THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

A STUDY OF IMAGE REPRESENTATIONS FOR CONTENT-BASED IMAGE RETRIEVAL

By

DONGHU SUN

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, 2004

The members of the Committee approve the thesis of Donghu Sun defended on April 5, 2004.

Xiuwen Liu
Professor Directing Thesis

Anuj Srivastava
Committee Member

Daniel Schwartz
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 Father, Mother, Yajing, Xiaobo and Yi ...

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincerest appreciation to my adviser Prof. Xiuwen Liu. He offered me a research assistantship in Florida State Vision Group without which it would be impossible for me to even start my study here at FSU. His patient guidance and stimulating suggestions helped me so much throughout the past three years, in research, in the classes he taught and in the planning of my future career. His nice and respectful personality has built a great role model and I will definitely benefit a lot from it in my future life.

I would also like to thank Prof. Anuj Srivastava in Statistic Department at FSU and Prof. Deliang Wang in Ohio State University on whose research work this thesis depends heavily. My deepest thanks also extend to my committee member Prof. Daniel Schwartz. I learned many technologies about J2EE in the Java class he taught.

I would like to thank my wife Yi for her greatest love. She was always there to cheer me up when I felt low. Without her encouragement and support I would not finish my study here. I would like to thank my parents Qitai Sun and Lanzhen Zhang, my sisters Yajing and Xiaobo for their understanding and support.

I would like to thank my groupmates Christopher Warning, Haitao Wu, Lei Cheng and Qiang Zhang for their kindly help when I was in trouble. I won't forget the happy time we have had together.

My thanks also go to all the professors that taught me during my graduate study in FSU. I have learned a lot from their classes and this built a solid foundation for my future career.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The performance of a content-based image retrieval system depends on the representation of images. As a typical image consists of different objects, an image segmentation is needed for more accurate representations of contents. The first part of this thesis describes a generic image segmentation algorithm based on local spectral histograms of images. This algorithm, demonstrated by experimental results, is shown to be effective for both texture and non-texture images, and comparable to other segmentation algorithms. Due to the time constraint of an image retrieval system, the second part of this thesis focuses on low dimensional representations of images. By analyzing the semantics of commonly used linear subspace representations through sampling their intrinsic generalizations, their limitations are illustrated and a nonlinear representation, called Spectral Subspace Analysis (SSA) that overcomes these limitations is proposed. In addition, to obtain optimal retrieval performance, an algorithm for learning optimal representations is developed by formulating the problem as an optimization one on a Grassmann manifold and exploiting the underlying geometry of the manifold. Experimental results on different datasets show that both the SSA representation and the learned optimal representations can improve retrieval performance significantly for content-based image retrieval systems.

CHAPTER 1

INTRODUCTION

In recent years, with the advances in imaging technology, digital images are available at an increasing speed, resulting in large collections of images. Searching in these collections becomes more and more important in many fields, such as commerce, medical and government applications. This motivates research for content-based image retrieval and various application systems have been developed. A content-based image retrieval system is to search through an image database to find out images that are similar in content to a given query one. However, most existing retrieval systems, if not all, suffer one problem: when a user submits a query, the system retrieves images far from the user's expectation, leading to low retrieval performance. Santini and Jain [31] report that many existing systems give meaningless responses; Lesk [18] summarizes a generic phenomenon in retrieval systems using "What you see is what you get, but not what you want" as the title. Smeulders et al. [35] attribute it to a problem known as the semantic gap.

In this thesis we argue that the root of meaningless responses is the lacking of semantically meaningful representations of images. An ideal representation of images should be semantically meaningful so that it can group together content-similar images in the image space and is semantically discriminating for content-dissimilar images. This necessitates investigating the semantics of representations of images.

