THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

APPEARANCE-BASED CLASSIFICATION AND RECOGNITION USING SPECTRAL HISTOGRAM REPRESENTATIONS AND HIERARCHICAL LEARNING FOR OCA

By

QIANG ZHANG

A Thesis submitted to the
Department of Computer Science
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Spring Semester, 2005

The members of the Committee approve the thesis of Qiang Zhang defended on March 22, 2005.

Xiuwen Liu
Professor Directing Thesis

David Whalley
Committee Member

Kyle Gallivan
Committee Member

Approved:

Sudhir Aggarwal, Chair
Department of Computer Science

Donald Foss, Dean, College of Arts and Sciences

The Office of Graduate Studies has verified and approved the above named committee members.

To my wife, Hua ...

ACKNOWLEDGEMENTS

I am deeply indebted to my major professor, Dr. Xiuwen Liu, for his guidance, encouragement, patience and valuable technical help. I am very grateful to my committee members, Dr. Whalley and Dr. Gallivan, for their service and comments.

I benefited greatly from the excellent education at the Florida State University. My thanks go to the many teachers who ushered me in many different directions. They include Dr. Anuj Srivastava, Dr. Mio Washington, Mr. David A. Gaitros, Dr. Lois Hawkes, Mr. Bob Myers and Dr. Xin Yuan. My appreciation extends to the friends who helped me a lot. They include Yibo Cai, Lei Chen, Donghu Sun, Christopher Waring, and many others.

I want to thank my old friend Dr. Feiming Chen, for our ten years friendship.

I am especially grateful to my wife Hua Liang, for her love.

Finally, I want to give special thanks to my parents in China, Jingyu Zhang and Menglin Guan, whose love always seem to be around me although I am thousands miles away from them. Also I want to show my thank to my sister Qian Zhang, for her care to me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This thesis is composed of two parts.

Part one is on Appearance-Based Classification and Recognition Using Spectral Histogram Representations. We present a unified method for appearance-based applications including texture classification, 2D object recognition, and 3D object recognition using spectral histogram representations. Based on a generative process, the representation is derived by partitioning the frequency domain into small disjoint regions and assuming independence among the regions. This gives rise to a set of filters and a representation consisting of marginal distributions of those filter responses. We provide generic evidence for its effectiveness in characterizing object appearance through statistical sampling and in classification by visualizing images in the spectral histogram space. We use a multilayer perceptron as the classifier and propose a filter selection algorithm by maximizing the performance over training samples. A distinct advantage of the representation is that it can be effectively used for different classification and recognition tasks. The claim is supported by experiments and comparisons in texture classification, face recognition, and appearance-based 3D object recognition. The marked improvement over existing methods justifies the effectiveness of the generative process and the derived spectral histogram representation.

Part two is on Hierarchical Learning for Optimal Component Analysis. Optimization problems on manifolds such as Grassmann and Stiefel have been a subject of active research recently. However the learning process can be slow when the dimension of data is large. As a learning example on the Grassmann manifold, optimal component analysis (OCA) provides a general subspace formulation and a stochastic optimization algorithm is used to learn optimal bases. In this paper, we propose a technique called hierarchical learning that can reduce the learning time of OCA dramatically. Hierarchical learning decomposes the original optimization problem into several levels according to a specifically designed hierarchical organization and the dimension of the data is reduced at each level using a

shrinkage matrix. The learning process starts from the lowest level with an arbitrary initial point. The following approach is then applied recursively: (i) optimize the recognition performance in the reduced space using the expanded optimal basis learned from the next lower level as an initial condition, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show that the optimal performance obtained in the reduced space is maintained after the expansion. Therefore, the learning process of each level starts with a good initial point obtained from the next lower level. This speeds up the original algorithm significantly since the learning is performed mainly in reduced spaces and the computational complexity is reduced greatly at each iteration. The effectiveness of the hierarchical learning is illustrated on two popular datasets, where the computation time is reduced by a factor of about 30 compared to the original algorithm.

CHAPTER 1

APPEARANCE-BASED CLASSIFICATION AND RECOGNITION USING SPECTRAL HISTOGRAM REPRESENTATIONS

1.1. Introduction

With the recent development of sophisticated learning algorithms, it has been realized that the performance of a classification and recognition system critically depends on the underlying representation [17, 7]. For example, Geman et al. [17] suggested that "the fundamental challenges are about representation rather than learning" (p. 1); Bishop stated in his book [7] that "in many practical applications the choice of pre-processing will be one of the most significant factors in determining the performance of the final system" (p. 295). The importance of representation was also greatly emphasized by Marr [39].

While the human visual system can recognize/classify different kinds of objects (such textures and faces) based on images effortlessly, deriving a computational model that is effective for variety of objects seems difficult. In the literature, this problem is largely avoided by (artificially) dividing recognition/classification into different problems such texture classification and object recognition and then developing separate models for each problem. In this chapter, we attempt to develop a unified model for appearance-based recognition/classification applications based on a generative process proposed by Grenander and Srivastava [21]. The generative process relates the appearance of imaged objects to the underlying object models. By seeking a compromise between computational complexity and discriminability, we derive a spectral histogram representation, consisting of the marginals of filter responses and also the filters by partitioning the frequency domain into small regions. To demonstrate its generality and effectiveness for appearance-based applications, we show

that the representation can characterize the appearance of different kinds of objects including textures and objects like faces.

To demonstrate the performance of appearance-based applications, we use a multi-layer perceptron (MLP) as the classifier to capture/model variations within each class. Because of the desirable properties of the spectral histogram representation, we obtain good performance on separate test sets for several problems. To be statistically significant, we repeat experiments under one setting many trials, reporting the statistics. Given the marked improvement over existing methods, we argue that the spectral histogram representation may provide a unified representation for appearance-based applications.

The rest of the chapter is organized as follows. In Section 1.2 we derive the spectral histogram representation based on a generative process and provide generic justification through statistical sampling and visualization. Section 1.3 presents a filter selection algorithm within the spectral histogram representation by maximizing the training performance of MLP. In Section 1.4 we show substantial experimental results on three problems, namely, 3D object recognition, face recognition, texture classification, and then on a combined dataset. Section 1.5 concludes the chapter with discussion on a number of issues related to our approach.

1.2. Spectral Histogram Representation

1.2.1. Derivation

The starting point of deriving the spectral histogram representation is the following scenario. Suppose that we have a large number of different objects which may appear on a uniform background according to a Poisson process and the objects are not known explicitly in any other way. We are interested in a translation invariant statistical feature that can be used to characterize the appearance of images. In other words, the derived feature(s) should be effective for classifying and recognizing the large number of objects based on the observed images.

Before we proceed further, we want to point out that important problems in computer vision such as texture modeling and face recognition can be approximated by this simple model. More interestingly, some popular techniques such as eigen decompositions [54] do not satisfy the requirements imposed here as they are not translation invariant.

An obvious choice is the histogram of the given image, which is translation invariant. However, for a large number of objects, their histograms can be very close or even identical, making the histogram not sufficient for recognition and classification. If all the pixel values are statistically identical and independent, then the histogram is the only choice. In appearance-based applications, however, this assumption of independence is not valid as the pixels belonging to one object are dependent.

Another obvious choice is to build one joint probability model of all the pixels for each class. The joint probability model captures completely the statistical dependence among pixels on objects. With these probability models, one can use Bayesian classifier for optimal classification [11]. However, the dimension of the joint space makes the implementation impossible. (For a 128×128 image space, its dimension is 16384 and it has approximately $10^{39,456}$ different realizations assuming 256 values for each pixel; see [33] for some further arguments.)

Here we seek a compromise between the two extreme choices. Instead of assuming that the pixels are independent, we assume that small disjoint regions in the frequency domain are independent. In other words, we partition the frequency domain into small disjoint regions and model the corresponding response of each region in the spatial domain by its histogram. How shall we partition the frequency domain? One sensible way is to partition it into rings with a small range of radial frequencies, as shown in Fig. 1.1 (a), which was proposed by Coggins and Jain [10] to design filters for textures. Another way is to partition it into regions with a small range of radial center frequencies and orientations, as shown in Fig. 1.1 (c), suggested by Jain and Farrokhnia [28]. Yet another way is to partition it into regions with a small range of orientations, which is not explored in this chapter.[1]

These small regions give rise to ideal band pass filters with infinite support in the spatial domain. To make the corresponding spatial filters local and compact, and thus more efficient to implement, we use a Gaussian window function. Under this setting, each ring in Fig. 1.1(a) can be approximated by a difference of Gaussian filter, which can be implemented using a Laplacian of Gaussian (LoG) filter [39], given by:

$$LoG(x, y|T) = (x^2 + y^2 - T^2)e^{-\frac{x^2+y^2}{T^2}}, \tag{1.1}$$

[1]We can also partition the frequency domain to get filters that are sensitive to orientation but not sensitive to radial center frequency. This idea leads to partition the frequency domain into regions of wedge as in [10].

where $T = \sqrt{2}\sigma$ determines the spatial scale of the filter and σ is the variance of the Gaussian window function. These filters are referred to as $LoG(T)$. As examples, Fig. 1.1(b) shows four LoG filters with different scales in spatial domain (bottom row) and their frequency response (top row). As shown in Fig. 1.1(b), LoG filters are band-pass filters, and are sensitive to a given radial center frequency, but not sensitive to orientation. It can also be shown that each small region in Fig. 1.1(c) leads to a Gabor filter [16]. Gabor filters with both sine and cosine components are given by:

$$Gabor(x, y|T, \theta) \;=\; e^{-\frac{1}{2T^2}(4(x\cos\theta+y\sin\theta)^2+(-x\sin\theta+y\cos\theta)^2)} e^{-i\frac{2\pi}{T}(x\cos\theta+y\sin\theta)}, \tag{1.2}$$

where T is a parameter scale. The cosine and sine components of these filters are referred to as $Gcos(T, \theta)$ and $Gsin(T, \theta)$ respectively. Figure 1.1(d) shows four Gabor filters with different orientations and frequencies in spatial domain (bottom row) and their frequency response (top row). Gabor filters are band-pass filters and are sensitive to both orientation and radial center frequency as shown in Fig. 1.1(d).

While the constructed filters may not be entirely statistically independent, the independence is valid to a certain extent for natural images, as recent numerical studies show that Gabor filters (e.g. [42, 25]) and edge detectors (e.g. [4])[2] share similarities with independent components of natural images. Assuming that their responses are statistically independent, the Kullback-Leibler distance between two joint distributions is the sum of their corresponding marginal distributions as shown by:

$$\begin{aligned} &KL(p_1(x_1, \cdots, x_n), p_2(x_1, \cdots, x_n)) \\ &= \int_{x_1} \cdots \int_{x_n} p_1(x_1, \cdots, x_n) \log \tfrac{p_1(x_1,\cdots,x_n)}{p_2(x_1,\cdots,x_n)} dx_1 \cdots dx_n \\ &= \sum_{i=1}^{n} \int_{x_i} p_1(x_i) \log \frac{p_1(x_i)}{p_2(x_i)} dx_i \\ &= \sum_{i=1}^{n} KL(p_1(x_i), p_2(x_i)), \end{aligned} \tag{1.3}$$

where $p_i(x_1, \cdots, x_n)$ is the joint distribution and $p_i(x_j)$ the jth marginal distribution. This gives rise to the following representation for appearance-based applications. We partition the frequency domain into small regions and compute the corresponding spatial filters. For a given image, we convolve it with every filter and compute the marginal distribution of

[2] LoG filters are well known edge detectors [39].

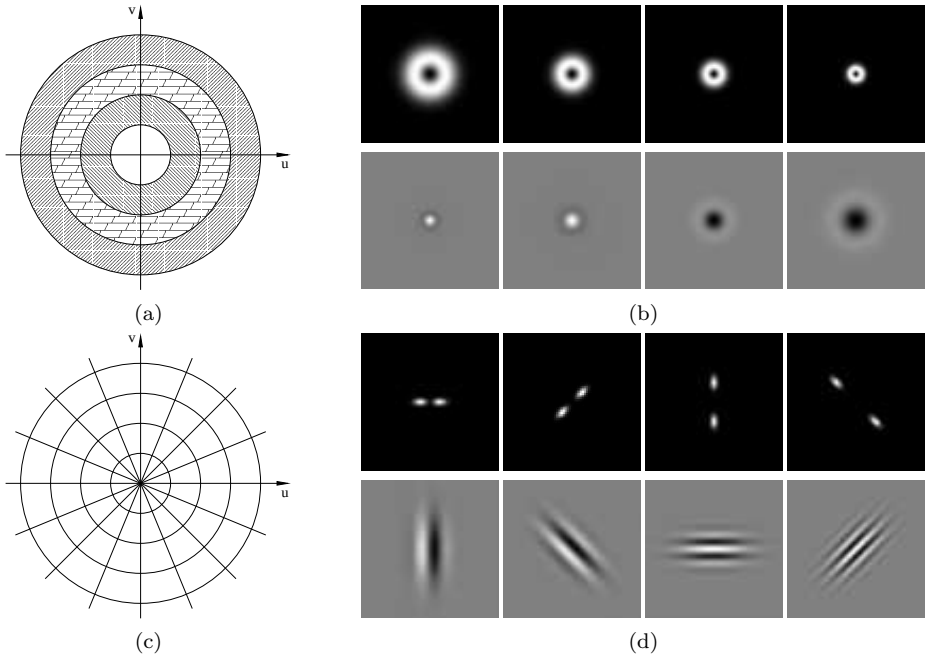


Figure 1.1. Two ways of partitioning the frequency domain. (a) Ring structures. (b) Four LoG filters of different center frequency in spatial domain (bottom row) and their frequency response (top row). (c) Small regions. (d) Four Gabor filters with different orientations and different frequencies in spatial domain (bottom row) and their frequency response (top row).

each response image. We then combine the marginal distributions together as the resulting representation of the input image. The last step is justified by Eq. (1.3). Therefore, in our approach, each image is represented by a vector consisting of the marginal distributions of chosen filter responses. We shall call this representation *spectral histogram representation*[3] of the image with respect to the chosen filters [36].

