classes of products, such as clothing and furniture. Yet, the experience of shopping in a physical retail store offers little more supplemental information than it did a decade ago. A shopper must physically seek out alternatives, and the comparisons must often be conducted sequentially, either because products are located across merchants, or because the shopper can evaluate only one item at a time, such as when trying on clothes.

To enable online shopping capability in physical stores, a system called the Responsive Mirror (Zhang et al. 2008b; Begole et al. 2008) was developed at the Palo Alto Research Center (PARC), illustrated in Figure 33 and Figure 34. As a customer interacts with a conventional mirror, frontal-view and ceiling-view cameras detect his pose as well as the style of the garment being worn. A display on the left of the mirror shows the shopper in a previously-worn garment, matching the pose of the image to the pose of the shopper as she moves. This allows the shopper to compare his current garment directly to another item he is considering. The display on the right of the mirror shows images of people wearing similar styles and different styles. These images are gathered from an online social fashion network, and they provide the shopper with information about the type of people wearing similar and different styles to the one she is considering. Users of this system do not need to be taught how to use the system – they simply behave naturally. This style of interaction is an example of "invisible computing" from the early visions of ubiquitous computing.

5.2.2 Computer Vision Components of the Responsive Mirror System

Various image features and learning algorithms are employed to provide interactive fitting display and intelligent clothes retrieval. The structure of the whole vision engine (Zhang et al. 2008b) is illustrated in Figure 35. This engine detects a person's presence in front of the mirror, detects and recognizes the clothes, and tracks the motion of the body. The following list describes the major components of the computer vision system, implemented in Visual C++ using Intel's Open Computer Vision library (*OpenCV*):

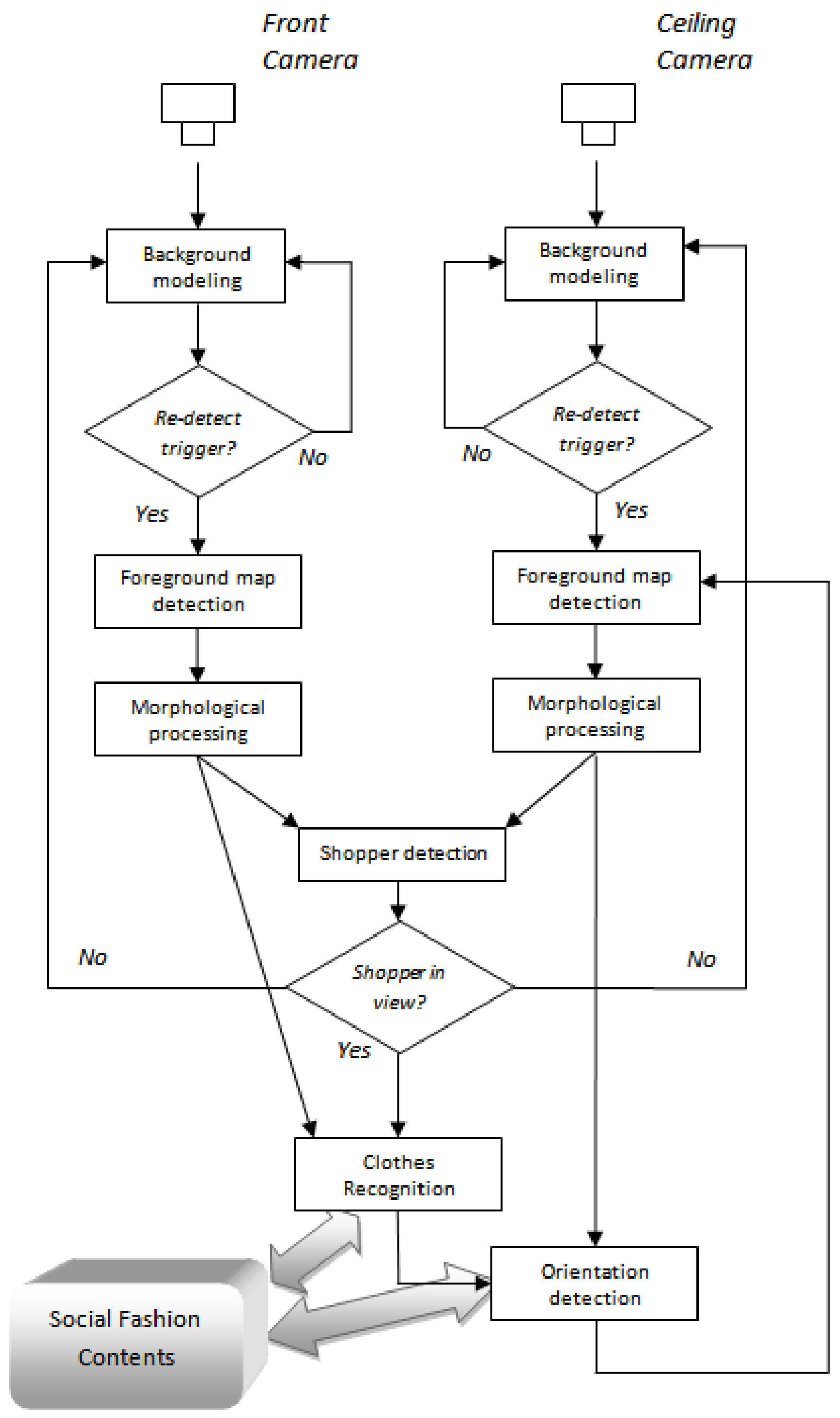


Figure 35. The computer vision components of the Responsive Mirror system.

1. Background modeling: When there is no shopper in the camera’s views, the system stays in “background modeling” state. The background models of both cameras are updated to adapt to shifting lighting.

2. Raw foreground map detection using background subtraction: After a predefined period of time, the system automatically re-detects the raw foreground map from the cameras. A pixel is identified as foreground if it is distinctive enough from the background model.

3. Morphological image processing: Opening, closing and connected component analysis are then performed on the raw foreground map to remove false detections and fill holes.

4. Shopper detection: The shopper is “in view” if she can be detected by both cameras. Then the system goes into clothes recognition and orientation detection. Otherwise, the system stays in “background modeling”.

5. Clothes detection and recognition (Zhang et al. 2008a; 2008b; 2008c): The clothes worn by the shopper are located using the bounding box of the shopper. Then clothes are classified into different categories based on different features (e.g., collar, sleeve, etc.). This component will be described in detail in Section 5.3.

6. Body orientation detection: When the system detects that a user has changed orientation to the mirror, the front camera records the image of the person in that orientation and notifies the display to show the correspondingly oriented images. The body orientation is detected by tracking the motion of the user from the ceiling camera using a particle filter.

7. Pose matching: The shoppers tend to look at themselves in different poses, so it is important to estimate not only the orientation, but also the pose of a person. Two poses are matched based on the measurement of the difference in the processed grayscale images.

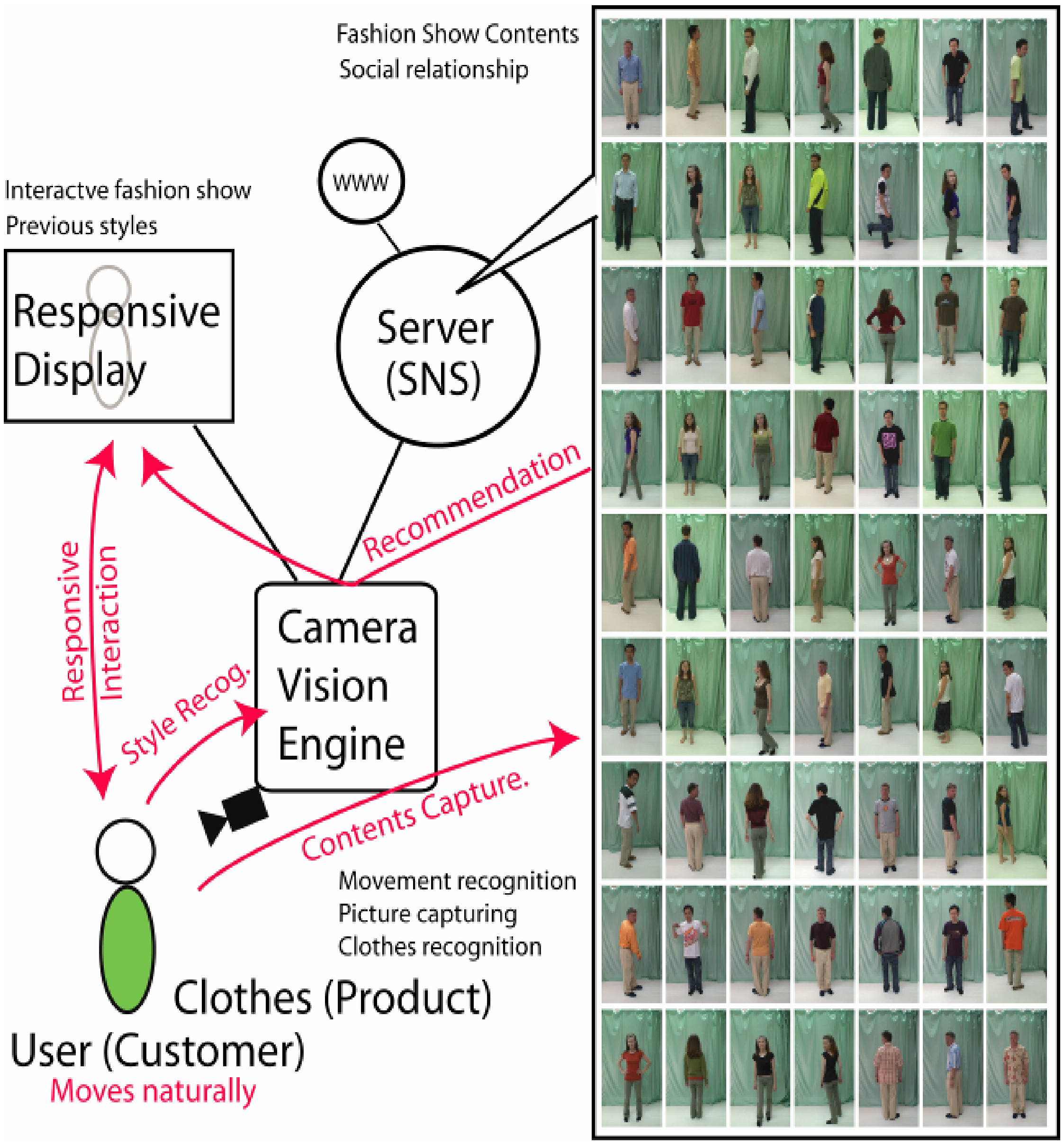


Figure 36. The concept of social fashion comparison module in the Responsive Mirror system.