As images' contents, or objects, are of significant importance, it would semantically be advantageous to do a segmentation first. There are numerous algorithms for image segmentation (see [33] for a recent literature review). For gray level images with piecewise smooth regions, the Mumford-Shah model [28] is representative in that most criterion used in existing segmentation algorithms are its special cases [26]. In this thesis, we present a segmentation algorithm by extending the Mumford-Shah model to images consisting of piecewise smooth regions as well as texture ones using local spectral histograms [19, 22]

as a generic representation. Because a local spectral histogram of an image window consists of histograms of response images of chosen filters, it captures local spatial patterns through filtering and global patterns through histograms. Assuming that a representative spectral histogram is available for each region in an image, the segmentation algorithm is implemented as follows: 1) estimating a probability model for each region and classifying the image windows to obtain an initial segmentation, 2) iteratively updating pixels along region boundaries based on the derived probability models, and 3) further localizing region boundaries using refined probability models derived based on spatial patterns in segmented regions. Additionally, this algorithm addresses the issues of automatically identifying regions, selecting optimal filters for spectral histograms, and choosing an optimal window size for segmentation. This segmentation algorithm is effective for both texture and non-texture regions, justified by experimental results on different kinds of images. Also, it leads to more accurate segmentation results, which is justified by comparison with normalized cut method [34] and other methods [29]. As an image representation for content-based image retrieval, new representations need to be employed to represent the segmented objects. This is not included in this thesis and will be studied further in the future.

Due to the time constraint of a retrieval system, low dimensional representations are used to reduce the time for computing and sorting. Low dimensional representations impose equivalence relations in image space. Ideally, in the context of content-based image retrieval, only images with semantically similar contents should be grouped into an equivalence class. This semantics-related equivalence class is named as *intrinsic generalization*. For example, the histogram of pixel values of an image is widely used. This histogram representation is semantically too weak as different kind of images tend to be grouped in one equivalence class, which leads to meaningless responses. In this thesis, we study the equivalence class structures of low dimensional representations by sampling their intrinsic generalizations. To demonstrate the effectiveness of the sampling method, we utilize it to compare two low dimensional representation families, namely linear subspaces of images and spectral subspaces. The linear representations include principal component analysis (PCA) [16], independent component analysis (ICA) [9] and Fisher discriminant analysis (FDA) [11]. The study shows that the linear representations of images group semantically dissimilar images in one equivalence class and are not semantically meaningful for content-base image retrieval.

By analyzing two problems with linear subspace representations, we propose a nonlinear representation called Spectral Space Analysis which improves the intrinsic generalization. The proposed representation has been applied to a large dataset and improved retrieval performance has been obtained.

Additionally, we further present an algorithm that can be used to improve retrieval performance by explicitly finding optimal representations in both image and spectral spaces for content-based image retrieval applications. While it does not solve the semantic gap problem, it offers a method to reduce the semantic gap through labeled training images. The key to the proposed algorithm is to formulate the problem on Grassmann manifold and utilize an effective optimization algorithm, MCMC simulated annealing, on the manifold. The experimental results on different datasets demonstrate the feasibility and effectiveness of the proposed method.

The remainder of this thesis is organized as follows. Chapter 2 introduces an image segmentation algorithm using local spectral histograms. Chapter 3 analyzes the semantics of representations for content-based image retrieval. Chapter 4 presents an algorithm for learning optimal representations for image retrieval applications. Chapter 5 makes a brief conclusion for this thesis.

CHAPTER 2

IMAGE SEGMENTATION USING LOCAL SPECTRAL HISTOGRAMS

In this chapter, a generic segmentation algorithm by extending Mumford and Shah's model[28] using local spectral histogram representation is presented. Algorithms for boundary localization, automated seed selection, automated filter and integration scale selection are also provided to improve segmentation performance. Experimental results and comparison with other algorithms show that the proposed segmentation algorithm is effective and comparable to the best available.