The generative process used here can be seen as a simplified version of the *transported generator* model proposed by Grenander and Srivastava [21]. The derived representation has also been suggested through psychophysical studies on texture modeling [6] and texture

[3]The naming is based on two considerations: 1) filters are chosen according to a particular spectral representation (i.e., filters are derived by sampling the frequency domain in ways shown in Fig. 1.1, and 2) we use the histogram as a feature to summarize filter responses. In the literature, spectral based representations are also applied directly to texture modeling. For example, by assuming that a texture is a homogeneous random field, it can be decomposed into different homogeneous components, each of which can be characterized by a spectral representation with a unique pattern [15, 32].

discrimination [36], and has been used in the texture modeling and synthesis [23, 58], and texture classification [1, 37]. Both the histogram of input images [53] and joint histograms of local fields [47] have been used for object recognition. Filter selection was studied for modeling a single texture [58] and for texture classification [37].

While this representation can be used directly to characterize rigid objects, objects are often subject to deformations. Here we adopt the learning from examples methodology and use a standard multiple-layer perceptron [24] to learn a model for each class based on the spectral histogram representation.

1.2.2. Generic Justifications

Before we present experimental results on appearance-based applications using real datasets, here we give some generic justifications for the effectiveness of the spectral histogram representation for appearance-based applications from two perspectives. We analyze its intrinsic generalization [35] using statistical sampling and show its clustering of perceptually similar images through visualization.

From the discussion so far, a spectral histogram representation can be seen as a feature vector. If filters are chosen properly, it can be a low dimensional representation of the input image. For a representation to be effective for different kinds of appearance-based applications, it should only group images with similar underlying models together; otherwise, its performance is intrinsically limited. In [35], Liu et al. proposed a way to analyze this by studying a representation's intrinsic generalization. An intrinsic generalization of a representation function for an input image is the set of all the images that share the same representation and it can be studied through statistical sampling, where Gibbs sampler [18] is commonly used.

We apply this technique to analyzing a spectral histogram representation's intrinsic generalization as follows. Given a set of filters $\{F^{(\alpha)}|\alpha = 1, ..., K\}$ and an image I_{obs}, we first compute its spectral histogram, given by $\{H^{(\alpha)}_{I_{obs}}|\alpha = 1, ..., K\}$. Then for any image I, we define an energy function as

$$E(I) = \sum_{\alpha=1}^{K} D(H^{(\alpha)}_{I}, H^{(\alpha)}_{I_{obs}}),$$

and the corresponding Gibbs distribution as

$$q(I) = \frac{1}{Z_\Theta} exp(-\frac{E(I)}{\Theta}).$$

where D is a distance measure between spectral histograms and Θ is a parameter, often called temperature. When $\Theta \to 0$, only images whose spectral histogram is sufficiently similar to $\{H^{(\alpha)}_{I_{obs}} | \alpha = 1, ..., K\}$ will have significant non-zero probability. This allows us to generate samples (which are images in this case) with similar spectral histograms statistically using sampling techniques. Here we use a Gibbs sampler with annealing [18]. The basic idea is to update pixels one by one based on their conditional probability so that the resulting image's probability will improve statistically. For a Gibbs sampler algorithm, see [57]. From the definition of the spectral histogram, it is translational invariant. While this is desirable for recognition, this makes it not possible to align the given image with generated samples. This problem is avoided by using different images as boundary conditions as used in [33].

Figure 1.2 shows an example of a face, shown in Fig. 1.2(a). Here as in other examples, forty filters[4] including LoG and Gabors are used. The Gibbs sampler starts from any image as the initial condition, here a white noise image is used, which is shown in Fig. 1.2(b) with boundary conditions. Figure 1.2(c)-(e) show three realizations by running the Gibbs sampler three times. Note that the essential perceptual characteristics of the input image are captured even though details are missing. As argued in [35], this helps reduce the necessary training samples and improve the generalization performance as its intrinsic generalization has larger cardinality. This point is also emphasized by Vapnik [55].

Figure 1.3 shows four more synthesis examples. In each case, the left shows the given image and the rest are examples from the Gibbs sampler. In all these examples, the local features as well as the global configuration are captured by the spectral histogram. As the spectral histogram only groups perceptually similar images in an image's intrinsic generalization, it should be effective for appearance-based applications, which is consistent with the empirical evidence shown in Section 1.4.

Another important aspect for appearance-based applications is that the distance between images from the same class should be smaller than that between images from different classes. For a particular application, this depends on the choice of training and test sets. Here we use an example to show why it is more effective to do classification/recognition in a spectral

[4]For synthesis, the intensity filter is also included, whose corresponding component in the representation is the histogram of the given image.

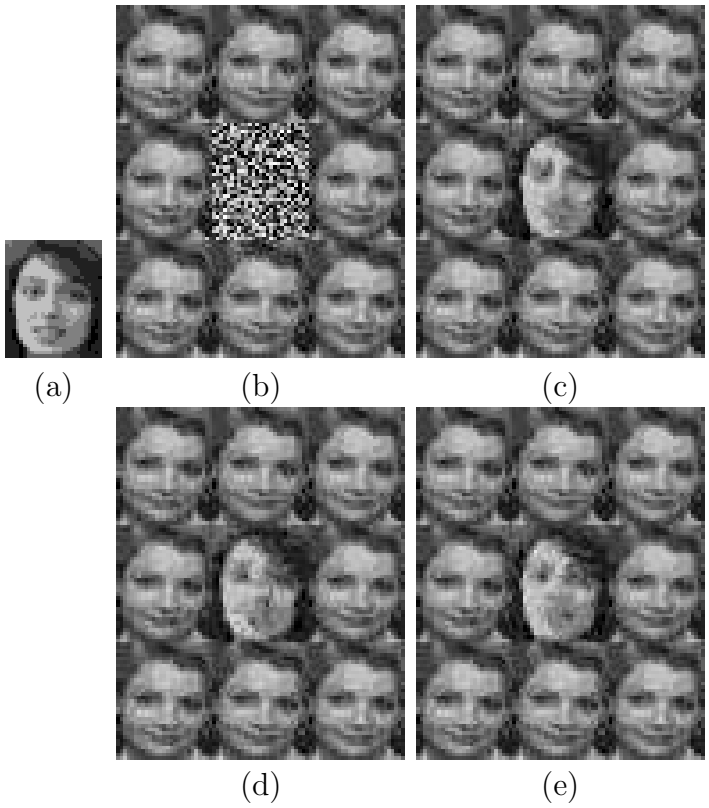


Figure 1.2. An object synthesis example. (a) The given image. (b) The initial image (a white noise one) with boundary conditions used for synthesis. (c)-(e) Three samples by running the Gibbs's sampler three times. Note that the perceptual characteristics of the input are captured even though details are missing.

histogram representation space than in the original image space. In this example, we have 500 images from five classes, with 100 in each class, some from which are shown in Fig. 1.4. Note that each class corresponds a perceptually meaningful texture class. To visualize the five images in the original image space as well as the spectral histogram space, we first apply principal component analysis technique to reduce the dimension to two. Then we project each image into the respective low dimensional space. Figure 1.5(a) shows the 500 images in the principal space of the original image space and Fig. 1.5(b) shows that of the spectral histogram space. As the images from the same class do not form meaningful clusters in the original image space, it is difficult to achieve good generalization performance regardless

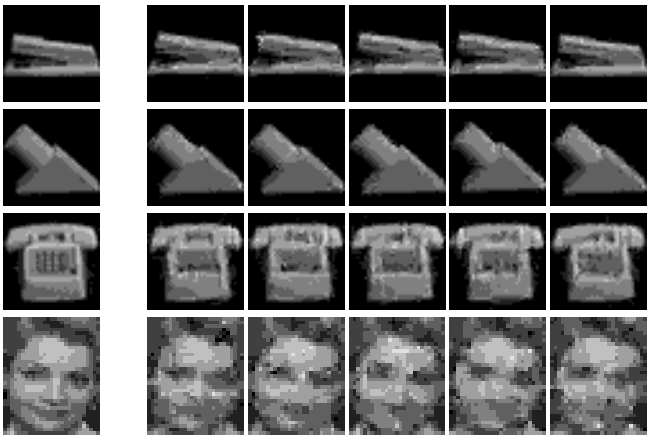


Figure 1.3. More object synthesis examples. Here different objects are used as boundary conditions as in Fig. 1.2. Each row corresponds one object. (a) The given images. (b) In each row, five realizations of the corresponding image by running the Gibbs sampler five times.

of the choice of the classifier. On the other hand, as each class forms a tight cluster in the spectral histogram space, any reasonable classifier would achieve good generalization performance as the good performance on the training would imply good performance on the test. This shows one distinctive advantage of the spectral histogram representation. Of course, its effectiveness for appearance-based applications needs to be demonstrated as datasets typically consist images with significant variations and empirical evidence is shown in Section 1.4.

1.3. Filter Selection

Given a large number of filters, it has been shown that the marginal distributions are sufficient to represent an arbitrary image up to a translation [58, 37]. Intuitively each filter imposes some constraints on all the images that have the same marginal distributions, and with sufficiently many filters, all the images with the same marginal distributions will be identical up to a translation. This can also be seen from the corresponding frequency domain representation. With more and more filters, the frequency response can be reconstructed and thus the original image (through an inverse Fourier transform). Therefore, under the spectral

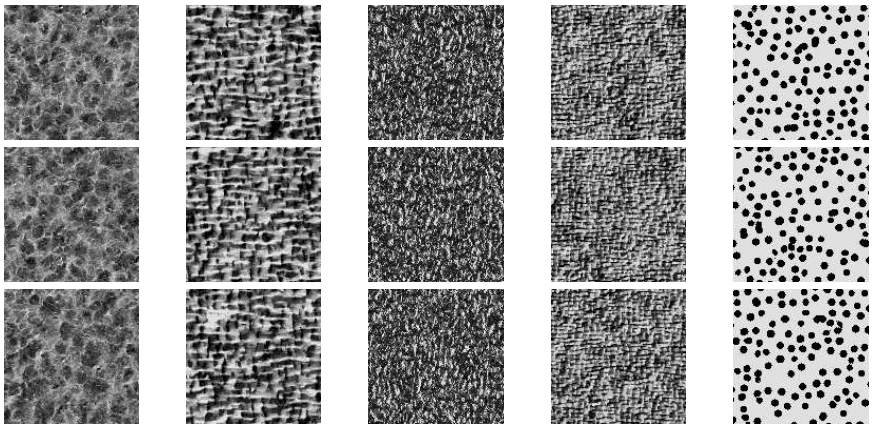


Figure 1.4. Three examples from each of the five classes.

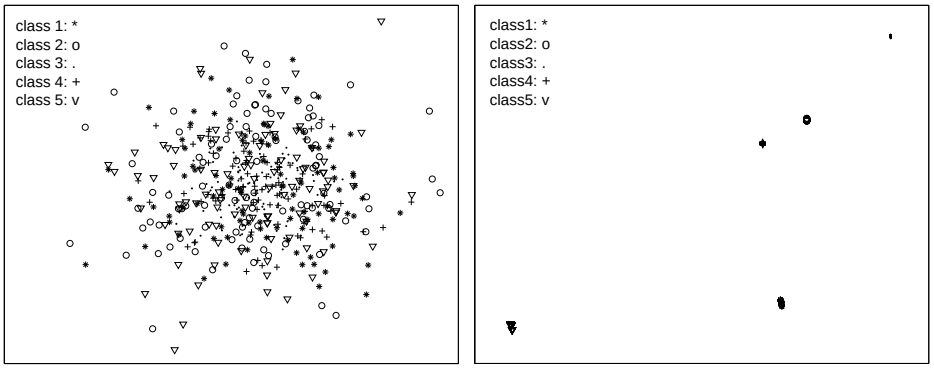


Figure 1.5. The 500 images in the image space (a) and in the spectral histogram space (b). Here only the two most prominent dimensions are displayed. Note that many points in (b) are overlapped.

histogram representation, a critical question for image classification and recognition is how to select filters such that the performance is optimized and the computational complexity is minimized. The basic idea here is simple, similar to that in [37]. We optimize the performance and complexity on the given training data, formally known as empirical risk minimization [55] by selecting most effective filters. As histogram operations are nonlinear [37], this makes an analytical solution for filter selection difficult. Instead, we seek a numerical procedure to select filters for MLPs.