5.2.3 Clothes Recognition and Retrieval

As introduced in Section 5.2, the Responsive Mirror has a “social” comparison module that displays images of others wearing outfits of both similar and different styles. The shopper is meant to use the display to help determine whether the style she is trying is close to a presentation of self that she would want to project. The concept of this idea is illustrated in Figure 36.

The key vision component in this module is the automated clothes detection, recognition and matching engine. This is defined as a “social” vision problem to address the social and semantic aspects of this challenging problem. We explore the use of various image features and learning algorithms to recognize the classes and attributes of clothing, specifically shirts, from the frontal-view camera. These approaches will be presented in Section 5.3.

5.3 Socially-Driven Clothes Recognition

Clothes choice is an important way in which people communicate their individuality, identity, tastes, status, age, wealth, and so on. Computer vision techniques have been used to automatically recognize and match the clothes in personal photos. In previous work, clothing is mainly used as contextual information for human identity recognition. Song and Leung (2006) use the clothes as additional contextual information to provide rich cues for recognizing people in pictures. The clothes location is first estimated by running face detection and taking some parts below the head; then refined by segmentation and skin-area removal. The detected clothes regions are represented by the clusters (codewords) (see Section 2.3) of “eigen-patches”, which are obtained by applying principle component analysis (PCA) (Duda et al. 2001) on the densely sampled image patches. The similarity between two pieces of clothing is measured by the normalized scalar product of their weighted histogram of occurrence vectors. Song and Leung (2006) also proposed a skin detection algorithm by matching the color tone of the face skin to the query region. The eigen-patches representations and the skin detection algorithm are employed in our clothes recognition system presented in this section.

Anguelov et al. (2007) also employs clothing appearance as one of the additional contextual cues to improve the performance of identity recognition in personal photo albums. The clothes are detected and segmented using the same method as in Song and Leung (2006). Then the similarity between two pieces of clothing is measured by the Earth Mover's Distance (EMD) between their signatures. This is very similar to the visual codebook approach in Zhang et al. (2007), described in Section 2.3.

The clothes recognition system presented in this dissertation also analyzes the color and texture information in clothes. But instead of using clothes as contextual information for human identification, the purpose of the system is to recognize the clothes themselves and match them to images of other clothes. As described in Chapter 1, clothes recognition is difficult in a number of ways: (1) the social nature of the problem definition. We approach this as an instance of "*social vision problems*". (2) The real-time requirement of the algorithm. (3) The complexity of the vision problems involved in clothes recognition, for example, the high intra-class variation and deformable configurations of the clothes. To tackle these challenges, we conducted a user study to discover the most salient clothes factors which people use to determine similarity between clothes, more specifically, shirts (see Section 5.3.1). Then we pursue a divide-and-conquer approach to shirt recognition. A factor classifier is developed to recognize each salient factor in the shirt images using image features (see Sections 5.3.2 – 5.3.8). Then the factor features are fit into regression models to measure the pair-wise shirt similarities (see Section 5.3.9). The evaluation of the algorithms is given along with the findings which are valuable for future research.

5.3.1 Social Vision Problem

The clothes recognition problem described in previous section appears at first glance to resemble a typical computer vision problem. Several interesting problems arise, however, in the course of implementing such a system, an example of a class of problems which we term *social vision problems* (Zhang et al. 2008c). As opposed to traditional problems in computer vision, social vision problems require us to appreciate and understand how humans perceive objects that have social meaning.

The following discussion highlights the underlying problems in the proposed application. First, given an image of a shirt, what features need to be recognized? A computer vision algorithm may be able to accurately detect the distance between a shirt's buttons, but do people care about this feature when looking at shirts? Thus, we must ensure that our system recognize the features that people care about. We refer to this as the *socially-driven feature selection problem*.

Secondly, once the relevant features are known, how well can the vision system detect those features in a piece of clothing? This falls in the traditional computer vision space, but clothing provides some unique challenges due to its large intra-class diversity. This is in addition to the fact that both human bodies and clothes are deformable and the algorithm must be invariant to these deformations.

Finally, once we have examined the relevant features of a piece of clothing, how do we determine which of our stored images are similar? Is a blue dress-shirt more similar to a green dress-shirt or to a blue t-shirt? We refer to this as the *socially-driven feature comparison problem*.

Social vision problems, like the one described, are computer vision problems that are inherently social and which cannot be solved by computer algorithms alone. Creating an application that recommends similar clothing requires us to also understand how people determine whether two pieces of clothing are similar or not.

In this dissertation, we describe a process for solving social vision problems as well as the specific instantiation of the process for the clothes recognition and retrieval system we described. First we needed to address the social aspects of the problem − identifying factors of shirts that people use to determine similarity between shirts. As mentioned before, features that may be easy for a computer vision algorithm to detect may not be relevant to people at all. Therefore, it is important to identify the relevant clothes factors instead of haphazardly selecting features to detect using vision algorithms.

To solve the socially-driven feature selection and feature comparison problems, a user study was conducted using a web-based survey. 65 users were invited to participate in the user study. The experimental dataset was created by photographing 12 people (male and female) wearing shirts from their personal wardrobe, for a total of 165 articles of clothing. There is significant variation in types, styles, colors, and patterns among these shirts. Each image is a single frontal view in full color with a resolution of 320×240. Higher resolution is likely to improve clothes recognition, but will result in higher computation cost.

From our dataset, we selected 25 men's shirts and 15 women's shirts that covered much of the variation in the two samples. In the survey, respondents were shown 40 randomly chosen pairs of men's shirts and 20 randomly chosen pairs of women's shirts. Respondents were then asked to rate the similarity of each pair of shirts on a 5-point scale, labeled from 1 (Not Similar At All) to 5 (Extremely Similar). At the end of the survey, respondents were asked in an open-ended question to list the most salient factors they used to determine similarity between pairs of shirts.

To analyze the open-ended responses, each unique factor listed in a participant's response was coded. The coded factors listed in order of decreasing frequency were: sleeve length, color, collar presence, shirt type, pattern, button presence, neckline, emblem/logo placement, and material/texture. Thus, we focused on 1) sleeve length, 2) shirt color, 3) collar presence, 4) pattern, 5) placket, and 6) emblem placement. Since we believed that shirt type could be deduced from the six listed clothes factors, we did not list shirt type as a separate factor. It is interesting to notice that color is not identified as the most salient clothes factor as we have expected. There is no significant difference between male and female ratings. Having addressed social aspects of the clothes recognition problem, we then integrated the findings from the user experiences into the computer vision algorithm.

A clothes recognition system is usually composed of the following stages (Song and Leung 2006; Anguelov et al. 2007): detection and segmentation of clothes, extraction of image features, and recognition of clothes based on the features. These stages will be described in the following sections.

5.3.2 Clothes Parts Segmentation

In order to recognize the clothes (shirts), we needed to detect the location of the different parts of the shirts first. For outfit images, the detection of shirts is equivalent to the detection of the body parts of the human. Manual labeling of each clothes part is most accurate but tedious. In Song and Leung (2006) and Anguelov et al. (2007), the torso is detected by first detecting the human face and then examining a very narrow part below the head. Although this approach is invariant to different human poses, a great deal of clothes information is discarded, and it is potentially sensitive to the way the person is wearing the clothes (e.g., zipped versus unzipped) and possible accessories (e.g., necklace, scarf).