2.1 Local Spectral Histogram Representation

Given an input image window $\mathbf{W}$ and a chosen bank of filters $\{F^{(\alpha)}, \alpha = 1, 2, \ldots, K\}$, for each filter $F^{(\alpha)}$, we compute a sub-band image $\mathbf{W}^{(\alpha)}$ through a linear convolution, i.e., $\mathbf{W}^{(\alpha)}(v) = F^{(\alpha)} * \mathbf{W}(v) = \sum_u F^{(\alpha)}(u)\mathbf{W}(v-u)$, where circular boundary condition is used for convenience. For $\mathbf{W}^{(\alpha)}$, we define the marginal distribution, or histogram

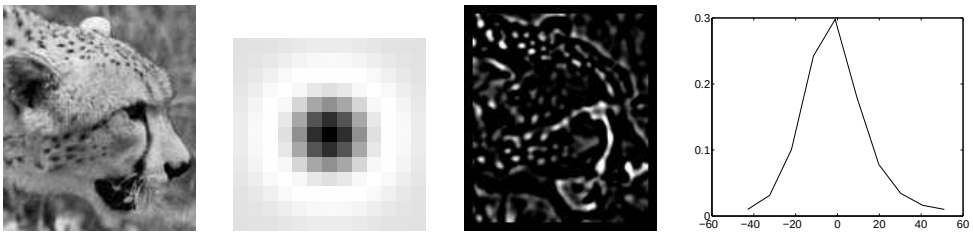
$$H_W^{(\alpha)}(z) = \frac{1}{|W|} \sum_v \delta(z - W^{(\alpha)}(v)). \tag{2.1}$$

Figure 2.1 shows an example of computing histogram. Figure 2.1(a) shows an input image. Figure 2.1(c) shows the image after linear convolution with a Laplacian of Gaussian filter shown in Figure 2.1(b). Figure 2.1(d) shows the histogram of the filtered image.

Then we define the spectral histogram with respect to the chosen filters as

$$H_W = (H_W^{(1)}, H_W^{(2)}, \ldots, H_W^{(K)}). \tag{2.2}$$

Figure 2.2 shows the spectral histograms of two images. For each image in Figure 2.2(a), it is first convoluted by each of the eight filters shown in Figure 2.2(c), then the corresponding



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

Figure 2.1. An example of computing histogram. (a) An input image. (b) A Laplacian of Gaussian filter. (c) The filtered image. (d) The histogram of filtered image.

histograms are computed. The spectral histogram is obtained by concatenating the eight histograms. The spectral histogram of an image or an image patch is essentially a vector consisting of marginal distributions of filter responses. The size of the input image or the input image patch is called *integration scale*. Because the marginal distribution of each filter response is a distribution, a similarity measure can be defined as χ^2-statistic, which is a first-order approximation of Kullback-Leibler divergence and is used widely to compare two histograms H_{W_1} and H_{W_2}

$$\chi^2(H_{W_1}, H_{W_2}) = \sum_{\alpha=1}^{K} \sum_{z} \frac{(H_{W_1}^{(\alpha)}(z) - H_{W_2}^{(\alpha)}(z))^2}{H_{W_1}^{(\alpha)}(z) + H_{W_2}^{(\alpha)}(z)}. \tag{2.3}$$

The spectral histogram provides a normalized feature statistic to compare image windows of different sizes. The input image windows do not need to be aligned; misalignment is a serious problem for approaches that use filter responses directly as features, such as those studied in [29], due to the inhomogeneity of filter responses. In this chapter, unless otherwise specified, eight fixed filters are used: the intensity filter, two gradient filters, LoG with two scales and three Gabor filters with different orientations. An automatic filter selection algorithm is introduced later in this chapter. When proper filters are chosen, the spectral histogram is sufficient in characterizing texture appearance. Figure 2.3 shows three types of images, where the typical images are generated using a Gibbs sampler [39]. In Figure 2.3(a), the spectral histogram captures the perceptual appearance of both regions. Given that the circular boundary is used for a typical image, the typical image represents closely the observed one. Figure 2.3(b) shows a texton image, where the spectral histogram captures