Initially, we train one MLP using back propagation for each filter and we choose the first filter to be the one that gives the minimum training error. Here, the training samples are represented by their spectral representation of the corresponding filter. We then iteratively choose filters one by one: for every unselected filter, we train one MLP and choose the one that mostly improves the performance. Here the training data are the spectral representations of the training images of the already selected filters and the corresponding filter. The filter selection stops when the error is less than some predefined threshold. The algorithm is summarized in Fig. 1.6. In Fig. 1.6, W are weights of filters that are used to incorporate prior knowledge of filter preference. For example, small filters might be preferred over large filters for computational reasons and rotation invariant filters may be preferred for a rotational invariant representation. This can be achieved by weighting the training error with the prior information as shown in Fig. 1.6.

This greedy algorithm is computationally efficient but may not guarantee an optimal solution. Because the spectral histogram representation is robust, this procedure works well on all the datasets we have used. As demonstrated in Section 1.4, the filters selected by our algorithm for each dataset seem effective to capture discriminative features of datasets.

1.4. Results and Comparison

In this section, we demonstrate the effectiveness of our proposed method on three different problems. For all the datasets used here, we start with 39 filters including: 1) 9 Laplacian of Gaussian filters of different scales, and 2) 30 Gabor filters of five different scales with six orientations at each scale. We exclude the intensity filter, corresponding to the histogram of the input image, to make our representation more illumination invariant. The neural network used here is a standard three-layer perceptron trained using the back-propagation

Filter Selection Algorithm

$S = \phi$
$B = \{F^{(1)}, \cdots, F^{(K)}\}$
$W = \{W^{F^{(1)}}, \cdots, W^{F^{(K)}}\}$
repeat
  for each filter $F \in B$
    Train a MLP with filters $S \cup \{F\}$
    Compute the training error e^F
  end
  $F^* = min_{F \in B}\{e^F \cdot W^F\} \quad e^* = e^{F^*}$
  $S = S \cup \{F^*\} \quad B = B \setminus \{F^*\}$
until $e^* < \epsilon$

Figure 1.6. Filter selection algorithm. Here B is the set of all the candidate filters, S is the set being chosen, W is the set of prior weights for filters, and ϵ is a threshold.

algorithm [24]. While the number of input units is determined by the filters chosen by the selection algorithm, the hidden units are fixed to be 40.

1.4.1. 3D Object Recognition

We have applied our method to appearance-based 3D object recognition. For evaluation of our method and comparison with existing methods, we use the Columbia Object Image Library (COIL-100)[5] dataset, which consists of the color images of 100 3-D objects with varying pose, texture, shape and size. For each object there are 72 images taken at different view angles with 5° apart. Therefore there are 7,200 color images in the entire dataset. Figure 1.7 shows the 100 objects from the database from one fixed view angle.

A number of appearance-based schemes have been proposed to recognize 3D objects and applied to the COIL dataset. Murase and Nayar [41] proposed a parametric method to recognize 3D objects and estimate the pose of the object at the same time. Pontil and Verri [44] applied Support Vector Machines (SVMs) [55] to appearance-based object recognition and their method was tested using a subset of the COIL-100 dataset with half for training and the other half for testing. As pointed out by Yang et al. [56], this dense sampling of training views made the recognition less challenging. Yang et al. [56] applied Sparse Network

[5]Available at http://www.cs.columbia.edu/CAVE.

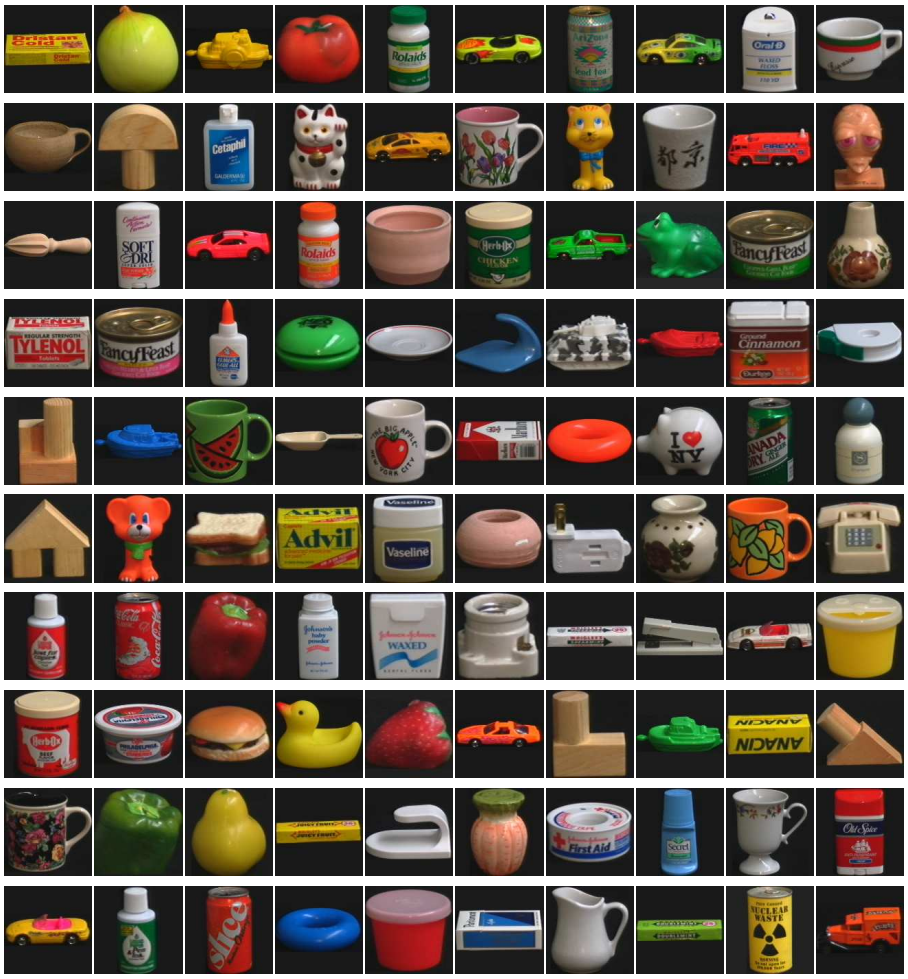


Figure 1.7. The 100 3-D objects in the COIL database. Each image is a color image of 128×128.

of Winnows (SNoW) to recognition of 3D objects and they used the full set of COIL-100 dataset and compared their method with SVM.

As in [56], we vary the number of training views per object to make the 3D recognition more challenging. Given the images in the training set, we apply our filter selection algorithm starting with 39 filters. It is interesting to note the filters selected by our algorithm. It first chose a LoG filter at the largest scale, then chose four Gabor filters at the largest scale with different orientations, and then another LoG filter and a Gabor filter at a smaller scale. This seems consistent with our intuition. Those objects do not prefer a particular orientation

Table 1.1. Recognition results of different methods using the 100 objects in the COIL-100 dataset

Methods	# of training views per object			
	36	18	8	4
	3600 tests	5400 tests	6400 tests	6800 tests
Our Method	100.00%	99.50%	96.33%	84.76%
SNoW[56]	95.81%	92.32%	85.13%	81.46%
Linear SVM[56]	96.03%	91.30%	84.80%	78.50%
Nearest Neighbor[56]	98.50%	87.54%	79.52%	74.63%

and the global patterns and shapes are the most effective for recognition as most of the objects contain uniform surfaces, making the local patterns ineffective for discrimination among different objects.

With the chosen filters, an MLP is trained and the learned network is then used to recognize the testing images at novel views. The unit with the highest output is taken as the result from the system. Table 1.1 shows our recognition results using different number of training views along with the results reported in [56]. With eight views for training, our system gives a correct recognition rate of 96.3%. If we allow the correct to be within the closest five, the correct recognition rate is 99.0%.

We have also compared our recognition results with other methods in [56]. As shown in Tab. 1.1, our method gives the best result under all the test conditions and improves significantly when fewer training views are used. This improvement is essentially because our representation is more meaningful than pixel- and edge-based representations used in [56]. Note that the nearest neighbor result is based on the pixel-wise distance between images. Its performance confirms the images from the same class do not form clusters in the original image space, consistent with Fig. 1.5.

One of the distinct advantages of the spectral histogram representation is its invariance to a number of changes. As is easy to see from its definition, it does not depend on the positions of the target object. In other words, the representation is translation invariant. On the other hand, Pontil and Verri[44] stated that their method can only tolerate a small amount of positional shifts. The intensity filter is excluded from all our experiments here and this makes our representation more illumination invariant because all the filters we use are derivative filters and their responses do not depend on the absolute pixel values. By

preferring filters that are rotation and scale invariant, we can also make our representation rotation and scale invariant. Scale invariance, however, is problematic for SVM and SNoW methods [56] as the input dimension must be the same in order to perform dot product of the kernel functions.

One of the disadvantages of the spectral histogram representation is the required computation if implemented on serial computers as the convolution operation is time consuming. This problem can be alleviated substantially by estimating the histograms based on a subset of pixels. For example, we can do convolution only at pixels on a $m \times m$ grid and thus reduce the convolution time by m^2. Figure 1.8(a) shows the recognition error rate in percentage and the average computation time for recognition with respect to the grid size. Here 8 views are used for training and the rest 64 views are used for test. As shown in Fig. 1.8(a), we can reduce the computation time dramatically, but the recognition performance does not change quickly even though the recognition error increases as the grid size increases. But if we consider the correct being within the five closest, the error increase is within one percent, as shown by the dashed line shows in Fig. 1.8(a). As our method is a bottom-up method, this shows it is effective for candidate shortlisting (see e.g. [51]) for a wide range of m. If we compare this result with other methods shown Tab. 1.1, our recognition performance is still beyond 90.0% for grid size up to 8, significantly better than the best from other methods (which is 85.13%).

To show that our method is insensitive to the choice of classifier configurations, we repeat the experiment shown in Fig. 1.8(a) but with one MLP for each class. In other words, we train simultaneously 100 MLPs, one for each class and the label is assigned to the class with the highest output. The results are shown in Fig. 1.8(b). Compared to that shown Fig. 1.8(a), they are very similar, supporting our claim that the choice of the classifier is not critical.

1.4.2. Face Recognition

In recent years, the problem of face recognition has been studied extensively in the literature (see [9] for a survey). Our method is different from most existing methods for face recognition in that it is a general method for image classification and recognition. However,

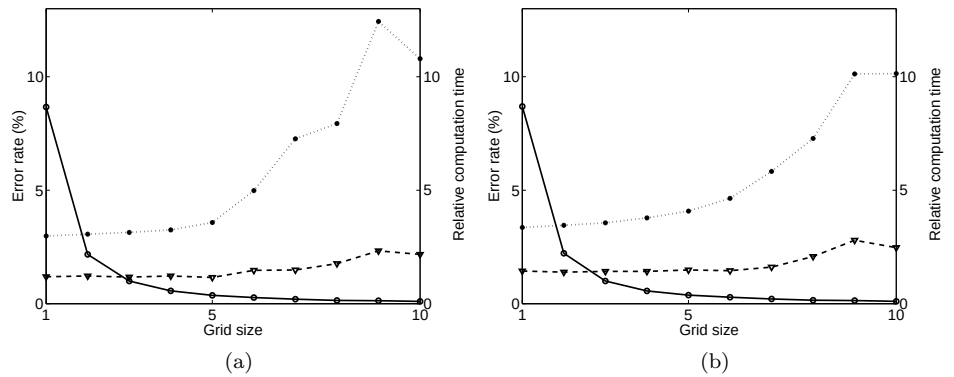


Figure 1.8. The computation time and recognition error rate in percentage with respect to grid size. In each plot, solid line is the relative computation time; dotted line is the error rate with the correct class being the first; dashed line the error rate with the correct being among the five closest. (a) One MLP for all the classes. (b) One MLP for each class.

due to the perceptually meaningful representation, our method is also very effective for face recognition as demonstrated using a face recognition dataset.

Here we use the ORL database of faces[6]. The dataset consists of faces of 40 different subjects with 10 images for each subject. The images were taken at different times with different lighting conditions on a dark background. While only limited side movement and tilt were allowed in this dataset, there was no restriction on facial expression. All the subjects are shown in Fig. 1.9(a) and all the 10 images of a particular subject are shown in Fig. 1.9(b) to demonstrate the variations of facial expression and lighting conditions.