Due to the real-time requirement of the algorithm, we prefer a detection method with light computational cost. For outfit images, usually the bounding box (the black boxes in Figure 37) of the human body is easy to obtain. For user-uploaded images, a simple two-click operation can give the bounding box. For an

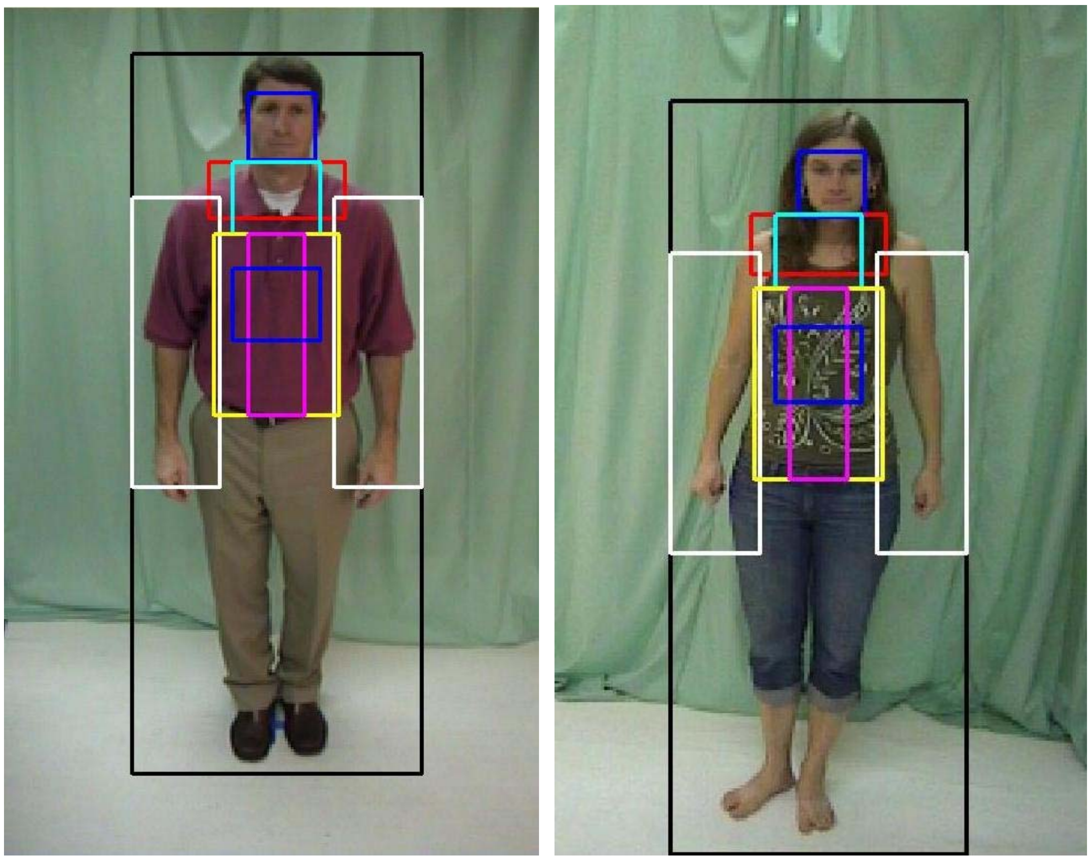


Figure 37. Examples of clothes parts detection.

outfit video, the object tracking algorithm can automatically detect the bounding box of the person. Since in the outfit images, the person is typically standing upright in front of the camera, our system detects the clothes parts by simply segmenting the bounding box with heuristic ratios, as illustrated in Figure 37.

In order to explore the salient shirt factors identified in the user study, we adopted various image features and learning algorithms (Zhang et al. 2008a; Zhang et al. 2008b; Zhang et al. 2008c) to detect and recognize these factors from a camera sensor. Considering the potential real-time requirement of the application of the algorithm, we extracted low-level features which can be computed

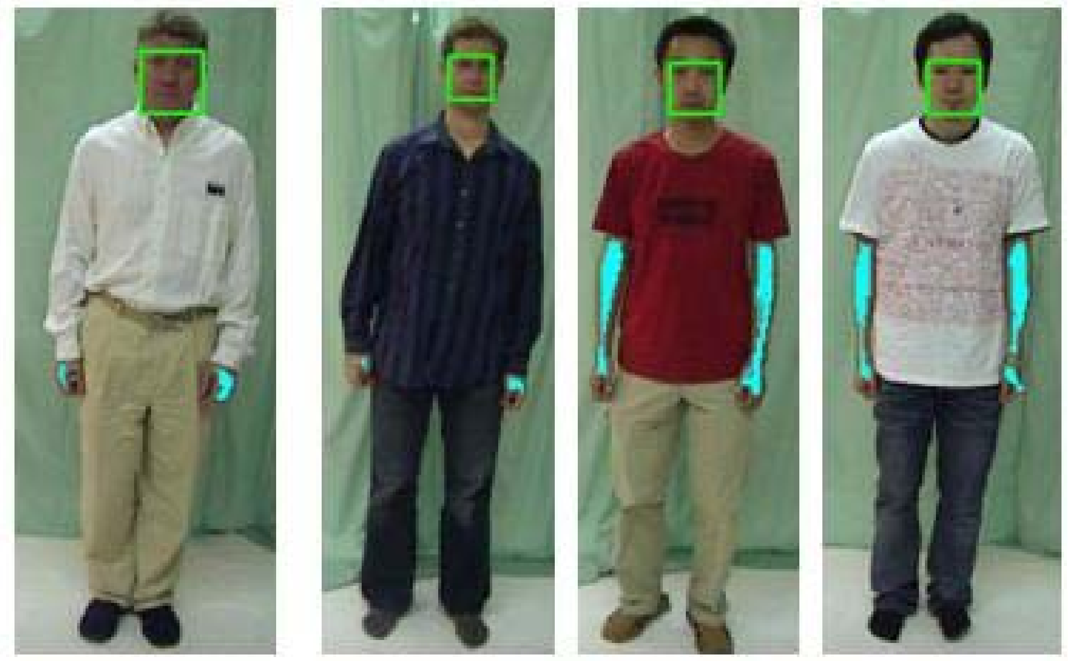


Figure 38. Examples of face detection and arm skin detection.

efficiently in the images. The factor features are then generated by simple analysis of the low-level features. After being formulated as classification problems, linear Support Vector Machines (SVMs) (Duda et al. 2001; Christopher 1998) or Decision Stumps (just for sleeve recognition) classifiers are learned on these features to recognize the factors. We also discovered the significances of the features and the relations between the factors by analyzing the weights of the decision boundaries learned by linear SVMs. The image features we employed are described in the following sections.

5.3.3 Sleeve Length Recognition

Sleeve length is the most important factor suggested by participants from the user study. It is intuitively a significant cue to discriminate between polo-shirts, casual shirts and t-shirts (class 1: short-sleeve or no-sleeve) against business work shirts (class 2: long-sleeve). In order to recognize these categories, we assumed that long-sleeve shirts have less arm skin area than short-sleeve or no-sleeve shirts. So sleeve recognition is reduced to two problems: skin detection and sleeve classification.

Generic skin detection is a difficult problem due to the difference in skin types and lighting conditions. But motivated by previous work (Song and Leung 2006), we observed that the skin tone from a person's face is usually similar to the skin tone of his/her arms. Therefore we first ran an efficient face detector (Kienzle et al. 2005) to detect the location of the person's face (the green boxes in Figure 38) from his frontal view image. Then we clustered the RGB values of the extracted facial pixels using a Gaussian Mixture Model (Duda et al. 2001). The number of clusters was set to 2 empirically, and we expected one of the clusters to represent the person's skin tone.

Then for every pixel $\boldsymbol{x}$ in the rough arm area (the white boxes in Figure 37), a small patch $\boldsymbol{p}(\boldsymbol{x})$ of size 5×5 is extracted centered at $\boldsymbol{x}$. $\boldsymbol{x}$ is identified as a skin pixel only if the following two conditions hold:

1. Patch $\boldsymbol{p}(\boldsymbol{x})$ is coherent in color: the variance of RGB values in $\boldsymbol{p}(\boldsymbol{x})$ is smaller than a threshold. This is to prevent false detections from skin-like colors in sleeves.

2. The minimum Mahalanobis distance from the mean of the RGB values within $\boldsymbol{p}(\boldsymbol{x})$ to the two face pixel clusters is smaller than threshold t_S. The skin detection results using $t_S = 3$ are shown in Figure 38 with light blue areas.

After skin detection, the sleeve length is described by the inverse of the number of skin pixels detected in the arms ($\boldsymbol{A}_1$). A decision stump (see Section 2.4 and Section 3.3.2) is learned on these features to recognize the sleeve length. 5-fold cross-validation experiments were conducted on our dataset to test the performance of this sleeve recognition algorithm. For each round, the classifier is trained on 4/5 of the examples from each class, and tested on the other 1/5; the experiment is repeated 5 times. The result is summarized in Table 13 and Figure

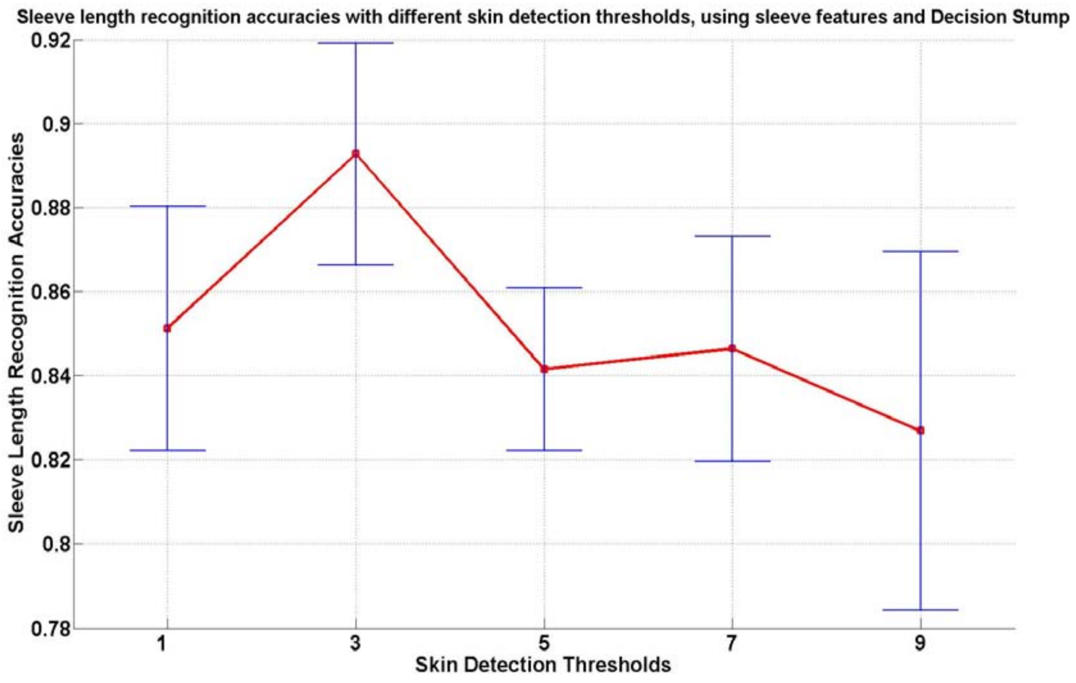
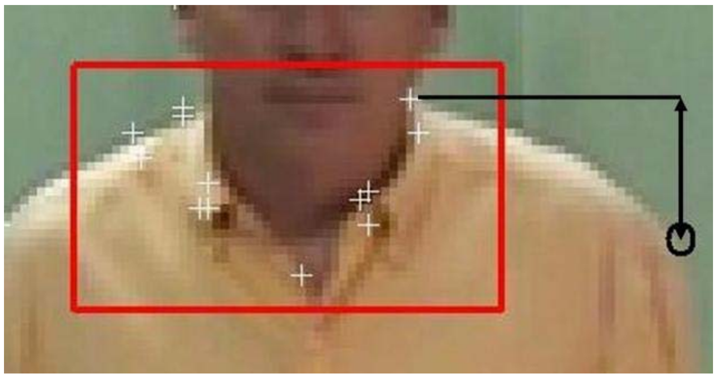