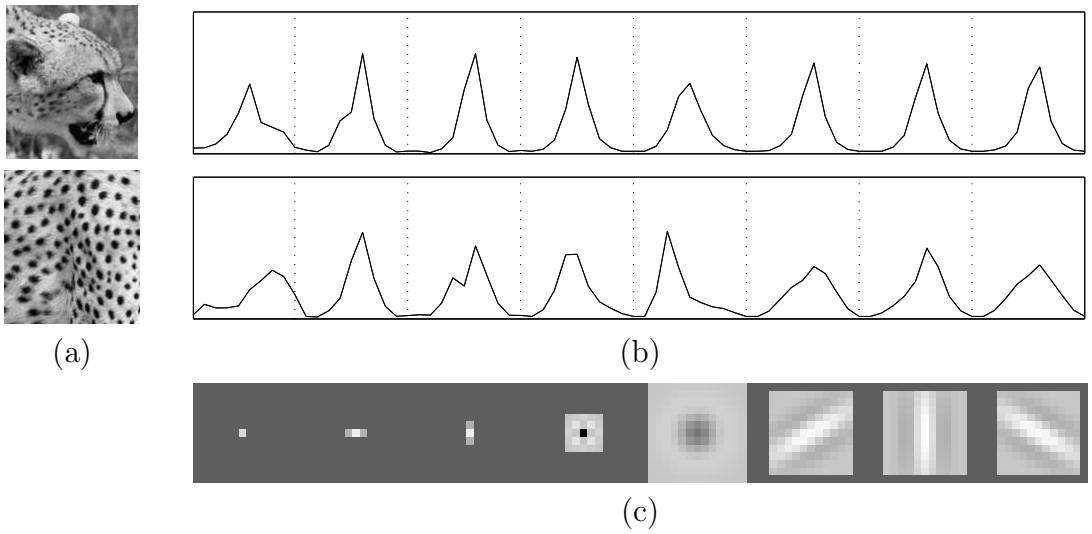


Figure 2.2. An example of spectral histograms of two images. (a) The input images. (b) The corresponding spectral histograms. (c) The used eight filters .

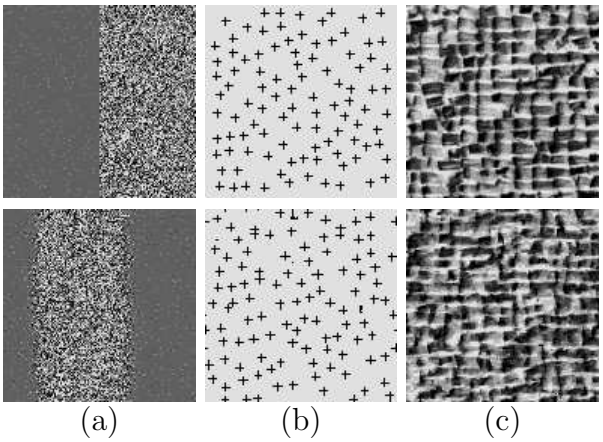


Figure 2.3. Different types of images characterized by spectral histograms. Top row shows an observed images and the bottom row a typical image that shares the same spectral histogram. (a) A gray-level image consisting of two piecewise constant regions with additive Gaussian noise. (b) A texton image consisting of cross elements. (c) A stochastic texture image.

the texton elements and the element density. This example demonstrates clearly that the spectral histogram provides a description for textons without specifying texton elements explicitly [22]. Figure 2.3(c) shows a stochastic texture and the spectral histogram captures the perceptual appearance well.

Note that the spectral histogram is defined on any type of images. Piecewise constant images with additive Gaussian noise are a special case where the spectral histogram has a unique pattern. In addition, it can also characterize patterns with topological structures (e.g. a face) with appropriate boundary conditions [20]. The spectral histogram representation was first suggested in psychophysical studies on texture modeling [4], and has been used in texture modeling and synthesis [15, 40, 19], texture classification [1, 19, 23] and modeling human texture discrimination [22]. Histograms of local fields have also been used for object recognition [32, 20].