Because there are only 10 faces per subject, we use 5 of them as training and the remaining 5 for testing. As some images are more representative for a subject than others, we randomly choose training faces to avoid the potential bias on the performance. We then repeat the same procedure many times to have a statistically more significant evaluation. It is interesting to see the filters selected by our algorithm for this dataset. It chose four Gabor filters at the largest scales to characterize the global face patterns and two Laplacian of Gaussian filters whose scales are comparable with local facial patterns. Table 1.2 shows the recognition results for 100 trials. Here we report the average, the best and worst performance among

[6]http://www.uk.research.att.com/facedatabase.html

(a)

(b)

Figure 1.9. ORL face database. The size of the images is 112×92. (a) 40 subjects in the database. (b) 10 face images of one subject taken at different facial expression and illumination conditions.

the 100 trials. On average we have achieved over 95% correct recognition rate. Compared to a study using hidden Markov models [46], where the topographic configuration of faces is incorporated, our method outperforms theirs by a large margin.

Table 1.2. Recognition results for the 40 face dataset shown in Fig. 1.9

Criterion	Average correct rate	Best correct rate	Worst correct rate
Correct to be the first	95.4 %	98.5%	90.5%
Correct within the first three	98.9%	100%	96.0%

To show the effectiveness of the filter selection algorithm, we compute the recognition rate with different number of chosen filters. The result is shown in Tab. 1.3. Note that while the selection algorithm is based on the training set only, the performance of the test set also improves initially with the number of filters chosen. As shown in the table, the selection algorithm selects six most effective filters and they give the best average performance over 100 trials.

Table 1.3. Recognition performance with respect to number of selected filters

Number of filters selected	Average error rate	Best error rate	Worst error rate
1	66.29%	70.60%	60.12%
2	87.11%	89.40%	84.46%
3	91.14%	93.21%	87.86%
4	93.78%	95.48%	91.13%
5	94.27%	95.89%	90.54%
6*	95.40%	98.50%	90.50%
39	95.20%	97.14%	92.14%

1.4.3. Texture Classification

Without any change, we have also applied our method to the problem of texture classification, which has been studied extensively as a separate topic in computer vision. We argue that texture models should be consistent with perceptual models for objects as they need to be addressed within one generic recognition system; we demonstrate here that our method can be applied equally well to the texture classification problem.

To demonstrate the effectiveness of our approach, we use a dataset consisting of 40 textures, as shown in Fig. 1.10. Each texture image is partitioned into non-overlapping patches with size 32×32 and then all the obtained patches are divided into a training set and a test set with no common patch between the two sets. As for 3D object recognition and face recognition, we start with the same 39 filters and apply our filter selection algorithm on the training set. The network trained with the chosen filters is then used to classify the patches in the test set. To avoid a bias due to the choice of the training set, we randomly choose the training set for each texture and run our algorithm many times for a better evaluation. We also change the number of patches in the training set to demonstrate the generalization capability of our representation.

Table 1.4 shows the classification result with 100 trials for each setting. Compared to the filters chosen for COIL-100 and ORL datasets, our filter selection algorithm chose filters whose scale is comparable with dominant local texture patterns. This dataset is very challenging in that some of textures are perceptually similar to other textures in the dataset and some are inhomogeneous with significant variations. With as few as 8 training patches, our method achieves a correct classification rate of 92% on average. With half of the patches

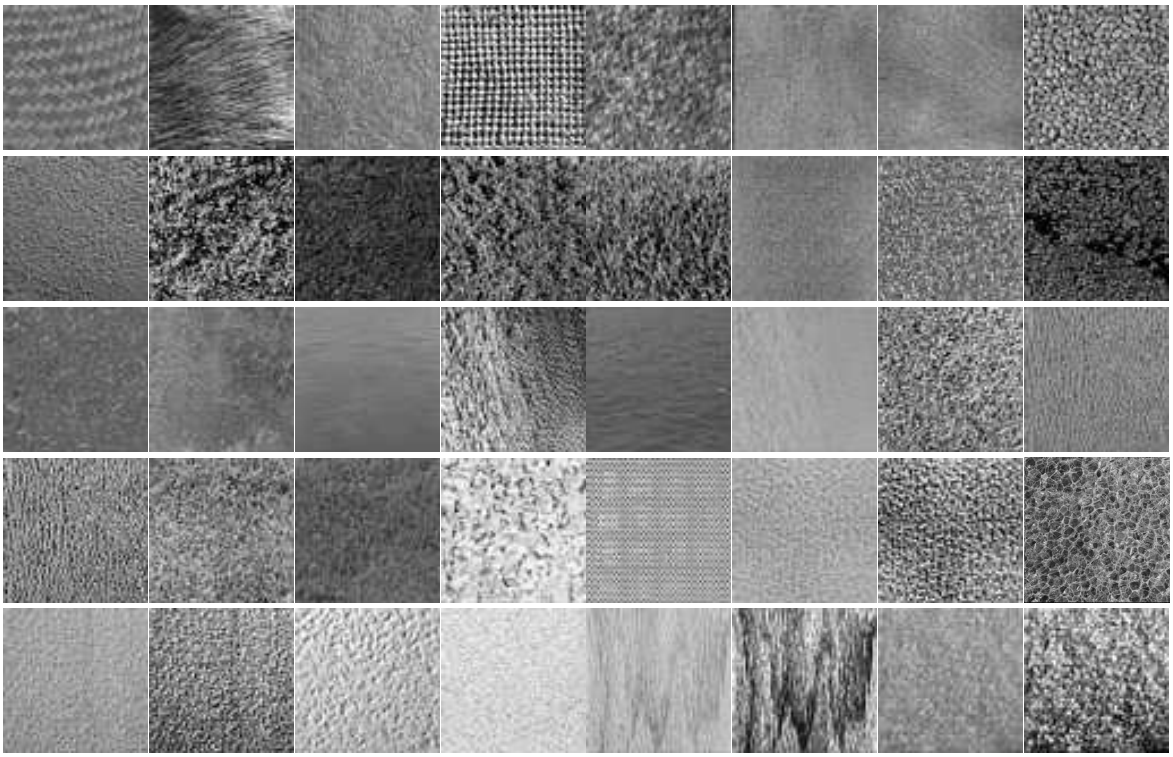


Figure 1.10. Forty natural textures used in the classification experiments. The input image size is 256 × 256. These images are available at http://www-dbv.cs.uni-bonn.de/image/texture.tar.gz.

used for training, we achieve an average classification rate over 96%. Table 1.5 is more convincing, which shows the correct classification rate when the correct is within the closest three. The worst performance is above 97%, demonstrating the good generalization of our system.

To further demonstrate the effectiveness of our method and compare with existing methods, we apply our method to the two datasets that were shown to be very challenging for all the methods included in a recent comprehensive comparative study [45]. Randen and Husoy [45] studied and compared close to 100 different methods for texture classification. For the dataset shown in Fig. 11(h) in [45], Fig. 1.11(a) shows the correct classification rate of all the methods with an average of 52.55% and best being 67.70%. Similarly, for the other dataset shown in Fig. 11(i) in [45], the average correct classification rate is 54.02% with

Table 1.4. Classification results for the 40-texture dataset shown in Fig. 1.10

Test-to-training ratio	Average correct rate	Best correct rate	Worst correct rate
56/8	92.07%	94.20%	90.22%
48/16	94.74%	95.83%	93.07%
42/22	95.64%	96.73%	94.35%
32/32	96.36%	97.42%	95.16%

Table 1.5. Classification results of the correct within the closest three for the 40-texture dataset shown in Fig. 1.10

Test-to-training ratio	Average within three	Best within three	Worst within three
56/8	98.70%	99.42%	97.86%
48/16	99.32%	99.69%	98.70%
42/22	99.52%	99.94%	99.11%
32/32	99.62%	99.92%	99.14%

the best 72.20%, plotted here in Fig. 1.11(b). For datasets of 10 textures, a classification error of 33.3% is very significant. For a fair comparison, we apply our method to the same dataset[7]. As in the original experiment setting, we use a training set to train the neural network with filter selection. The learned network is then applied to a separate test set. The results from our methods are summarized in Tab. 1.6. Because the texture images are perceptually quite similar, an accurate perceptual texture model is needed in order to classify the textures correctly. In addition, two different sets of textures for training and testing make the classification even more difficult. The significant improvement demonstrates the necessity of a perceptually meaningful model for texture classification such as the one proposed here.

Table 1.6. Classification results of the two datasets used in [45]

dataset	Correct to be the first	Correct within first two	Best result from[45]
Fig. 11(h)[45]	93.49%	97.75%	67.70%
Fig. 11(i)[45]	92.96%	98.34%	72.20%

[7]Available at http://www.ux.his.no/∼tranden.

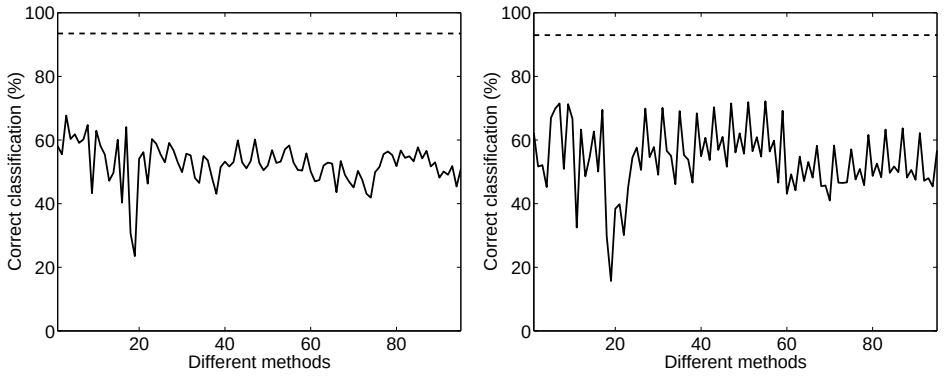


Figure 1.11. The correct classification rate of all the methods studied in [45] on the texture dataset shown in Fig. 11(h) of [45]. The dashed line shows our result for comparison.

Table 1.7. Recognition rate for the combined dataset

	Total training / test images		
	5080 / 5080	2560 / 7600	1300 / 8860
Correct to be the first one	99.37%	97.97%	94.09%
Correct within the first three	99.92%	99.21%	99.73%

1.4.4 Combined Dataset

To further demonstrate the generality of the spectral histogram representation, we have combined the three datasets together to form one with 40 texture types, 100 3-D object classes, and 40 face classes with a total of 10,160 images. In order to perform well on this dataset, the representation must be able to discriminate among different types and within each type, therefore a challenging problem. However, as the spectral histogram representation can capture the perceptual characteristics of faces, textures, and objects as shown in Sect. 1.2.2, we expect that it should work well. Table 1.7 shows the recognition result on the combined dataset with respect to different settings. If we consider to be correct if the correct class is among the closest three, the recognition error here under all the cases is over 99%. This further demonstrates the generality of the proposed approach for appearance-based applications.

1.5. Summary and Discussion

This chapter has presented spectral histogram representations for classification and recognition of images derived from a generative process. While the derived model is simple, it is effective for different appearance-based applications that have been primarily studied separately. The marked improvement of our method over existing ones, along with the image synthesis and clustering visualization results, justifies the effectiveness and generality of the spectral histogram representation. Not only is our approach generic as demonstrated through different datasets of real images, the representation also provides other advantages such as illumination, rotation, and scale invariance by choosing proper filters.

Our representation along with the filter selection algorithm provides a unified framework for appearance-based object recognition and image classification. Within this framework, the difference among general object recognition, face recognition, and texture classification is the choice of the most effective filters. While filters with large scales are most effective for face recognition as faces are topographically very similar, filters whose scales are comparable with texture elements are most effective for texture classification. Our filter selection algorithm chooses the most effective set of filters in this regard. This may lead to a system that is effective for different types of images, which is a key requirement for a generic recognition system.

Our representation has been derived and proposed not as an ultimate solution to the classification and recognition problem. Rather, it is proposed as an effective bottom-up feature statistic which can prune irrelevant templates for more accurate top-down matching methods [51]. In this regard, the filter responses can also be used as top-down templates, an example of which was implemented by Lades et al. [31] for object recognition in general and face recognition in particular. This is also consistent with our generative process discussed in Sect. 1.2. With marginal distributions as bottom-up feature statistics and filter responses as templates, the top-down and bottom-up solutions can be integrated in a coherent framework.

In this chapter, we have chosen MLPs as the classifier for our experiments. Recently support vector machines (SVMs) [55] seem to provide better performance for some applications. As discussed and shown earlier, the desirable properties of the spectral histogram representation makes the choice of classifiers not critical. To further justify our claim, we

have also used an SVM on the face recognition problem using the spectral histogram. Because the SVM is designed for two-class problems, it needs to be extended to the multiple class problems. There are two commonly used methods [29]: 1 against the rest and 1 against another. For a K class problem, in the first case, one needs K SVMs and in the second case, one needs $K*(K+1)/2$ SVMs. We implemented both methods and the results are shown in Tab. 1.8. Compared to the result using an MLP shown in Tab. 1.2, there is no significance between the MLP result and the SVM result using one against the rest.