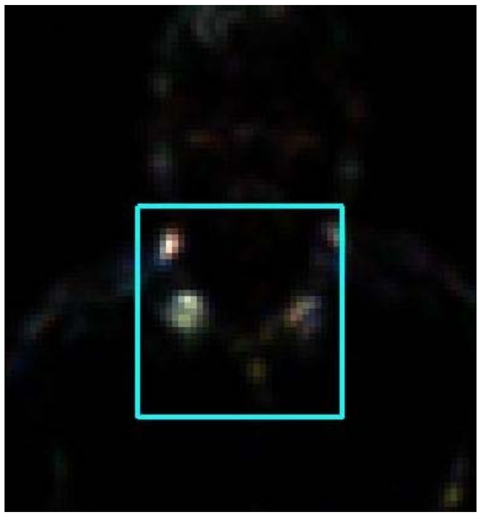
Figure 39 Sleeve length recognition accuracies (standard deviations) using sleeve feature ($\boldsymbol{A}_1$) against different values of skin detection threshold t_s.

Table 13. Sleeve length recognition accuracy [95% confidence interval].

Factor	*# of short-sleeve/ no-sleeve shirts*	*# of long-sleeve shirts*	*Recognition accuracy (%)*
Sleeve	117	48	89.2 [86.9 91.5]

39. The 10% failures were mainly due to the incorrect skin detections on the clothes when the cloth texture is coincidentally close to the skin tone.


(a)


(b)

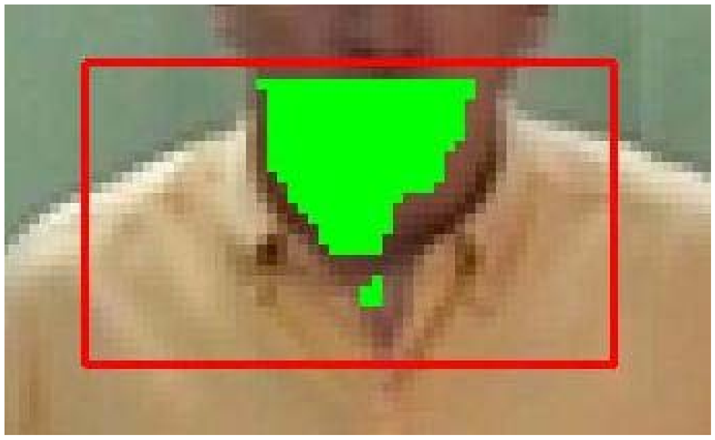

(c)

Figure 40. Illustration of image features for collar recognition

5.3.4 Collar Recognition

Participants in the study identified the presence of a collar on a shirt to be an important cue to discriminate between t-shirts and other types of shirts (e.g., business shirts and polo shirts). We explored a number of image features for collar recognition from observations of images and suggestions from users.

$\boldsymbol{B}_1$: The number of Harris corner points (Mikolajczyk and Schmid 2004; Harris and Stephens 1988) detected in the collar part (the white points within the red box in Figure 40 (a)). This is assuming that a collar shirt has more spikes and corners, which can be detected by the Harris corner detector.

$\boldsymbol{B}_2$: The variance of the y-coordinates of the Harris corner points detected in collar part. This assumes that a collar shirt usually has a corner point detected in the lower neck part, while a non-collar shirt does not.

$\boldsymbol{B}_3$: The sum of the Harris measure within the central collar region (shown in Figure 40 (b)). We observed that a collar shirt usually has stronger corners in the central neck area, but a non-collar shirt does not.

$\boldsymbol{B}_4$: The distance (shown in Figure 40 (a)) from the upper left/right corner points detected in the neck part to the shoulder points. This is assuming that a collar shirt has larger neck-to-shoulder distance than a non-collar shirt.

$\boldsymbol{B}_5$: The skin area in the central collar part (the green area within the red box in Figure 40 (c)). This assumes that a collar shirt has less skin area than a non-collar shirt. The skin area is detected using the same method as for arm skin detection.

A linear SVM with soft decision boundary (Duda et al. 2001; Christopher 1998) is learned on $\boldsymbol{B}_1 - \boldsymbol{B}_5$ to recognize the presence of collars. The 5-fold cross-validation performance of the collar recognition algorithm is summarized in Table 14 and Figure 41.

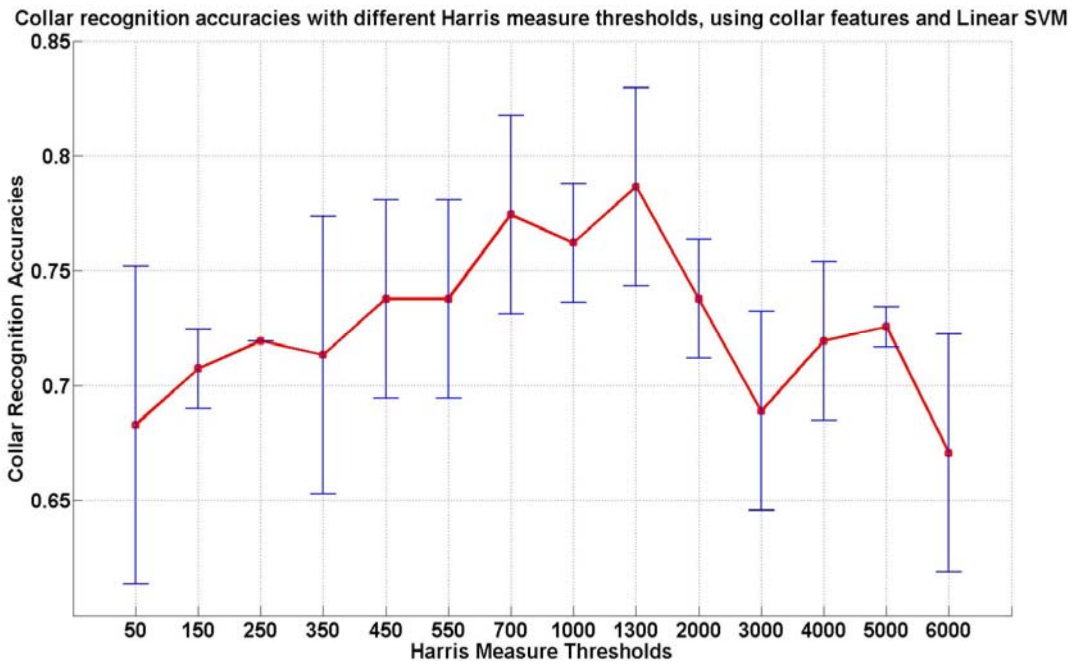


Figure 41. Collar recognition performance using collar features ($\boldsymbol{B}_1$– $\boldsymbol{B}_5$) against different Harris corner detection thresholds.

Table 14. Collar recognition accuracy [95% confidence interval].

Factor	*# of non-collar shirts*	*# of collar shirts*	*Recognition accuracy (%)*
Collar	65	100	78.7 [74.9 82.5]

From the weights of the learned linear SVM, we found that $\boldsymbol{B}_1$ is the most discriminative feature for collar recognition; while $\boldsymbol{B}_5$ is the least informative. Missed detections by the Harris corner detector are the major cause of the mis-classifications.

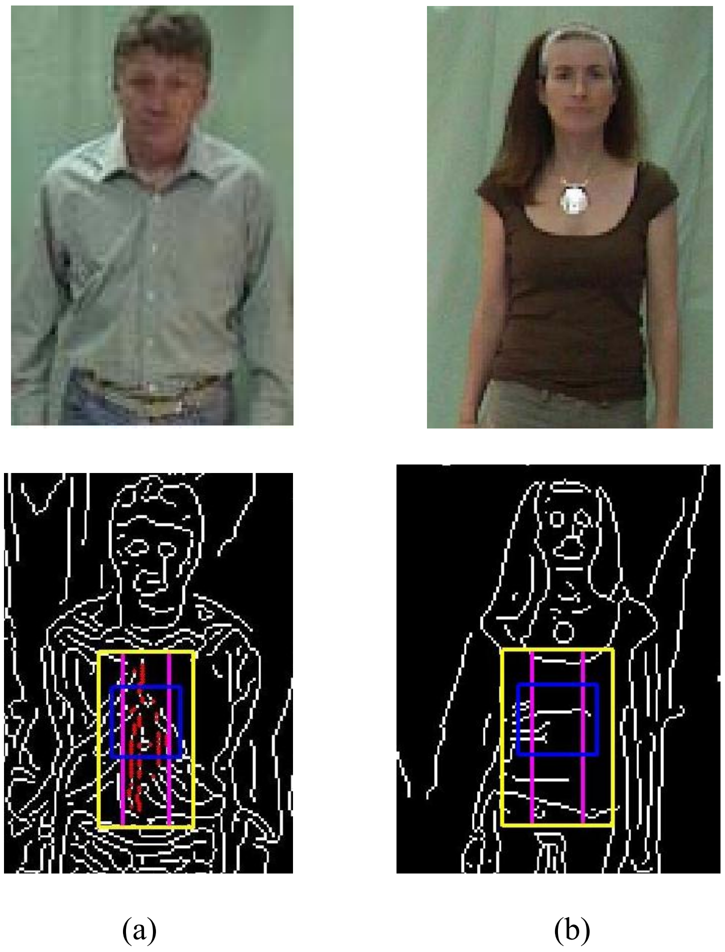


Figure 42. Illustration of image features for placket recognition on (a) a full-placket business work shirt and (b) a no-placket t-shirt.