2.2 Segmentation Algorithm

In [28], Mumford and Shah define a representative energy functional for segmentation of which segmentation criteria in most existing segmentation algorithms are special cases [26]. In their formulation, segmentation is posted as the problem of finding optimal partitions where each region should be as homogeneous as possible. The homogeneity of a region is measured by its variance; this, however, is not effective for a textured region. To overcome this limitation, we use local spectral histograms as local features and measure homogeneity of a region using distance among spectral histograms. As spectral histograms can characterize effectively both texture and non-texture regions, this formulation is effective for both types of regions.

To be more specific, let R be a grid defined on a planar domain and R_i, $i = 1, \cdots, n$ be a disjoint subset of R, Γ_i be the piecewise smooth boundary of R_i, and R be the union of R_i and Γ of Γ_i, $i = 1, \cdots, n$. A feature $\mathcal{F}_i$ is associated with each region R_i, $i = 1, \cdots, n$. We also define R_0, which is called *background* [38], as $R_0 = R - (R_1 \cup \cdots R_n)$. Note that segmentation can be represented using either Γ or R. Using the latter, motivated by the Mumford and Shah's energy functional, for an input image I, we define an energy functional for segmentation as

$$E(R,n) = \lambda_R \sum_{i=1}^{n} \sum_{(x,y)\in R_i} D(\mathcal{F}_{R_i}(x,y), \mathcal{F}_i) + \lambda_{\mathcal{L}} \sum_{i=1}^{n} |\Gamma_i| - \lambda_{\mathcal{F}} \sum_{i=1}^{n} \sum_{j=1}^{n} D(\mathcal{F}_i, \mathcal{F}_j) - \sum_{i=1}^{n} |R_i| \tag{2.4}$$

Here D is a distance measure between a feature at a pixel location and the feature vector of the region, λ_R, $\lambda_{\mathcal{F}}$, and $\lambda_{\mathcal{L}}$ are weights that control the relative contribution of the corresponding term. $\mathcal{F}_{R_i}(x,y)$ is the feature vector at pixel location (x,y), and this notation implies that the feature vector depends on R_i in our implementation. In this chapter, spectral histograms are used as features vectors, i.e., $\mathcal{F}_{R_i}(x,y) = H_{W^{(s)}(x,y)}$, where $W^{(s)}(x,y)$ is a local neighborhood, the size and shape of which are given by integration scale $W^{(s)}$ for segmentation, a predefined neighborhood. In (2.4), the first term encodes the homogeneity requirement in each region R_i and the second term requires that boundaries of regions be as short as possible, or as smooth as possible. The third term requires that the features of different regions be as different as possible. The last term is motivated by the fact that some regions may not be described well by the selected features. In that case, those regions should be treated as background, which can be viewed as grouping through inhomogeneity [38].

We use an iterative but deterministic algorithm to minimize the energy functional given in (2.4), and implement it as two stages followed an additional localization stage given in the next section. The first term in (2.4) is first minimized using the minimum distance classifier in the first stage and further refined by (2.5) along with the second term. The third term is used in the feature selection procedure described in Section 2.4.1 so that selected features will be different from each other. The last term is incorporated in (2.6). Feature vectors $\mathcal{F}_i$ for regions are first extracted from windows centered at given or detected pixel locations; the size of a window is specified by integration scale $W^{(s)}$ for segmentation. To estimate probability models and parameters T_i (homogeneity thresholds for regions used in (2.6) below), we compute the spectral histogram centered at a pixel location; to save computation, it is done only at sub-sampled pixels. The χ^2-statistic distance measure may not provide an accurate measure close to boundaries due to inhomogeneity. For example, in the image shown in Fig. 2.4, the left region is homogeneous and the variation allowed should be small. In the right region, the variation allowed should be relatively large. To overcome this problem and provide a more accurate model, we estimate a probability model of the χ^2-statistic for each given feature vector $\mathcal{F}_i$. This is done by computing the histogram of the χ^2-statistic between the computed spectral histograms at all chosen pixels and the given