Table 1.8. Recognition results for the 40 face dataset shown in Fig. 1.9

Criterion	Average correct rate	Best correct rate	Worst correct rate
1-against-the rest	94.8%	98.0%	89.0%
1-against-another	91.0%	95.0%	85.5%

While the spectral histogram representation is insensitive to some amount of distortions, the appearance of an object may be heavily influenced by external conditions such as lighting conditions, view angles, view points, and by other objects. This may not impose a serious problem to our method as long as training examples under different conditions are available as is required by all example-based systems. However, a more effective alternative is to utilize the fact that the marginal distributions are functions of those conditions. By modeling the generative process under those conditions parametrically or numerically, inference can be done in the marginal distribution space, which in general is smooth with respect to the parameters. Our preliminary results on pose estimation are encouraging and we are currently investigating how to use our representation for multiple object recognition in a cluttered environment.

CHAPTER 2

HIERARCHICAL LEARNING FOR OCA

2.1 Introduction

Optimization problems on manifolds such as Grassmann [2, 20] and Stiefel [52] have been a subject of active research recently. For example, Smith [48] explored the geometry of the Stiefel manifold in the context of optimization problems and subspace tracking. Edelman et al. [12] developed new Newton and conjugate gradient algorithms on the Grassman and Stiefel manifolds. For more examples that use Grassmann and Stiefel manifolds see e.g. [50] for subspace tracking, [38] for motion and structure estimation, and [3, 13] for neural network learning algorithms. However the learning process can be slow when the dimension of the data is high. As a learning example on the Grassmann manifold, optimal component analysis (OCA) [34] provides a general subspace formulation and a stochastic optimization algorithm is applied to learn the optimal basis. In this chapter, we propose a technique called hierarchical learning that can reduce the learning time of OCA dramatically. Hierarchical learning decomposes the original optimization problem into several levels according to a specifically designed hierarchical organization and the dimension of the data is reduced at each level using a shrinkage matrix. The learning process starts from the lowest level with an arbitrary initial point. The following approach is then applied recursively: (i) optimize the recognition performance in the reduced space using the expanded result of optimal basis of the next lower level as an initial point, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show that the optimal performance obtained in the reduced space is maintained after the expansion. Therefore, the learning process of each level starts with a good initial point obtained from the next lower level. This speeds up the

original algorithm significantly since the learning is performed mainly in reduced spaces and the computational complexity is reduced greatly at each iteration. The effectiveness of the hierarchical learning is illustrated on two popular datasets, where the computation time is reduced by a factor of about 30 compared to the original algorithm.

The remainder of the chapter is organized into four sections. In Section *II* we review the relevent issues regarding optimal component analysis. The hierarchical learning is presented in Section *III*. Experimental results are given in Section *IV*. Section *V* concludes the chapter with a discussion.

2.2 Review of Optimal Component Analysis

As a learning example on the Grassmann manifold, optimal component analysis provides a general subspace formulation and a stochastic optimization algorithm is applied to learn the optimal basis. Comparing to principal component analysis [54], independent component analysis [26] [27] and Fisher discriminant analysis [5], optimal component analysis has shown its advantage in solving object recognition problems on some datasets. More specifically, in [34], the performance function F is defined in the following way. Let there be C classes to be recognized from the images; each class has k_{train} training images (denoted by $I_{c,1}, \ldots, I_{c,k_{train}}$) and k_{test} test images (denoted by $I'_{c,1}, \ldots, I'_{c,k_{test}}$) to evaluate the recognition performance measure.

$$F(U) = \frac{1}{Ck_{test}} \sum_{c=1}^{C} \sum_{i=1}^{k_{test}} h(\rho(I'_{c,i}, U) - 1). \tag{2.1}$$

In our implementation, $h(x) = 1/(1 + \exp(-2\beta x))$ and

$$\rho(I'_{c,i}, U) = \frac{\min_{c' \neq c, j} d(I'_{c,i}, I_{c',j}; U)}{\min_j d(I'_{c,i}, I_{c,j}; U) + \epsilon} \ . \tag{2.2}$$

Here

$$d(I_1, I_2; U) = \|\alpha(I_1, U) - \alpha(I_2, U)\|, \tag{2.3}$$

$\|\cdot\|$ denotes the 2-norm,

$$\alpha(I, U) = U^T I, \tag{2.4}$$

and $\epsilon > 0$ is a small number to avoid division by zero. As stated in [34], F is presicely the recognition performance of the nearest neighbor classifier when we let $\beta \to \infty$. The gradient vector of F at any point U is defined to be a skew-symmetric matrix given by:

$$A(U) = (\sum_{i=1}^{d} \sum_{j=d+1}^{n} \alpha_{ij}(U) E_{ij}) \quad \in \Re^{n \times n}, \tag{2.5}$$

where

$$\alpha_{ij}(U) = \lim_{\epsilon \downarrow 0} \left(\frac{(F(e^{\epsilon E_{ij}} U) - F(U))}{\epsilon} \right) .$$

α_{ij}s are the directional derivatives of F in the directions given by E_{ij}, respectively. Here E_{ij} is an $n \times n$ skew-symmetric matrix such that: for $1 \leq i \leq d$ and $d < j \leq n$,

$$E_{ij}(k,l) = \begin{cases} 1 & \text{if } k = i,\ l = j \\ -1 & \text{if } k = j,\ l = i \\ 0 & \text{otherwise .} \end{cases} \tag{2.6}$$

They form an orthogonal basis of the vector space tangent to $\mathcal{G}_{n,d}$ at identity.

The deterministic gradient flow is a solution of the following equation:

$$\frac{dX(t)}{dt} = A(X(t)), \quad X(0) = U_0 \in \mathcal{G}_{n,d} . \tag{2.7}$$

In [34], a Markov chain Monte Carlo (MCMC) type stochastic gradient-based algorithm is used to find an optimal subspace $\hat{U}$[1]. At each iteration, the gradient vector of F with respect to U, which is a skew-symmetric matrix, is computed. By following the gradient, a new solution is generated, which is used as a proposal and is accepted with a probability that depends on the performance improvement. If the performance of the new solution is better than the current solution, it is always accepted. Otherwise, the worse the new solution's performance, the lower the probability the solution is being accepted. The computational complexity C_n of each iteration of this algorithm is $C_n = O(d \times (n-d) \times k_{test} \times k_{training} \times n \times d)$. C_n is obtained by the following computation. $d \times (n-d)$ is the dimension of the gradient vector. For each dimension and for each test image, the closest images in all the classes need to be found to compute the ratio in Eqn. 2.2 and to compute the performance F in Eqn. 2.1. This gives the product $k_{test} \times k_{training}$. The term $n \times d$ comes from Eqn. 2.3. Therefore, we obtained the complexity for one iteration as the expression. The overall computational complexity is $C_n \times t$ where t is the number of iterations.

[1]Note that the optimal solution may not be unqiue.

2.3 Hierarchical Learning

From the analysis in previous section, we see that the computation at each iteration depends on several factors and the complexity is $O(n^2)$ in terms of n, the size of the data. For typical applications, n, which is the number of pixels in the image, is relative large. Thus the algorithm can be time consuming at each iteration. Also when n is large, the dimension of the searching space, which is the Grassmann manifold whose dimension is $d \times (n-d)$, is large. As the other factors in the computational complexity can not be avoided, here we propose a hierarchical learning process by decomposing the original optimization problem into several levels according to a specifically designed hierarchical organization and the dimension of the data is reduced at each level using a shrinkage matrix. The learning process starts from the lowest level with an arbitrary initial point. The following idea is then applied recursively: (i) optimize the recognition performance in the reduced space using the expanded learning result of optimal basis of the next lower level as an initial point, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show that the optimal performance obtained in the reduced space is maintained after the expansion. Therefore, the learning process of each level starts with a good initial point obtained from the next lower level. This speeds up the original algorithm significantly since the learning is performed mainly in reduced spaces and the computational complexity is reduced greatly at each iteration.

2.3.1 Speeding Up the Search

First we state the following setup that gives a specific way to shrink the dimension of the search space and expand the optimal basis obtained in the shrunk space to the original space with a nice property that the performance is maintained.

Basic learning setup: *If* $\hat{I}_1 = AI_1$, $\hat{I}_2 = AI_2$ *and* $U = A^T\hat{U}$, *in which* $I_1 \in \Re^{n_0}$, $I_2 \in \Re^{n_0}$, $\hat{I}_1 \in \Re^{n_1}$, $\hat{I}_2 \in \Re^{n_1}$, $A \in \Re^{n_1 \times n_0}$, $U \in \Re^{n_0 \times d}$, $\hat{U} \in \Re^{n_1 \times d}$, $d, n_0, n_1 \in Z^+$. *Then* $d(I_1, I_2; U) = d(\hat{I}_1, \hat{I}_2; \hat{U})$.

Proof: Using definitions of α (Eqn. 2.4) and $d(\cdot, \cdot; \cdot)$ (Eqn. 2.3), we get

$$
\begin{aligned}
d(I_1, I_2; U) &= \|\alpha(I_1, U) - \alpha(I_2, U)\| \\
&= \|U^T I_1 - U^T I_2\| \\
&= \|\hat{U}^T A I_1 - \hat{U}^T A I_2\| \\
&= \|\hat{U}^T \hat{I}_1 - \hat{U}^T \hat{I}_2\| \\
&= \|\alpha(\hat{I}_1, \hat{U}) - \alpha(\hat{I}_2, \hat{U})\| \\
&= d(\hat{I}_1, \hat{I}_2; \hat{U}).
\end{aligned} \tag{2.8}
$$

□

To simplify our narration in the future, we give the following terms. Let the size of the original image be n_0 and the size of the dimension reduced image be n_1. The matrix A in basic learning setup is called the *shrinkage matrix*, $m = \frac{n_0}{n_1}$ is called the *shrinkage factor*. Multiplying A^T with $\hat{U}$ to get U is called *basis expansion* $\hat{U}$ by shrinkage matrix A.

Hierarchical learning OCA setup: *Let F be the OCA performance function defined by Eqn. 2.1. If $\hat{I}_1 = AI_1$, $\hat{I}_2 = AI_2$ and $U = A^T\hat{U}$, in which $I_1 \in \Re^{n_0}$, $I_2 \in \Re^{n_0}$, $\hat{I}_1 \in \Re^{n_1}$, $\hat{I}_2 \in \Re^{n_1}$, $A \in \Re^{n_1 \times n_0}$,$AA^T = I$, $U \in \Re^{n_0 \times d}$, $\hat{U} \in \Re^{n_1 \times d}$, $d, n_0, n_1 \in Z^+$. Then $F(U) = F(\hat{U})$.*

Proof: By definition, F depends only on $d(\cdot, \cdot; \cdot)$, the distance between the representation of images. The conclusion of basic learning setup tells us that $d(I_1, I_2; U) = d(\hat{I}_1, \hat{I}_2; \hat{U})$. Therefore, $F(U) = F(\hat{U})$.

□

Therefore, by hierarchical learning OCA setup, to save time when searching for the optimal basis U of size $n_0 \times d$ on the Grassmann manifold $\mathcal{G}_{n_0,d}$ of dimension $d \times (n_0 - d)$ we can do the following three steps:

1. Shrink the image size to get dimension reduced training and test images of length n_1 by multiplying the shrinkage matrix A with original images.

2. Apply the OCA algorithm to search for the optimal basis $\bar{U}$ of size $n_1 \times d$ on the Grassmann manifold $\mathcal{G}_{n_1,d}$ of dimesion $d \times (n_1 - d)$ with dimension shrunk training and test images.

3. Expand the basis by equation $U = A^T\bar{U}$ to get the basis U of size $n_0 \times d$.

It's worthy to note the fact that the recognition rate using U is exactly the same as it is using $\bar{U}$.

Let's take a look at how much is gained by applying these steps. For each iteration, the computational complexity with images of size n_0 is $C_{n_0} = O(d \times (n_0 - d) \times k_{test} \times k_{training} \times n_0 \times d)$. While the computational complexity with images of size n_1 is

$$\begin{aligned} C_{n_1} &= O(d \times (\tfrac{n_0}{m} - d) \times k_{test} \times k_{training} \times \tfrac{n_0}{m} \times d) \\ &= \tfrac{n_0 - md}{m^2(n_0 - d)} C_{N_1} \\ &\approx \tfrac{1}{m^2} C_{n_0}, \end{aligned} \tag{2.9}$$

considering the fact $n_0 >> d$.

Obviously it is much more efficient to learn on $\mathcal{G}_{n_1,d}$ than on $\mathcal{G}_{n_0,d}$ when the dimension of search space is reduced from $d \times (n_0 - d)$ to $d \times (n_1 - d)$. With the nice property stated in hierarchical learning OCA setup, step 3 keeps the performance. Therefore, we get the basis U of size $n_0 \times d$ with performing the time saving learning process in a smaller space.