5.3.5 Placket Recognition

The presence and amount of buttons (or zippers) in shirts was indicated to be an important cue to discriminate between t-shirts, polo shirts, and other types of shirts (e.g., business shirts). But due to the low-resolution of the clothes images and the possible high color similarity between the button and the clothing, detecting buttons is very difficult. Fortunately, the presence and length of the placket line is usually equivalent to the distribution of buttons. Thus, we employed

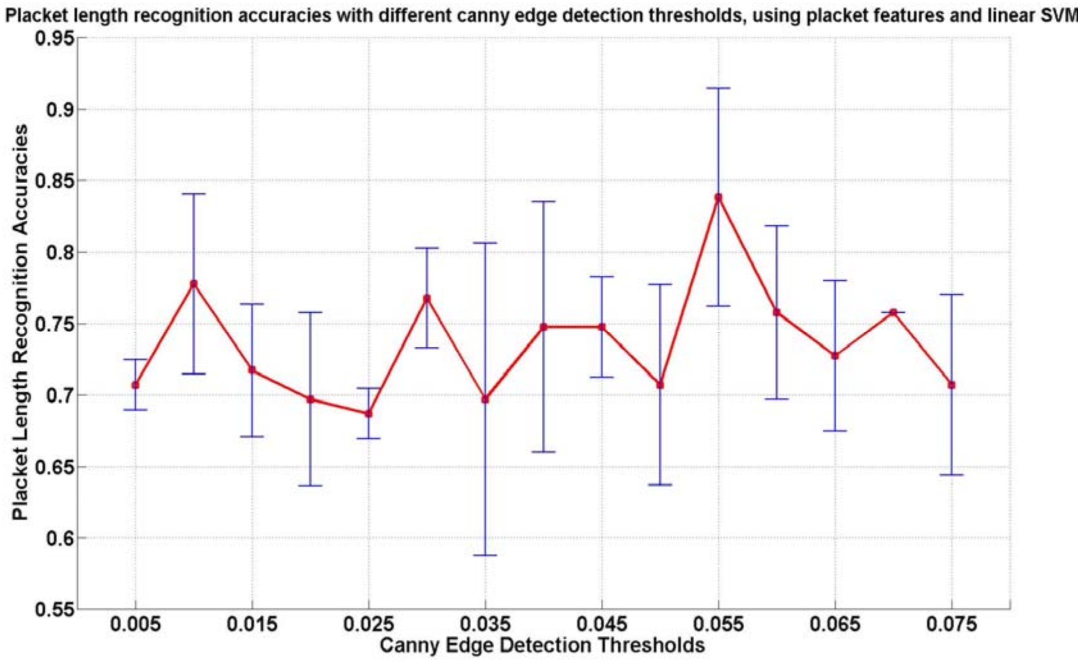


Figure 43. Placket recognition performance using placket features ($\boldsymbol{C}_1$– $\boldsymbol{C}_5$) against different Canny edge detection thresholds.

the Canny edge detector (Canny 1986) to detect the vertical placket points and measure their distribution to generate the features for placket recognition. They are described as follows:

1. The number of vertical (placket) Canny edge points (the red points in Figure 42) detected in the whole placket torso part ($\boldsymbol{C}_1$), and the numbers in the upper ($\boldsymbol{C}_2$) and lower ($\boldsymbol{C}_3$) placket torso areas (the purple boxes in Figure 37).

2. The vertical variance (distribution) of the vertical edge points in upper ($\boldsymbol{C}_4$) and lower ($\boldsymbol{C}_5$) placket torso areas.

The selection of the features is based on the following observation: the presence of large number of placket points that are sparsely distributed along vertical direction usually gives strong indication of the existence of placket line,

Table 15. Placket recognition accuracy [95% confidence interval].

Factor	*# of no/half placket shirts*	*# of full-placket shirts*	*Recognition accuracy (%)*
Placket	97	68	83.8 [77.1 90.5]

and in turn, the presence of buttons. The performance of the placket recognition algorithm is summarized in Figure 43 and Table 15. $\boldsymbol{C}_2$ is identified as the most discriminative feature for placket recognition. The incorrect predictions mainly result from the failure of the Canny edge detector to detect the vertical placket lines.

5.3.6 Pattern Complexity and Emblem Placement Recognition

The complexity of the pattern of the shirt is also indicated as valuable for clothes recognition. Intuitively, pattern complexity is related to the social semantics of the clothes. For example, a very colorful shirt is usually considered less suitable for a formal occasion than a solid shirt. Thus, we extracted the following features to recognize the pattern complexities of the shirts:

1. The number ($\boldsymbol{D}_1$) and spatial variance ($\boldsymbol{D}_2$) of the Harris corner points (Mikolajczyk and Schmid 2004; Harris and Stephens 1988) detected in the torso area (yellow box in Figure 37).

2. The number ($\boldsymbol{D}_3$) and spatial variance ($\boldsymbol{D}_4$) of the Canny edge points (Canny, 1986) detected in the torso area.

$\boldsymbol{D}_1 - \boldsymbol{D}_4$ were selected based on the observation that a patterned shirt usually has many edges or corners distributed sparsely in the torso area.

3. The complexity ($\boldsymbol{D}_5$) (measured by entropy) of the color histogram (Song and Leung 2006) extracted from the torso part. A colorful shirt will usually be considered to be a patterned shirt.

We are trying to recognize two pattern classes: (1) solid: the shirts which are plain in color and texture, no large-area patterns; (2) patterned: the shirts that are either colorful or patterned, for example, the block-patterned shirts. The performance of the pattern complexity recognition algorithm is summarized in Table 16. $\boldsymbol{D}_3$ is identified as the most discriminative feature by the SVM.

Although indicated as a less important factor for clothes recognition, detecting the emblem placement is needed for the recognition of a logo or character on the clothes. These are very valuable for clothes brand recognition and contextual information extraction. We focused on the centered vs. non-centered emblem recognition problem, because we noticed a lot of centered patterns or logos on the shirts in our dataset. The algorithm can be easily adapted to recognize other placements, for example, the pocket logo or shoulder patterns. The following features are extracted for this problem:

1. The average distance from the Harris corners ($\boldsymbol{E}_1$) and Canny edge points ($\boldsymbol{E}_2$) detected within the torso part (yellow boxes in Figure 37) to the center of the torso. If the patterns are all clustered in the center, then the emblem is usually in the center.

2. The difference in the number of the Harris corner points ($\boldsymbol{E}_3$) and Canny edge points ($\boldsymbol{E}_4$) detected within the central torso part (blue boxes in Figure 37) and the surround non-central torso parts.

3. The difference in the color complexities ($\boldsymbol{E}_5$) of the clothing patches within the central torso part and the surrounding non-central torso parts.

4. The normalized distance between the color histograms ($\boldsymbol{E}_6$) of the clothing patches within the central torso part and the surrounding non-central torso parts.

$\boldsymbol{E}_3$ – $\boldsymbol{E}_6$ are measuring the difference (in color and pattern complexity) between the central part and the surrounding clothes parts. The more distinct they

Table 16. Pattern complexity recognition accuracy [95% confidence interval].

Factor	*# of solid shirts*	*# of patterned shirts*	*Recognition accuracy (%)*
Pattern	107	58	87.9 [83.3 92.5]

Table 17. Emblem placement recognition accuracy [95% confidence interval].

Factor	*# of non-centered shirts*	*# of centered shirts*	*Recognition accuracy (%)*
Emblem	107	58	99.0 [98.0 100.0]

are, the more likely the emblem is located in the center. The linear SVM is employed as the classifier. The emblem recognition algorithm performed very well according to 5-fold cross-validation experiments, as summarized in Table 17. $\boldsymbol{E}_3$ is the most discriminative feature identified by the SVM.

5.3.7 Color Analysis

In our user study, participants identified color as one of the significant factors to measure the clothes similarity. Therefore, we employ color information as one of the factors for clothes matching. A color histogram ($\boldsymbol{H}_1$) is computed in Hue and Saturate channels from the segmented torso part (the yellow boxes in Figure 37). Then the histogram is compared with the histograms of other clothes images. The similarity between two clothes is measured in a way similar to Song and Leung (2006). Each histogram element is weighted by the total number of pixels quantized into the histogram bin. The similarity is then computed by the scalar product of the weighted histograms. Figure 44 shows an example of color-based clothes retrieval, where the clothes that are most similar and most dissimilar to the query clothes are retrieved.