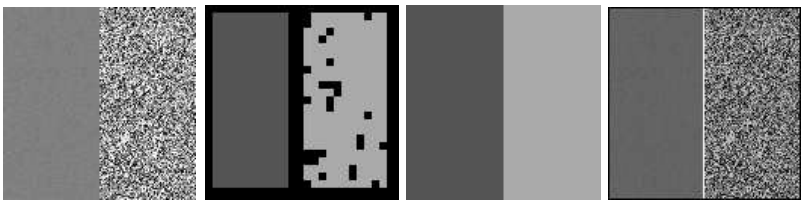
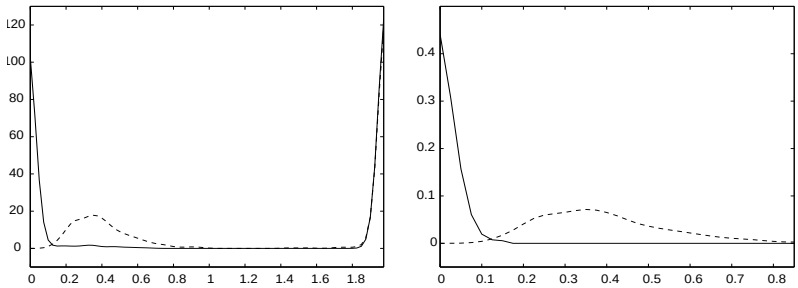


Figure 2.4. Gray-level image segmentation using spectral histograms. The integration scale $W^{(s)}$ for spectral histograms is a 15×15 square window, $\lambda_\Gamma = 0.2$, and $\lambda_B = 3$. Two features are given at $(32, 64)$ and $(96, 64)$. (a) A synthetic image with size 128×128. The image is generated by adding zero-mean Gaussian noise with different σ's at left and right regions. (b) Initial classification result. (c) Segmentation result. Each region is represented by a manually assigned gray value. All the pixels are perfectly segmented. (d) The segmentation result shown by region boundary (white). Note the original image is dimmed to make the boundary more visible.

spectral histogram $\mathcal{F}_i$ for the region. Figure 2.5(a) shows the histograms of the χ^2-statistic for the two regions in Fig. 2.4(a). Parameter T_i for each region is determined by the first trough after the leftmost peak from its histogram. Based on an assumption that feature vectors $\mathcal{F}_i$ are close to the true feature vectors, a probability model is derived by assigning zero probability for values larger than T_i. The derived probability models are given in Fig. 2.5(b). Then the input image is classified using a minimum distance classifier to minimize the first term in (2.4); the pixels whose minimum distance is larger than T_i, are classified as background. The classification result is used as initial segmentation. For the image in Fig. 2.4(a), the corresponding initial segmentation result is shown in Fig. 2.4(b). Note that the classification is done at sub-sampled pixels only and nearby pixels are assigned the same label.

To illustrate the effectiveness of the derived probability models and asymmetric windows, Fig. 2.6(a) shows a row from the image shown in Fig. 2.4(a). Figure 2.6(b) shows the probability of the two labels at each pixel using asymmetric windows. It can be seen that the edge point is localized precisely at the true location. Figure 2.6(c) shows the probability using windows centered at pixels. There is an interval where labels can not be decided because the spectral histogram computed in the interval does not belong to either of the regions. This demonstrates also that the probability models based on spectral histograms are sensitive to spatial patterns. For comparison, Fig. 2.6(d) shows the result using χ^2-statistic



(a) (b)

Figure 2.5. The histogram and derived probability model of χ^2-statistic for the given region features. Solid lines stand for left region and dashed lines stand for right region. (a) The histogram of the χ^2-statistic between the given feature and the computed ones at a coarse grid. (b) The derived probability model for the left and right regions.

directly for central windows. Here boundaries are systematically shifted by several pixels because χ^2-statistic distance favors homogeneous regions.