2.3.2 Hierarchical Learning Algorithm

Intuitively, the larger shrinkage factor m, the more computationally efficient in the search. However, the success of this procedure depends on whether the highest performance achievable in the reduced space $\mathcal{G}_{n_1,d}$ is acceptable or not. If the performance offered by the optimal basis $\bar{U}$ is not high enough, after expanding the basis, U also offers the same unsatisfied performance. To achieve high performance we have to search on $\mathcal{G}_{n_0,d}$ with an initial point U, which is potentially time consuming since the search space is large and the computational complexity of each iteration is high.

To overcome this problem, we propose to perform the search hierarchically at different levels. The basic idea is as follows. Instead of choosing large m to shrink the dimension only once, we can choose a relatively smaller shrinkage factor m_k to shrink the original images to get the training and test images at level k, where $k = 0, 1, ..., L$ and $m_0 < m_1 < ... < m_L$ where $m_0 \equiv 1$. Level 0 is called the highest level and level L is called the lowest level. The size of images at level k is $\frac{n_0}{m_k}$ where n_0 is the size of the original images. Our goal is to find an optimal basis of size $n_0 \times d$ at level 0. To fulfill this task, we do the search hierarchically

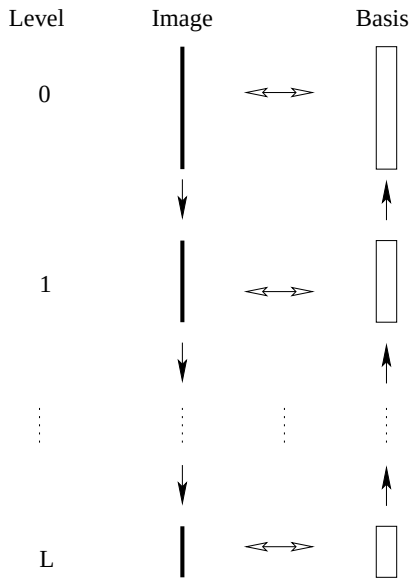


Figure 2.1. Hierarchical search process. Image size is reduced as level number increases. Firstly, the optimal basis U_L is obtained at level L as a learning result on $\mathcal{G}_{\frac{n_0}{m_L},d}$. We obtain a basis $\bar{U}_{L-1}$ of level $L-1$ by expanding U_L. Hierarchical learning OCA setup shows that $\bar{U}_{L-1}$ and U_L have the same performance. $\bar{U}_{L-1}$ is used as an initial point to perform the learning at level $L-1$ on $\mathcal{G}_{\frac{n_0}{m_{L-1}},d}$. The 'search – expand basis – search' process goes on till we get the optimal basis at level 0.

as follows. The search begins from level L on $\mathcal{G}_{\frac{n_0}{m_L},d}$ with dimension-reduced images of size $\frac{n_0}{m_L}$. The computational complexity at this level is $\frac{1}{m_L^2}C_{n_0}$ for each iteration. The search can be effectively done since the learning space $\mathcal{G}_{\frac{n_0}{m_L},d}$ is relatively small and the computational complexity of each iteration is low. After getting an optimal basis U_L at level L, we obtain a basis $\bar{U}_{L-1}$ of level $L-1$ by expanding U_L. Based on the above discussion we know that $\bar{U}_{L-1}$ and U_L have the same performance. If we use $\bar{U}_{L-1}$ as an initial point to perform the search at level $L-1$ on $\mathcal{G}_{\frac{n_0}{m_{L-1}},d}$ with images of size $\frac{n_0}{m_{L-1}}$, the search can be performed relatively effectively. The computational complexity at this level is $\frac{1}{m_{L-1}^2}C_{n_0}$ for each iteration. The search result of this level will be used to obtain a basis $\bar{U}_{L-2}$ of level $L-2$, which is used as an initial point for further search at level $L-2$. This process is repeated until we have reached level 0. In summary, the search is performed from the lowest level L to the highest level 0. The lower the level, the more efficient the search. The search result of the lower level

offers a good initial point for the next upper level. The recognition performance keeps on increasing at each level. We hope the acceptable recognition performance can be obtained at lower level. It's not then necessary to perform the time consuming search at upper levels and the optimal basis of level 0 is directly computed by expanding the basis from level 1.

This process is summarized and illustrated in Fig. 2.1. The hierarchical learning algorithm is given below.

Hierarchical Learning Algorithm: Suppose we decompose the learning process to $L+1$ levels, and the shrinkage factors are $m_1, ..., m_L$ with $m_0 \equiv 1, m_0 < m_1 < ... < m_L$. The original image size is n_0. Our aim is to find the optimal basis U_0 of level 0.

1. Choose the dimension shrinkage matrices A_k of size $\frac{n_0}{m_k} \times \frac{n_0}{m_{k-1}}$, for $k = 1, \ldots, L$. Next prepare new training and test images at each level i for $i = 1, \ldots, L$: shrinking the image dimension on the training and test images for i times by left-multilplying $\prod_{k=i}^{1} A_k$ with the original images.

2. Learn starting from level L for the optimal basis U_L at level L on $\mathcal{G}_{\frac{n_0}{m_L},d}$ with training and test images of size $\frac{n_0}{m_L}$.

3. For each $k = L-1, \ldots, 0$,

   BEGIN

   (a) let $\bar{U}_k = A_{k+1}^T U_{k+1}$,

   (b) using $\bar{U}_k$ as the initial point, search for the optimal basis U_k at level k on $\mathcal{G}_{\frac{n_0}{m_k},d}$ with training and test images of size $\frac{n_0}{m_k}$.

   END

2.3.3 Shrinking Dimension through Adaptive K-means

For the hierarchical search algorithm to be effective, the key is to keep the learning process to be performed mostly at lower levels and exempt the heavy computation at higher levels. This requires the best achievable performance in the higher level be preserved as high as

possible in the lower levels. In essence, this arises the question of how to reduce the dimension from the original images such that the performance obtained with the dimension reduced images is as high as possible. Since we reduce the image dimension by left-multiplying a shrinkage matrix, the above question may be restated as follows: how to choose a shrinkage matrix such that the performance of the dimension reduced image can be as high as possible? To answer the above question, ideally one would require a search over all shrinkage matrices to obtain the desired one. However, performing such a search may be more time consuming and complex than the original problem itself. Instead we propose an efficient way to reduce the dimension of images based on heuristics. The main idea is illustrated with the following example.

Example: Let

$$I_1 = (\ 10 \quad 30 \quad 10 \quad 80 \quad 30 \quad 80 \quad 10 \quad 80 \quad 30\ )^T,$$

$$I_2 = (\ 15 \quad 35 \quad 15 \quad 70 \quad 35 \quad 70 \quad 15 \quad 70 \quad 35\ )^T,$$

$$A_1 = \frac{1}{3}\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix},$$

$$A_2 = \frac{1}{3}\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

If we choose A_1 as a shrinkage matrix, we get

$$\hat{I}_1 = A_1 I_1 = (\ 10 \quad 30 \quad 80\ )^T,$$

$$\hat{I}_2 = A_1 I_2 = (\ 15 \quad 35 \quad 70\ )^T.$$

We can see that any performance achievable using I_1 and I_2 can be achieved using $\hat{I}_1$ and $\hat{I}_2$. Therefore A_1 is a good choice as a shrinkage matrix.

On the other hand, if we choose A_2 as a shrinkage matrix, we get

$$\tilde{I}_1 = A_2 I_1 = (\ 40 \quad 40 \quad 40\ )^T,$$

$$\tilde{I}_2 = A_2 I_2 = (\ 40 \quad 40 \quad 40\ )^T.$$

We can see that the reduced images $\tilde{I}_1$, $\tilde{I}_2$ do not provide any information for recognition any more, since $\tilde{I}_1$ and $\tilde{I}_2$ are identical while I_1 and I_2 are different. Therefore A_2 is a bad choice as a shrinkage matrix.

Let us do some analysis to see why A_1 performs well while A_2 does not. Let M be a matrix with I_1 and I_2 being its two columns.

$$\begin{aligned} \hat{M} &= A_1 M, \\ \tilde{M} &= A_2 M \end{aligned}$$

in which $\hat{M} = [\hat{I}_1 \quad \hat{I}_2]$,$\tilde{M} = [\tilde{I}_1 \quad \tilde{I}_2]$. Matrices A_1 and A_2 correspond to two different ways of grouping row vectors of M and represent them $\hat{M}$ and $\tilde{M}$ respectively. Matrix A_1 groups together rows of small distance and represent each cluster with its mean. However matrix A_2 groups rows of large distance and gives a bad result.

Based on the example, we argue that to keep the best achievable performance in the original space we should group pixels with similar values in the original image together and represent them in the dimension reduced images by their mean. To achieve this, we propose a pixel grouping algorithm called adaptive K-means, which is an adapted version of the commonly used K-means algorithm [14]. This method is used in the hierarchical learning algorithm to generate shrinkage matrix A_k implicitly. It is designed such that pixels that are clustered together contribute to the same coordinate in the reduced space.

Suppose the image sizes of n_l and n_{l+1} at level l and $l+1$ satisfy the relation $n_l = m n_{l+1}$, where m is the shrinkage factor. As all the training images should be shrunk in the same way, we put all of them in a matrix M of size $n_l \times n$ where each column of M is an image and n is the number of training images. We want to get a matrix $\bar{M}$ of size $n_{l+1} \times n$ where each row is the mean of m rows of M. Let $M = [M_1^T M_2^T ... M_{n_l}^T]^T$ where $M_i (i = 1, .., n_l)$ is the i - th row of M. Treating M_i as a point in $\Re^n$, we give the following algorithm to group the n_l points $M_i (i = 1, .., n_l)$ to n_{l+1} clusters $\overline{M}_i (i = 1, \ldots, n_{l+1})$.

Adaptive K-means Algorighm: Let $S = \{M_i | i = 1, \ldots, n_l\}$. Randomly choose n_{l+1} points $\overline{M}_i \in \Re^n, i = 1, \ldots, n_{l+1}$. Choose maximum iteration number T.

1. $j = 0$. For each $\overline{M}_i$, choose the nearest m points in S, group them into cluster $\overline{M}_i$. Remove the chosen points from S.

2. For each cluster with center $\overline{M}_i$, compute variance $\overline{V}_i$ and mean $(\overline{M}_i)_{new}$, set $\overline{M}_i = (\overline{M}_i)_{new}$.

3. Let $S = \{M_i | i = 1, \ldots, n_l\}$. Sorting according to $\overline{V}_i$, get list $L : \overline{V}_{i_1} \leq \overline{V}_{i_2} \leq \cdots \leq \overline{V}_{i_{n_{l+1}}}$, in which $(i_1, \cdots, i_{n_{l+1}})$ is a permutation of $(1, \cdots, n_{l+1})$.

4. According to L, for each cluster with center $\overline{M}_{i_k}$ we choose the nearest m points in S and put them into this cluster. Remove the chosen points from S.

5. For each cluster with center $\overline{M}_{i_k}$, compute variance $\overline{V}_{i_k}$ and mean $(\overline{M}_{i_k})_{new}$. If $(\overline{M}_{i_k})_{new} = \overline{M}_{i_k}$ for every cluster or $j > T$, stop, else $j = j + 1$, go to step 3.

2.3.4 Shrinking Dimension Through PCA

Besides the adaptive K-means algorithm, principal component analysis is another choice to shrink image dimension.

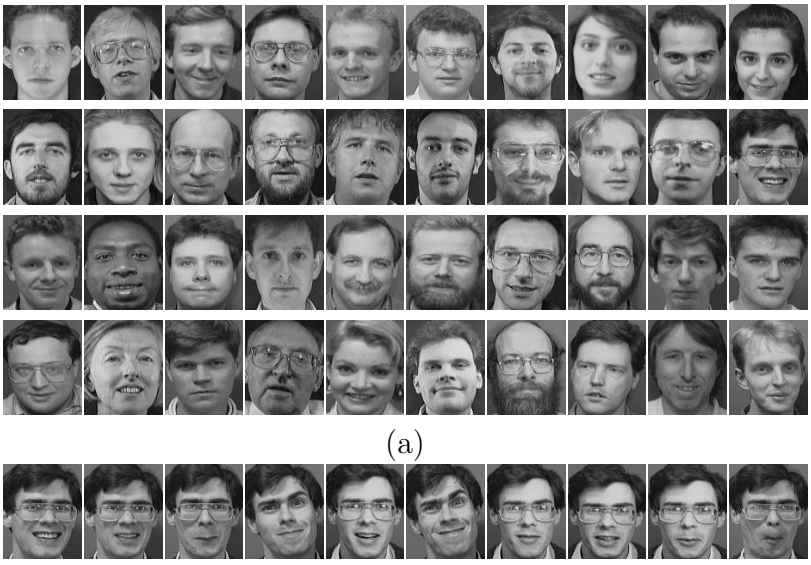
Considering the fact that for n images of size n_0 there are at most $p \triangleq min(n-1, n_0-1)$ PCA bases, as the rest of the eigenvalues of the covariance matrix are zero. The p bases span a subspace S of $\Re^n$. If we project the original images to S, the achievable recognition rate using the projected images should be the same as using the original images. This observation gives us the idea that we can use the matrix composed of PCA basis as a shrinkage matrix. The time used to compute the PCA basis is a part of the total time of the hierarchical learning. When p is relatively small, we prefer to use PCA to generate the shrinkage matrix, because the time spent on computing the PCA basis is relatively small by using the fast PCA algorithm. When p is relatively large, the time spent on computing PCA basis is relatively long too. In this case, we prefer to use the adaptive kmeans algorithm.