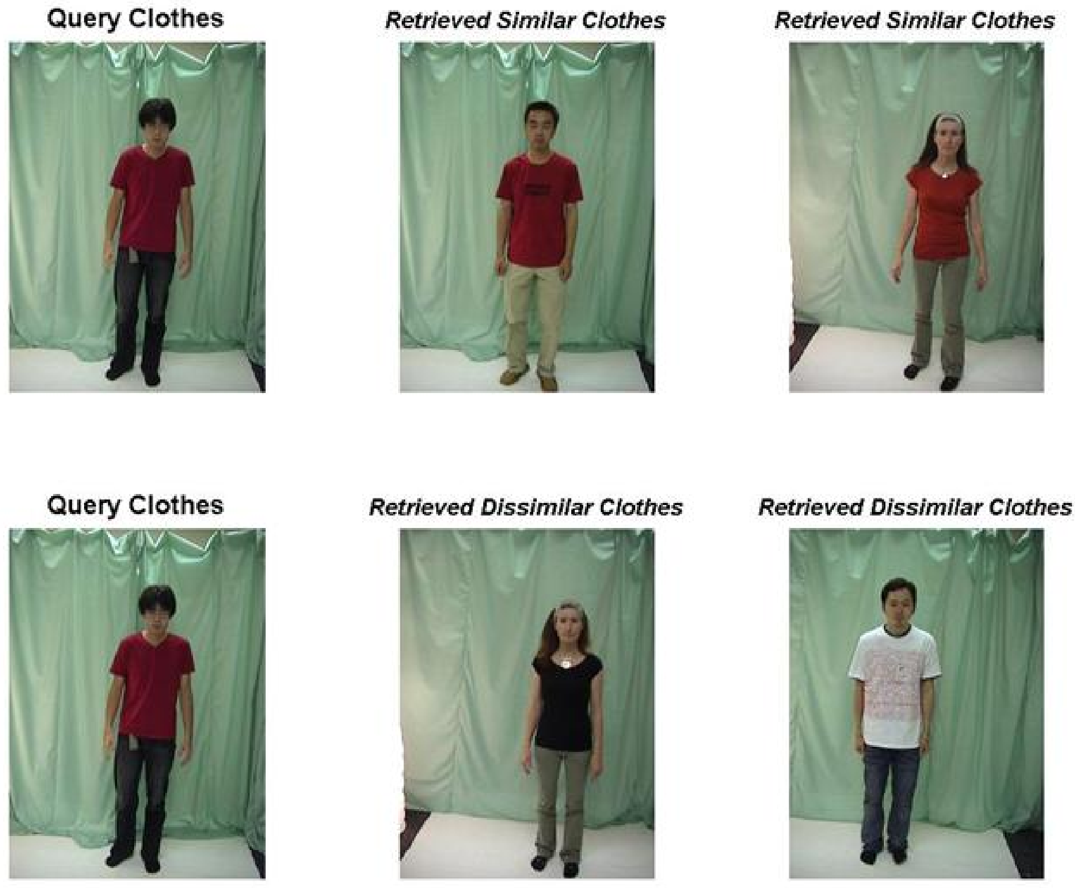

Query Clothes
Retrieved Similar Clothes
Retrieved Similar Clothes
Query Clothes
Retrieved Dissimilar Clothes
Retrieved Dissimilar Clothes

Figure 44. Example of color-based clothes image retrieval.

5.3.8 Shirt Style Recognition

We combine all the factor features described above ($\boldsymbol{A} - \boldsymbol{E}$) into a single feature vector $\boldsymbol{V}$, and apply a linear SVM to classify the shirts into different style categories. Note that the definition of shirt styles involves a lot social issues, and there is no existing clear categorization. We manually labeled the clothes images according to human experience. We defined the following shirt styles (along with the numbers of examples in the dataset):

1: T-shirt (65): no collar, no/short sleeve, no button.

2. Polo-shirt (32): has collar, short-sleeve, half-button.

3. Casual shirt (20): has collar, short-sleeve, full-button.

4. Business shirt (48): has collar, long-sleeve, full-button.

Since we have the most numbers of t-shirts and business shirts in our dataset, we first examined this binary classification problem. The result is summarized in Table 18. The confusion matrix is given along with the overall classification accuracy, which is the overall count of hits against the total number of test examples. We can see our algorithm performs very well on classifying the t-shirts against the business shirts.

We then focused on the more difficult four-class problem. The result is summarized in Table 19. Notice that the vision algorithm has significant confusion between polo and casual shirts versus business shirts. Providing more polo and casual shirts may marginally improve the performance, but we believe that the confusion mainly comes from the common features they are sharing.

The different clothes factors (sleeve length, collar, etc.) are not necessarily independent. In order to identify the true relevant image features for the clothes factors, we retrain a linear SVM on the combined feature vectors $\boldsymbol{V}$ for each of the factor classification problems, and then select the top 5 image feature dimensions that have largest weights in the construction of the learned decision boundary. These are the most relevant image features for the recognition of the corresponding factor. We then learn a separate SVM on the identified relevant features to recognize the factor. The results show that collar is the most correlated factor. Integrating the image features for pattern complexity and emblem placement recognition improves the collar recognition accuracy by 7.2%, from 78.7% to 85.9%. This is consistent with the intuition that collar shirts tend to be more formal, solid, and have fewer emblems. But for the other factors, employing the relevant subsets of image features does not help. This may because the relevant feature subsets are partially redundant or the factors are independent.

Table 18. Shirt style (t-shirt vs. business shirt) recognition accuracy.

Classified As –>	T-shirt	Business shirt
T-shirt	96.2%	3.8%
Business-shirt	5%	95%
Overall accuracy: 95.7%		

Table 19. Shirt style (four classes) recognition accuracy.

Classified As –>	T-shirt	Polo-shirt	Casual shirt	Business shirt
T-shirt	80.8%	3.9%	15.4%	0%
Polo-shirt	16.7%	41.7%	8.3%	33.3%
Casual shirt	0%	12.5%	50%	37.5%
Business shirt	0%	5%	5%	90%
Overall accuracy: 72.7%				

5.3.9 Clothes Matching

After we have the low-level image features and the learning algorithms to recognize the clothes factors, we turn to the problem of determining similarity between objects. We need to weigh the degree to which each factor is salient in human perception — do people favor certain factors when determining similarity? For example, is color similarity more important than collar presence similarity?

To address this problem, we turned to the data from the user study. Each respondent rated the similarity on 40 pairs of men's shirts and 20 pairs of women's shirts. We first coded each of the shirts along each of the six factors. For each pair of shirts, we then calculated a difference score for each factor. A 0 was given for each factor that matched and a 1 for a mismatch. For the color, we generated a score between 0 to 1 that depends on the normalized distance between the two color histograms. Thus, for each respondent's similarity rating, we also had six factor similarity scores.

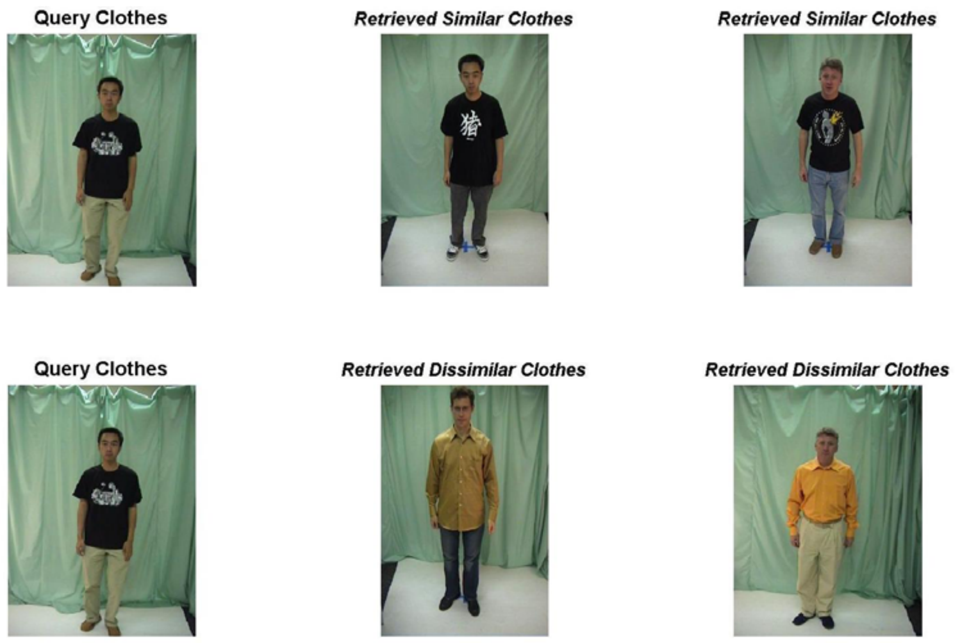
We conducted a linear regression using the ground-truth factor similarity scores to predict the similarity rating scores in order to determine the relative importance of each factor. We conducted separate analyses for men's shirts and women's shirts. For men's shirts, the resulting model was significant. All six factors except emblem were significant. The factors in order of decreasing importance were 1) Collar, 2) Placket, 3) Sleeve, 4) Color, 5) Pattern, and 6) Emblem. For women's shirts, the resulting model was also significant. Due to the limited variation of women's shirts in our dataset, collar, placket, and emblem placement did not vary and thus could not be included as predictor variables. The remaining factors in order of decreasing importance were 1) sleeve length, 2) pattern, and 3) color. The regression model allowed us to generate the weights for each factor that would approximate human perception of shirt similarity. The unstandardized coefficients provide the weights for each factor that we can use to generate similarity scores. For example, the regression equation for similarity ratings of men's shirts is the following:

$\textit{Similarity Rating} = 3.247 + (-\,0.63 \times \textit{Sleeve}) + (-\,0.19 \times \textit{Pattern}) + (2.40 \times \textit{Color}) + (-\,0.88 \times \textit{Collar}) + (-\,0.80 \times \textit{Placket}) + (-\,0.06 \times \textit{Emblem})$.

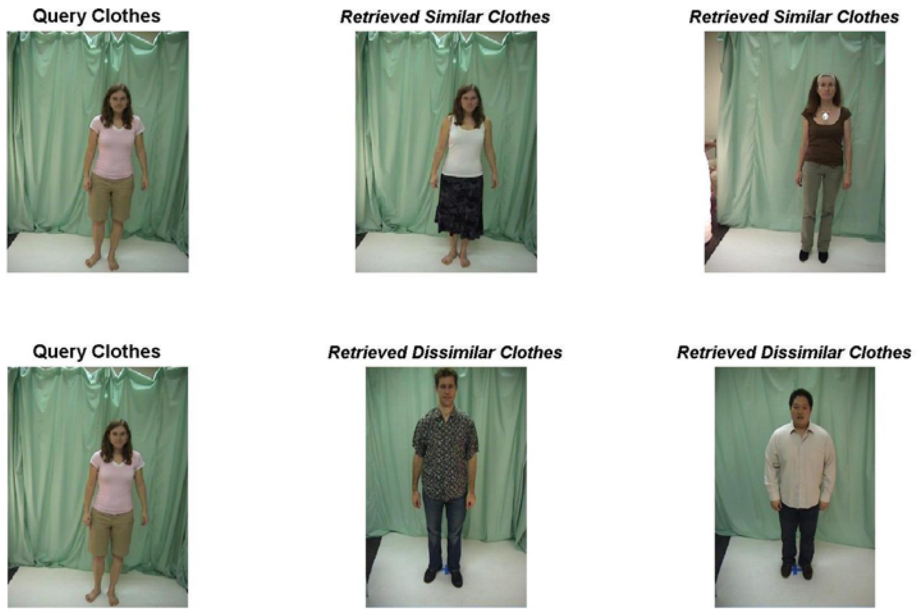
To understand how well the image features generated from our vision algorithms captured the variance in human ratings of shirt similarity, we conducted a linear regression using the 23 underlying features ($\boldsymbol{A} - \boldsymbol{E}$, $\boldsymbol{H}$) to predict the similarity rating. We conducted separate analyses for men's shirts and women's shirts. For men's shirts, the resulting model was significant. For women's shirts, the resulting model was also significant. The regression results show that in the case of men's shirts, the predicted similarity score (using the features detected from the vision algorithms) correlates with the actual ratings at 0.52. Examples of clothes retrieval results using ground-truth factor similarities and image feature similarities are shown in Figure 45.

In summary, this chapter has described the application of image features and learning algorithms on a difficult "social vision problem": clothes recognition and similarity measurement. A variety of image features are extracted from clothes fitting images to identify the clothing factors that are indicated as salient to humans. The approach and results presented here will benefit designers of similar applications in the future.

Currently, the clothes recognition engine is only designed for recognizing shirts. It needs to be generalized to other object classes, such as trousers, handbags, and so on, for real-world applications. The other challenge is integrating the clothes recognition engine with various end-user applications, for example, the RM system (Zhang et al. 2008b; Begole et al. 2008) or hand-held devices (e.g., cell-phone, PDA, etc.). Finally, potentially the clothes recognition engine can be updated according to user feedback to customize the engine to the flavor of each individual user.


Query Clothes
Retrieved Similar Clothes
Retrieved Similar Clothes
Query Clothes
Retrieved Dissimilar Clothes
Retrieved Dissimilar Clothes

(a)


Query Clothes
Retrieved Similar Clothes
Retrieved Similar Clothes
Query Clothes
Retrieved Dissimilar Clothes
Retrieved Dissimilar Clothes

(b)

Figure 45. Examples of clothes retrievals using (a) ground-truth factor similarities and (b) image feature similarities.

6 Conclusions and Future Work

6.1 Conclusions

Recent years have seen significant progress in object recognition with the development of new image feature extractors and classification algorithms. Image features generated by interest region detectors and descriptors are advantageous in their robustness to image transformations, occlusion and background noise. The learning algorithms are designed to be able to fuse the information from disparate types of image features for improved discrimination. This dissertation advanced the research on image features and learning algorithms for the recognition of biological, generic and social objects in the following aspects:

A novel structure-based interest region detector – the PCBR detector – was introduced that complements previous intensity-based detectors. The PCBR detector showed high performance on the identification of consistent regions across different views of complex scenes. The PCBR detector was able to reliably detect the regions corresponding to the salient and characteristic patterns on biological and generic objects. PCBR regions were combined with other types of image features for the recognition of complex objects. The resulting classifiers achieved high accuracies on various challenging recognition tasks.

The second contribution is the novel object recognition architectures that fuse various types of image features to recognize the categories of complex objects. Hierarchical recognition systems based on multi-scale PCBR regions and stacked decision tree ensembles have been successfully applied to the stonefly recognition task.

This dissertation extensively studied a particular class of mid-level approaches – visual codebooks. Visual codebooks allow us to transform bags of low-level image features into fixed-length attribute vectors for image classification. This dissertation first studied the construction, evaluation and application of generative visual codebooks for stonefly recognition and generic object recognition. Then it presented two discriminative visual codebook learning algorithms. The first was the generative/discriminative visual codebook learning approach using the IDC algorithm. The second was the non-redundant codebook learning framework that builds multiple visual codebooks that are complementary to each other for improved discrimination. Both of the new visual codebook learning approaches achieved high performance on the challenging stonefly recognition and generic object recognition datasets.

Finally this dissertation described the application of image features and learning algorithms to the challenging socially-driven clothes recognition and matching problem. This dissertation presented methods to efficiently extract low-level image features and learn image classifiers for the recognition of the key factors in shirts. This socially-driven clothes recognition engine has been successfully employed in a novel intelligent fitting room system – the Responsive Mirror − for the interactive social comparison display.

All of the new approaches presented in this dissertation have been implemented and extensively tested on benchmark object recognition datasets. They are expected to be beneficial for the future research on related topics.

6.2 Future Work

The new image features and learning algorithms presented in this dissertation provide many advantages over previous methods for the recognition of biological, generic and social objects. However, there is still plenty of room for additional improvements. Some of the future directions to explore are listed here.

Learning interest region detectors

Although learning has been successively applied to object recognition, in most of the cases, learning is only limited to the top-level image classifier. On the other hand, learning low-level operations (region detection and description) has received much less attention. The interest region detectors are usually hand-crafted. For some applications, learning at the high level is sufficient given the simplicity of the problem. But for more complex real-world applications, such as the object recognition tasks described in this dissertation, learning in the final step is not sufficient to achieve high accuracy.

To overcome the weakness of the existing handcrafted detectors, researchers have paid more and more attention recently to introducing learning into low-level image operations to develop problem-specific interest region detectors for object recognition. Further research on this topic is expected to be valuable for the development of new high-performance object recognition system that integrates learning in all its components.

Sensitivity analysis of region descriptors

It is commonly accepted that current detectors are not perfect in the localization and shape estimation of interest regions (Haja et al. 2008). The region descriptors affiliate to the interest regions are desired to be robust enough to the transformation of the regions (see Section 2.2.2). This is one of the reasons that the SIFT descriptor (Lowe 2004) achieves such high performance on image matching and object recognition applications: it is robust to planar and affine transformations of images to some degree. This has been demonstrated in the work of Mikolajczyk and Schmid (2005) using recall-precision criteria. But this evaluation is qualitative comparison instead of using quantitative metrics; and different transformations were assumed to be non-correlated, which is not necessarily true for real-world images. Up to now, the systematic quantitative evaluation of the robustness of region descriptors to various image transformations is still unexplored in the community.

Sensitivity analysis (Saltelli et al. 2008) can not only help us to understand the robustness of descriptors to non-perfect detections, but also guide the design or learning of new descriptor.

Learning spatially-constrained visual codebook

All the visual codebook approaches introduced in this dissertation built codebooks in the region descriptor space. They did not explore the spatial distribution of the codewords in image space. The result is that a codeword may not correspond to a consistent part of objects; but instead may correspond to regions spread in different parts of objects but coincidentally having similar appearance. Consequently, these approaches are not robust to the false detections.

Image coordinates of the features have been explored in previous work (Lazebnik et al. 2006) to help the recognition of generic objects in complex scenes. But the absolute image coordinates are not invariant to the transformation of objects across different images. Fortunately, for a lot of object recognition applications, the bounding boxes of the objects are available either by manual labeling (e.g., *PASCAL06*, see Figure 4) or by automated object detection (e.g., the stonefly dataset in Chapter 3 and clothes recognition dataset in Chapter 5). The position and scale of the bounding box can be used to impose transformation-invariant spatial constraints to the learning of codebook. These spatial constraints are able to reject the false detections and improve the performance of the object recognition system.

Bibliography

Agarwal, S., Awan, A. and Roth, D. (2004). Learning to detect ob-jects in images via a sparse, part-based representation. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 26(11), pages 1475-1490.

Anguelov, D., Lee, K., Gokturk, S. B. and Sumengen, B. (2007). Contextual identity recognition in personal photo albums. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1-7.

Begole, B., Matsumoto, T., Zhang, W. and Liu, J. (2008). Responsive mirror: fitting information for fitting rooms. *Workshop on Ambient Persuasion at CHI*.

Belongie, S., Malik, J. and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 24(4), pages 509–522.

Bouchard, G. and Triggs, B. (2005). Hierarchical part-based visual object categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 710-715.

Breiman, L. (1996). Bagging predictors. *Machine Learning,* volume 24(2), pages 123-140.