2.4 Experimental Results

Here we use the ORL database of faces[2] and part of the CMU PIE dataset [43]. The ORL dataset consists of faces of 40 different subjects with 10 images for each subject. The images

[2]http://www.uk.research.att.com/facedatabase.html

were taken at different times with different lighting conditions on a dark background. While only limited side movement and tilt were allowed in this dataset, there was no restriction on facial expression. All the subjects are shown in Fig. 2.2(a) and all the 10 images of a particular subject are shown in Fig. 2.2(b) to demonstrate the variations of facial expression and lighting condition. The PIE dataset we used consists of faces of 66 different subjects with 21 images each.

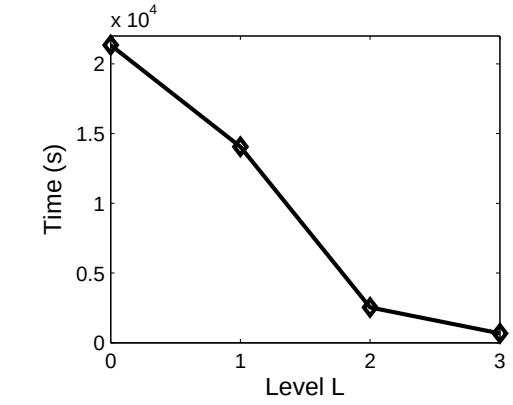


(a)

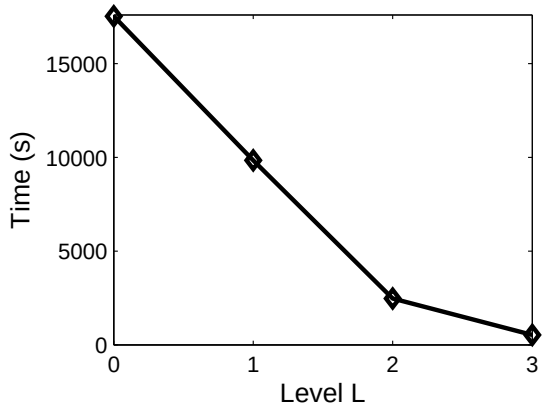
(b)

Figure 2.2. ORL face database. (a) 40 subjects in the database. (b) 10 face images of one subject taken at different facial expression and illumination conditions.

1. First, we have studied the time for searching for the optimal basis versus the total number of levels L. Figure 2.3 shows the experimental result, which highlights the effectiveness of hierarchical learning algorithm. For (a), when L is zero, i.e. learning without hierarchical algorithm, it takes 21341 seconds (6 hours) to get the optimal basis. However, after applying hierarchical learning algorithm, it only takes 675 seconds (11 minutes) to finish the search when L=3. The hierarchical algorithm speeds up the original one with a factor of 31. In this experiment, we shrink image size by adaptive K-means algorithm. The PIE dataset is used and the image size is 10000, image sizes



(a)



(b)

Figure 2.3. Plots of learning time versus the total number of levels L, shrinking with adaptive-kmeans. (a) Dataset: PIE, image size is 10000, reduced image sizes of level 1 throuth 3 are 2500, 625 and 157 respectively. $d = 10$, $k_{train} = 660$, and $k_{test} = 726$. (b) Dataset: ORL, image size is 2576, reduced image sizes of level 1 throuth 3 are 1288, 644 and 322 respectively. $d = 12$, $k_{train} = 120$, and $k_{test} = 160$.

of level 1 through 3 are 2500, 625 and 157 respectively, $d = 10$, $k_{train} = 660$, and $k_{test} = 726$. For (b), the ORL dataset is used and image size is 2576, image sizes of level 1 through 3 are 1288, 644 and 322. $d = 12$, $k_{train} = 120$, and $k_{test} = 160$. The result shows that the learning time is reduced from 17523 seconds (4.8 hours) to 527 seconds (8.8 minutes) when L=3. The speed up factor is 33.

2. Fig. 2.4 shows cases at different levels for the total number of levels $L = 3$. We plot the recognition rate versus iteration and the distance between X_t and X_0 versus t, in which X_t is the basis of the subspace at iteration t. The distance between any two subspaces U_1 and U_2 is computed as: $\|U_1U_1^T - U_2U_2^T\|$. We can see that it costs 600 iterations to achieve a performance near perfect at level 3 (a). Starting the search with the good initial point obtained at level 3 and searching in the relatively small space, it takes only 8 iterations at level 2 to achieve perfect performance (b). From (c) and (d) we see that without a computationally heavy search in a large space, it gets the optimal basis directly at level 1 (c) and level 0 (d), respectively, by expanding the optimal basis from the lower level. In this experiment we use the ORL dataset and shrink the image size by the adaptive kmeans algorithm. Original image size is chosen to be 2576, the reduced image sizes of level 1 through 3 are 644, 136 and 34 respectively. $d = 10$, $k_{train} = 5$, and $k_{test} = 5$.

3. When shrinking the image size by PCA, the proposed algorithm also shows its effectiveness in saving learning time. The experimental results showed by Fig. 2.5 once again convince the effectiveness of hierarchical learning algorithm.

   (a) The PIE dataset is used and original image size n is chosen to be 2500, the reduced image sizes of level 1 and level 2 are 1385 and 100 respectively. $d = 12$, $k_{train} = 660$, and $k_{test} = 726$.

   (b) The ORL dataset is used and original image size n is chosen to be 2576, the reduced image sizes of level 1 and level 2 are 279 and 100 respectively, $d = 12$, $k_{train} = 120$, and $k_{test} = 160$.

4. We also compared the two proposed shrink methods: adaptive kmeans and PCA. We found that shrinking the image size by the adaptive kmeans works better than PCA when the total number of training and test images is relatively large and shrink image size by PCA worked better when the number is relatively small. The reason for this phenomenon is that computing PCA is more time consuming than adaptive K-means when the subspace dimension is large. Fig. 2.6 shows the comparison between the two shrinking methods.

(a) The PIE dataset is used and original image size n is chosen to be 2500, the reduced image sizes of level 1 and level 2 are 1385 and 100 respectively, $d = 12$, $k_{train} = 660$ and $k_{test} = 726$. Here the solid line is the time using adaptive kmeans, dashed line PCA.

(b) The ORL dataset is used and original image size n is chosen to be 2576, the reduced image sizes of level 1 and level 2 are 279 and 100 respectively, $d = 12$, $k_{train} = 120$ and $k_{test} = 160$. Here the solid line is the time using adaptive kmeans, dashed line PCA.

2.5 Discussion

This chapter offers a new technique called hierarchical learning that can effectively reduce the learning time of one application of optimal component analysis, which can be treated as a learning example on the Grassmann manifold. Hierarchical learning decomposes the original optimization problem into several levels according to a specifically designed hierarchical organization and the dimension of the data is reduced at each level using a shrinkage matrix. The following idea is then applied recursively: (i) optimize the recognition performance in the reduced space using the expanded learning result of optimal basis of the next lower level as an initial point, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show that the optimal performance obtained in the reduced space is maintained after the expansion. Therefore, the learning process of each level starts with a good initial point obtained from the next lower level. This speeds up the original algorithm significantly since the learning is performed mainly in reduced spaces and the computational complexity is reduced greatly for each iteration. The experimental results show the effectiveness of the proposed technique.

To simplify our discussion, let $\mathcal{HL}$ denote a family of functions: $\mathcal{HL} = \{F|F(U)$ is a function, where U is an n -by- d matrix, $n, d \in Z^+$ and $F(\cdot)$ satisfies the condition: for any n' -by- d matrix $\bar{U}$ $(n' < n)$, there exists an n-by-n' matrix A such that $F(A\bar{U}) = F(\bar{U})$ $\}$. While the hierarchical learning technique is used here to speed up an application of OCA, the idea can be applied to a broad range of optimization problems on Grassmann and Stiefel manifolds. For example, the Newton and conjugate gradient algorithms on the Grassmann

and Stiefel manifolds proposed by Edelman et al. in [12] can be accelerated by hierarchical learning if the objective function is a member of the family of functions $\mathcal{HL}$.

Note the efficiency is gained by decomposing the learning process on a large Grassmann manifold to a number of hierarchically organized Grassmann manifolds with small dimensions, the effectiveness of the algorithm depends on the best achievable performance in the reduced dimension. While the proposed adaptive K-means algorithm is shown to be effective for the dataset we have used, its effectiveness is not guaranteed. The conditions under which the algorithm is effective need to be further investigated. Another future research issue is to find better pixel grouping algorithms to generate the shrinkage matrix.

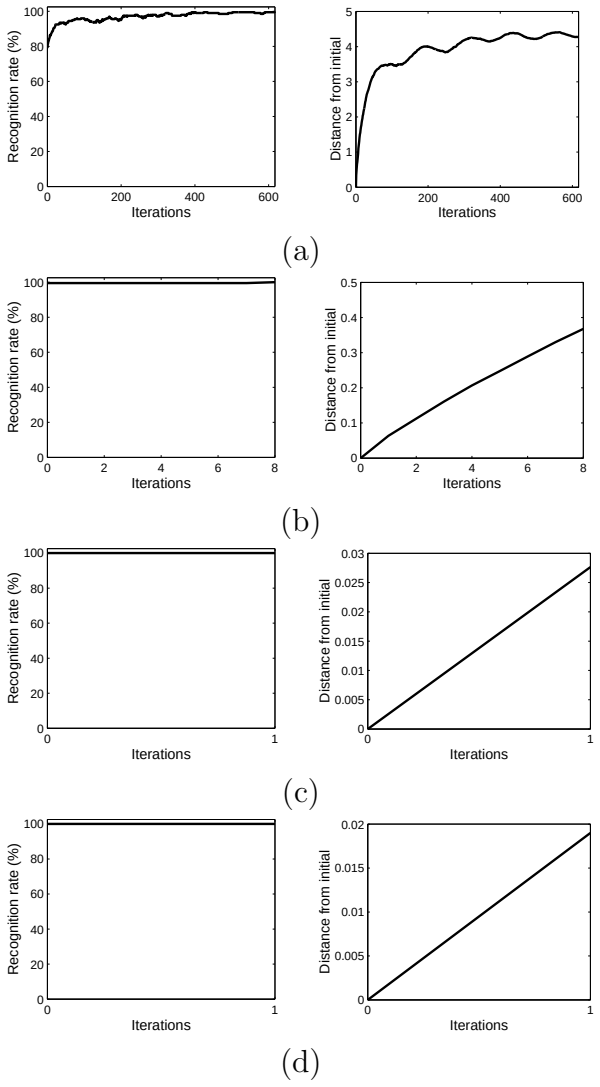
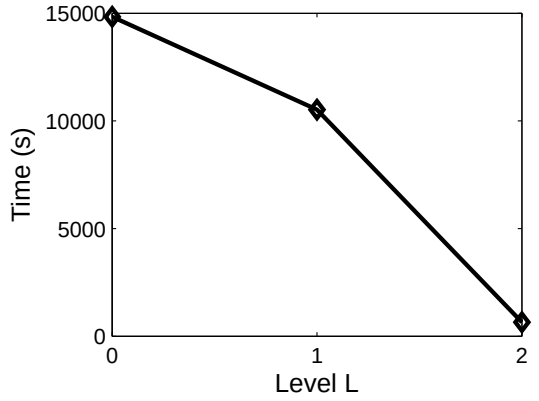
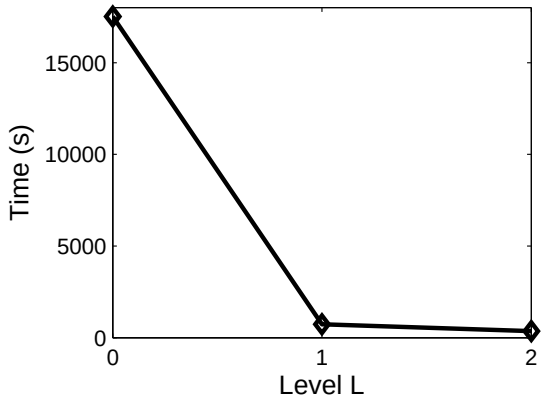


Figure 2.4. Plots of recognition rate $F(X_t)$ (left) and distance of X_t from X_0 (right) versus t for different level using hierarchical learning algorithm for $L = 3$. (a) level 3, (b) level 2, (c) level 1, (d) level 0. For these curves, image size is 2576, reduced image sizes of level 1 through 3 are 644, 136 and 34 respectively. $d = 10$, $k_{train} = 5$, and $k_{test} = 5$. Dataset: ORL.