BugID project: http://web.engr.oregonstate.edu/~tgd/bugid/

Caltech dataset: http://www.robots.ox.ac.uk/~vgg/data3.html

Caltech101 dataset: http://www.vision.caltech.edu/Image_Datasets/Caltech101/

Canny, J. (1986). A computational approach to edge detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 8(6), pages 679-698.

Chen, Y., Bi, J. and Wang, J. (2006). MILES: Multiple-instance learning via embedded instance selection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 28(12), pages 1931-1947.

Christopher, J. C. B. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, volume 2(2), pages 121-167.

Csurka, G., Dance, C. R., Fan L., Willamowski, J. and Bray, C. (2004). Visual categorization with bags of keypoints. *Workshop of European Conference on Computer Vision,* pages 59-74.

Cui, Y., Fern, X. and Dy, J. (2007). Non-redundant multi-view clustering via orthogonalization. In *Proceedings of the IEEE International Conference on Data Mining*, pages 133-142.

Deng, H., Zhang W., Mortensen, E., Dietterich, T. and Shapiro, L. (2007). Principal curvature-based region detector for object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Dietterich, T. and Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, volume 2, pages 263-286.

Dietterich, T., Lathrop, R. and Lozano-Perez, T. (1997). Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, volume 89(1-2), pages 31-71.

Dietterich, T. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, volume 10, pages 1895-1923.

Dietterich, T. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, volume 40 (2), pages 139-157.

Domeniconi, C., Gunopulos, D., Ma, S., Yan, B., AI-Razgan, M. and Papadopoulos, D. (2007). Locally adaptive metrics for clustering high dimensional data. *Data Mining and Knowledge Discovery Journal*, volume 14(1), pages 63-97.

Dorko, G. and Schmid, C. (2004). Object class recognition using discriminative local features. *Technical Report, INRIA-Rhone-Alpes*.

Duda, R. O., Hart, P. E. and Stork, D. G. (2001). *Pattern Classification*, Second edition, John Wiley & Sons, Inc.

Epshtein, B. and Ullman, S. (2005). Feature hierarchies for object classification. In *IEEE International Conference on Computer Vision*, volume 1, pages 220-227.

Fergus, R., Perona, P. and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264-271.

Fergus, R., Perona, P. and Zisserman, A. (2007). Weakly supervised scale-invariant learning of models for visual recognition. In *International Journal of Computer Vision*, volume 71(3), pages 273-303.

Freeman, W. T. and Adelson, E. H. (1991). The design and use of steerable filters. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 13(9), pages 891-906.

Freund, Y. and Schapire, R. (1996). Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148-156.

Freund, Y and Schapire, R. (1999). Large margin classification using the perceptron algorithm. *Machine Learning*, pages 277-296.

GRAZ dataset: http://www.emt.tugraz.at/~pinz/data/GRAZ_01/

Haja, A., Jahne, B. and Abraham, S. (2008). Localization accuracy of region detectors. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147-151.

INRIA dataset: http://www.robots.ox.ac.uk/~vgg/research/affine/

Jain, P., Meka R. and Dhillon I. (2008). Simultaneous unsupervised learning of disparate clusterings. *Statistical Analysis and Data Mining*, volume 1(3), pages 195-210.

Jurie, F. and Schmid, C. (2004). Scale-invariant shape features for recognition of object categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 90-96.

Jurie, F. and Triggs, B. (2005). Creating efficient codebooks for visual recognition. In *IEEE International Conference on Computer Vision*, volume 1, pages 604-610.

Kadir, T., Zisserman A. and Brady, M. (2004). An affine invariant salient region detector. In *European Conference on Computer Vision*, pages 228-241.

Kalal, Z., Matas, J. and Mikolajczyk K. (2008). Weighted sampling for large-scale boosting. In *The British Machine Vision Conference*.

Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: a more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 506-513.

Kienzle, W., Bakir, G., Franz, M. and Scholkopf, B. (2005). Face detection - efficient and rank deficient. In *Proceedings of Neural Information Processing Systems*, pages 673-680.

Landwehr, N., Hall, M. and Frank, E. (2005). Logistic model trees. *Machine Learning*, volume 59(1-2), pages 161-205.

Larios, N., Deng, H., Zhang, W., Sarpola, M., Yuen, J., Paasch, R., Moldenke, A., Lytle, D., Ruiz Correa, S., Mortensen, E., Shapiro, L. and Dietterich, T. (2008). Automated insect identification through concatenated histograms of local appearance features. *Machine Vision and Applications*, volume 19 (2), pages 105-123.

Lazebnik, S., Schmid, C. and Ponce, J. (2006). Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169-2178.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision,* volume 60(2), pages 91-110.

MacKay, D. (2003). *Information theory, inference, and learning algorithms*. Cambridge University Press.

Martínez-Muñoz, G., Zhang, W., Payet, N., Todorovic, S. and Dietterich, T. (2009). Stacked decision tree ensembles for categorizing very similar objects without using visual dictionaries. Accepted to *IEEE Conference on Computer Vision and Pattern Recognition*.

Matas, J., Chum, O., Urba, M. and Pajdla, T. (2002). Robust wide baseline stereo from maximally stable extremal regions. In *The British Machine Vision Conference*, pages 384-396.

Mikolajczyk, K. and Schmid, C. (2004). Scale and affine invariant interest point detectors. In *International Journal of Computer Vision,* volume 60(1), pages 63-86.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T. and Van Gool, L. (2005a). A comparison of affine region detectors. In *International Journal of Computer Vision,* volume 65(1-2), pages 43-72.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 27(10), pages 1615–1630.

Mikolajczyk, K., Leibe, B. and Schiele, B. (2005b). Local features for object class recognition. In *IEEE International Conference on Computer Vision*, volume 2, pages 1792-1799.

Moosmann, F., Triggs, B. and Jurie, F. (2007). Fast discriminative visual codebooks using randomized clustering forests. In *Proceedings of Neural Information Processing Systems*, pages 985-992.

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

OpenCV: http://sourceforge.net/projects/opencvlibrary/

PASCAL06: http://pascallin.ecs.soton.ac.uk/challenges/VOC/databases.html

Perronnin, F. (2008). Univeral and adapted vocabularies for generic visual categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 30(7), pages 1243-1256.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann Publishers Inc.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D. Saisana, M. and Tarantola, S. (2008). *Global sensitivity analysis: the primer*. John Wiley & Sons Ltd.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, volume 24 (5), pages 513–523.

Salton, G. and McGill, M. (1983). *Introduction to modern information retrieval*. McGraw-Hill.

Schmid, D., Mohr, R. and Bauckhage, C. (2000). Evaluation of interest point detectors. In *International Journal of Computer Vision,* volume 37(2), pages 151-172.

Shapiro, L. and Stockman, G. (2001). *Computer vision*. Prentice Hall.

Shental, N., Hertz, T., Weinshall, D. and Pavel, M. (2002). Adjustment learning and relevant component analysis. *European Conference on Computer Vision*, pages 776-792.

Shokoufandeh, A., Marsic, I. and Dickinson, S. J. (1999). View-based object recognition using saliency maps. *Image and Vision Computing*, volume 17(5-6), pages 445–460.

Shotton, J., Johnson, M. and Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Sivic, J. and Zisserman, A. (2004). Video Google: a text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, volume 2, pages 1470-1477.

Song, Y. and Leung, T. (2006). Context-aided human recognition – clustering. In *European Conference on Computer Vision*.

Steger, C. (1998). An unbiased detector of curvilinear structures. in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 20(2), pages 113-125.

Tuytelaars, T. and Van Gool, L. (2004). Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, volume 59(1), pages 61–85.

UIUCCar dataset: http://l2r.cs.uiuc.edu/~cogcomp/Data/Car/

Van de Sande, K., Gevers, T. and Snoek, C. (2008). Evaluation of color descriptors for object and scene recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511-518.

Winn, J., Criminisi, A. and Minka, T. (2005). Object categorization by learned universal visual dictionary. In *IEEE International Conference on Computer Vision*, volume 2, pages 1800-1807.

Yang, L., Jin, R., Sukthankar R. and Jurie, F. (2008). Unifying discriminative visual codebook generation with classifier training for object category recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

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

Zhang, Q. and Goldman, S. (2001). EM-DD: An improved multiple-instance learning technique. In *Proceedings of Neural Information Processing Systems*, volume 14, pages 1073-1080.

Zhang, W., Deng, H., Dietterich, T. and Mortensen, E. (2006). A hierarchical object recognition system based on multi-scale principal curvature regions. In *IEEE International Conference of Pattern Recognition,* volume 1, pages 778-782.

Zhang, W., Begole, B., Chu, M., Liu, J. and Yee, N. (2008a). Real-time clothes comparison based on multi-view vision. *2nd ACM/IEEE International Conference on Distributed Smart Cameras*.

Zhang, W. and Deng. H. (2008). Understanding visual dictionaries via maximum mutual information curves. In *IEEE International Conference of Pattern Recognition*.

Zhang, W. and Dietterich. T. G. (2008). Learning visual dictionaries and decision lists for object recognition. In *IEEE International Conference of Pattern Recognition*.

Zhang, W., Matsumoto, T., Liu, J., Chu, M. and Begole, B. (2008b). An intelligent fitting room using multi-camera perception. *International Conference on Intelligent User Interface*, pages 60-69.

Zhang, W., Yee, N., Liu, J., Chu, M. and Begole, B. (2008c). Augmented fashion exploration based on clothes recognition. Accepted to *Advanced Concepts for Intelligent Vision Systems*.

Zhang, W., Surve, A., Fern, X. and Dietterich, T. (2009). Learning non-redundant codebooks for classifying complex objects. Submitted to *International Conference on Machine Learning*.