(a)



(b)

Figure 2.5. Plot of searching time versus L, shrinking with PCA. (a) Dataset: PIE, image size is 2500, reduced image sizes of level 1 and level 2 are 1385 and 100 respectively, $d = 12$, $k_{train} = 660$, and $k_{test} = 726$. (b)Dataset: ORL, image size is 2576, reduced image sizes of level 1 and level 2 are 279 and 100 respectively, $d = 12$, $k_{train} = 120$, and $k_{test} = 160$.

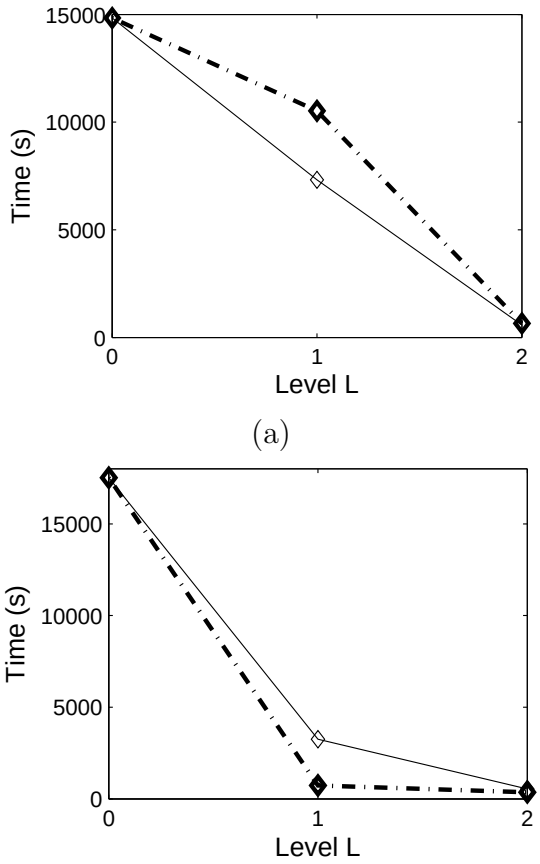


Figure 2.6. Comparison of the two shrinkage methods: adaptive K-means and PCA. Plot of searching time versus L. (a) Dataset: PIE, image size is 2500, reduced image sizes of level 1 and level 2 are 1385 and 100 respectively, $d = 12$, $k_{train} = 660$, and $k_{test} = 726$. Here the solid line is the time using adaptive kmeans, dashed line PCA. (b)Dataset: ORL, image size is 2576, reduced image sizes of level 1 and level 2 are 279 and 100 respectively, $d = 12$, $k_{train} = 120$, and $k_{test} = 160$. Here the solid line is the time using adaptive kmeans, dashed line PCA.

CHAPTER 3

CONCLUSION

In this thesis, two techniques are proposed. Part one is Appearance-Based Classification and Recognition Using Spectral Histogram Representations. In this part, we presented spectral histogram representations for classification and recognition of images derived from a generative process. While the derived model is simple, it is effective for different appearance-based applications that have been primarily studied separately. The marked improvement of our method over existing ones, along with the image synthesis and clustering visualization results, justifies the effectiveness and generality of the spectral histogram representation. Not only is our approach generic as demonstrated through different datasets of real images, the representation also provides other advantages such as illumination, rotation, and scale invariance by choosing proper filters. Our representation along with the filter selection algorithm provides a unified framework for appearance-based object recognition and image classification. Within this framework, the difference among general object recognition, face recognition, and texture classification is the choice of most effective filters. While filters with large scales are most effective for face recognition as faces are topographically very similar, filters whose scales are comparable with texture elements are most effective for texture classification. Our filter selection algorithm chooses the most effective set of filters in this regard. This may lead to a system that is effective for different types of images, which is a key requirement for a generic recognition system.

Part two is Hierarchical Learning for Optimal Component Analysis. In this part, we offer a new technique called hierarchical learning that can effectively reduce the learning time of one application of optimal component analysis, which can be treated as a learning example on the Grassmann manifold. Hierarchical learning decomposes the original optimization problem into several levels according to a specifically designed hierarchical organization and the dimension of the data is reduced at each level using a shrinkage matrix. The following

idea is then applied recursively: (i) optimize the recognition performance in the reduced space using the expanded learning result of optimal basis of the next lower level as an initial point, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show that the optimal performance obtained in the reduced space is maintained after the expansion. Therefore, the learning process of each level starts with a good initial point obtained from the next lower level. This speeds up the original algorithm significantly since the learning is performed mainly in reduced spaces and the computational complexity is reduced greatly for each iteration. The experimental results show the effectiveness of the proposed technique.

REFERENCES

[1] R. Azencott, J. P. Wang, and L. Younes, "Texture classification using windowed Fourier filters," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 19(2), pp. 148-153, 1997.

[2] D. Abbott, ed., "The Biographical Dictionary of Scientists; Mathematicians, " P. Bedrick Books, New York, 1986.

[3] S. Amari, "Natural Gradient Works Efficiently in Learning," *Neural Computation*, vol. 10, pp. 251-276, 1998.

R. Azencott, J. P. Wang, and L. Younes, "Texture classification using windowed Fourier filters," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 19(2), pp. 148-153, 1997.

[4] A. Bell and T. Sejnowski, "The Independent Components of natural scenes are edge filters," *Vision Research*, vol. 37, no. 23, pp. 3327–3338, 1997.

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

[6] J. R. Bergen and E.H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, pp. 363–367, 1988.

[7] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[8] W. M. Boothby, *An Introduction to Differential Manifolds and Riemannian Geometry*, Academic Press, 1986.

[9] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and machine recognition of faces: a survey," *Proceedings of IEEE*, vol. 83, no. 5, pp. 705–740, 1995.

[10] J. M. Coggins and A. K. Jain, "A spatial filtering approach to texture analysis," *Pattern Recognition Letters*, vol. 3, no. 3, pp. 195–203, 1985.

[11] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd Ed., John Wiley & Sons, 2001.

[12] A. Edelman, T. Arias, and S. T. Smith, "The Geometry of Algorithms with Orthogonality Constraints," *SIAM J. Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303-353, 1998.

[13] S. Fiori, "A Theory for Learning by Weight Flow on Stiefel-Grassmann Manifold," *Neural Computation*, vol. 13, pp. 1625–1647, 2001.

[14] D. A. Forsyth and J. Ponce, *Computer Vision, A Modern Approach*. pp. 315-317. Prentice Hall, 2003.

[15] J. M. Francos, A. Z. Meiri, and B. Porat, "A unified texture model based on a 2-D Wold-like decomposition," *IEEE Transactions on Signal Processing*, vol. 41, no. 8, pp. 2665–2678, 1993.

[16] D. Gabor, "Theory of communication," *Journal of IEE (London)*, vol. 93, pp. 429-457, 1946.

[17] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, pp. 1-58, 1992.

[18] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721-741, 1984.

[19] U. Grenander, *General Pattern Theory*, Clarendon Press, Oxford, 1993.

[20] H. G. Grassmann, *Die Ausdehnungslehre,* Enslin, Berlin, 1862.

[21] U. Grenander and A. Srivastava, "Probability models for clutter in natural images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(4), pp. 424-429, 2001.

[22] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference and prediction*, Springer-Verlag, New York, 2001.

[23] D. J. Heeger and J. R. Bergen, "Pyramid-based texture analysis/synthesis," in *Proceedings of SIGGRAPHS*, pp. 229–238, 1995.

[24] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Reading, MA, 1991.

[25] A. Hyvärinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, pp. 411–430, 2000.

[26] A. Hyvarinen. "Survey on Independent component analysis," *Neural Computing Surveys*, vol. 2, pp. 94–128, 1999.

[27] A. Hyvarinen. "Fast and robust fixed-point algorithm for independent component analysis," *IEEE Transactions on Neural Networks*, vol. 10, pp. 626–634, 1999.

[28] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[29] U. H.-G. Kreßel, "Pairwise classification and support vector machines," In *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola (eds.), pp. 255-268, The MIT Press, Cambridge, MA, 2002.

[30] N. Kruger, "Learning object representations using a priori constraints within ORASSYLL," *Neural Computation*, vol. 13, pp. 389-410, 2001.

[31] M. Lades, J. C. Vorbruggen, J. Buhmann, J. Lange, C. von de Malsburg, R. P. Wurtz, and W. Konen, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Transactions on Computers*, vol. 42, pp. 300-311, 1993.

[32] F. Liu and R. W. Ricard, "Periodicity, directionality, and randomness: Wold features for image modeling and retrieval," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 18, no. 7, pp. 722–733, 1996.

[33] X. Liu and L. Cheng. "Independent spectral representations of images for recognition," *Journal of the Optical Society of America, A*, vol. 20, no. 7, pp. 1271–1282, 2003.

[34] X. Liu, A. Srivastava, and K. Gallivan, "Optimal Linear Representations of Images for Object Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 662-666, 2004.

[35] X. Liu, A. Srivastava, and D. L. Wang, "Intrinsic generalization analysis of low dimensional representations," *Neural Networks*, vol. 16, no. 5/6, pp. 537–545, 2003.

[36] X. Liu and D. L. Wang, "A spectral histogram model for texton modeling and texture discrimination," *Vision Research*, vol. 42, no. 23, pp. 2617–2634, 2002.

[37] X. Liu and D. L. Wang, "'Texture classification using spectral histograms," *IEEE Transactions on Image Processing*, vol. 12, no. 6, pp. 661–670, 2003.

[38] Y. Ma, J. Kosecka, and S. Sastry, "Optimization Criteria and Geometric Algorithms for Motion and Structure Estimation," *Int'l J. Computer Vision*, vol. 44, no. 3, pp. 219–249, 2001.

[39] D. Marr, *Vision*, W. H. Freeman and Company, New York, 1982.

[40] A. M. Martinez and A. C. Kak, "PCA versus LDA, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(2), pp. 228–233, 2001.

[41] S. K. Murase and S. K. Nayar, "Visual learning and recognition of 3-d objects from appearance," *International Journal of Computer Vision*, vol. 14, pp. 5-24, 1995.

[42] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607-609, 1996.

[43] T. Sim, S. Baker, and M. Bsat, "The CMU Pose, Illumination, and Expression Database, " *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 12, pp. 1615–1618, Dec. 2003.

[44] M. Pontil and A. Verri, "Support vector machines for 3D object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20(6), pp. 637-646, 1998.

[45] T. Randen and J. H. Husoy, "Filtering for texture classification: A comparative study," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 21(4), pp. 291–310, 1999.

[46] F. S. Samaria and A. C. Harter, "Parameterization of a stochastic model for human face identification," In *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pp. 138–142, 1994.

[47] B. Schiele and J. L. Crowley, "Recognition without correspondence using multidimensional receptive field histograms," *International Journal of Computer Vision*, vol. 36, pp. 31–50, 2000.

[48] S. T. Smith, "Geometric Optimization Methods for Adaptive Filtering," Ph.D. thesis, Harvard University, Cambridge, MA, 1993.

[49] A. Srivastava, A. B. Lee, E. P. Simoncelli and S. C. Zhu. "On advances in statistical modeling of natural images, " *Journal of Mathematical Imaging and Vision*, vol. 18(1), 2003.

[50] A. Srivastava, "A Bayesian Approach to Geometric Subspace Estimation," *IEEE Trans. Signal Processing*, vol. 48, no. 5, pp. 1390–1400, 2000.

[51] A. Srivastava and X. Liu, "Statistical hypothesis pruning for identifying faces from infrared images," *Journal of Image and Vision Computing*, vol. 21, no. 7, pp. 651–660, 2003.

[52] E. Stiefel, "Richtungsfelder und fernparallelismus in n-dimensionalem mannig faltigkeiten," *Commentarii Math. Helvetici*, 8 (1935-1936), pp. 305–353.

[53] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, 11-32, 1991.

[54] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, pp. 71-86, 1991.

[55] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed., Springer-Verlag, New York, 2000.

[56] M. H. Yang, D. Roth, and N. Ahuja, "Learning to recognize 3D objects with SNoW," in *Proceedings of the Sixth European Conference on Computer Vision*, vol. 1, pp. 439-454, 2000.

[57] S. C. Zhu, X. Liu, and Y. Wu, "Exploring Texture ensembles by efficient Markov chain Monte Carlo," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 22, pp. 554-569, 2000.

[58] S. C. Zhu, Y. N. Wu, and D. Mumford, "Minimax entropy principle and its application to texture modeling," *Neural Computation*, vol. 9, pp. 1627–1660, 1997.

BIOGRAPHICAL SKETCH

Qiang Zhang

Qiang Zhang was born in Beijing, China in April, 1975. He received his B.S. in Pure Mathematics from Beijing Normal University, China in 1998. He received his M.S. in Applied Mathematics (Fuzzy and Artificial Intelligence) from Beijing Normal University, China in 2001 and M.S. in Computer Science in Spring 2005 from The Florida State University, U.S.A..