After the initial segmentation and probability models are obtained, the segmentation is then iteratively updated based on the following local updating rule at pixel (x, y):

$$\pi_i(x,y) \;=\; (1-\lambda_\Gamma)P(\chi^2(H_{W^{(s)}(x,y)}, H_i)) + \lambda_\Gamma \textstyle\sum_{(x_1,y_1)\in N(x,y)} \frac{L^i_{N(x_1,y_1)}(x_1,y_1)}{|N(x,y)|}. \tag{2.5}$$

Here $L^i_{N(x,y)}(x,y)$ is the number of pixels in $N(x,y)$ whose current labels are i and $N(x,y)$ is a user-defined neighborhood (the eight nearest neighbors are used) and provides an approximation of the boundary term in (2.4). Parameter λ_Γ controls the relative contributions from the region and boundary terms. For a given pixel (x,y) to be updated, the spectral histogram is first estimated using asymmetric windows around the pixel. Because there are several windows to choose at pixel (x,y), for each $\mathcal{F}_i$, we use the window that has the most number of labels of R_i, and thus the feature at pixel (x,y) for different labels can be different. The new label of (x,y) is assigned the one that gives the maximum $\pi_i(x,y)$. A special case of (2.5) is for pixels along boundaries between the background region and a given region because we do not assume any model for the background region. For pixel $(x,y) \in R_0$, which is adjacent to region R_i, $i \neq 0$, if

$$\chi^2(H_{W^{(s)}(x,y)}, H_i) < \lambda_B * T_i, \tag{2.6}$$

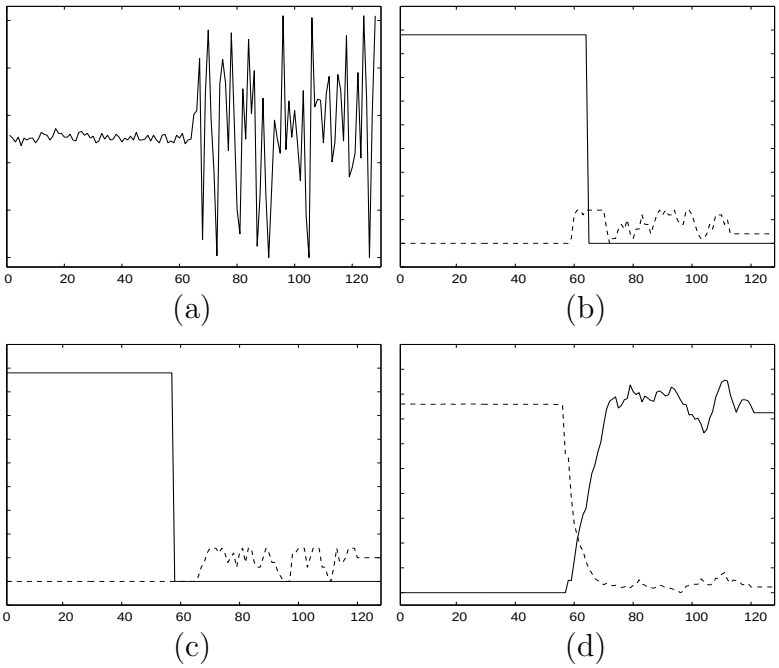


Figure 2.6. Effectiveness of derived probability models. In (b)-(d), solid lines stand for the left region and dashed lines the right region. (a) The 64th row from Fig. 2.4(a). (b) The probability of the two given regional features using asymmetric windows to compute spectral histograms, where the edge point is correctly localized. (c) Similar to (b) but using windows centered at the pixel to compute spectral histogram. Here the edge point cannot be localized. (d) χ^2-statistic from the two given regional features using centered windows, where the edge point is wrongly localized between pixel 61 and 62.

label i is assigned to (x, y). Here T_i is the estimated homogeneity threshold for region R_i, and λ_B is a parameter which determines relative penalty for unsegmented pixels. Figure 2.4(c) shows the segmentation result for the image in Fig. 2.4(a), and the resulting region boundary is shown in Fig. 2.4(d). Here all the pixels are segmented correctly. Since the image consists of two images with similar mean values but different variance, if we apply nonlinear smoothing algorithms, the segmentation result would not be accurate.

2.3 Localization of Texture Boundaries

Because textures need to be characterized by spatial relationships among pixels, relatively large integration windows are needed in order to extract meaningful features. A large

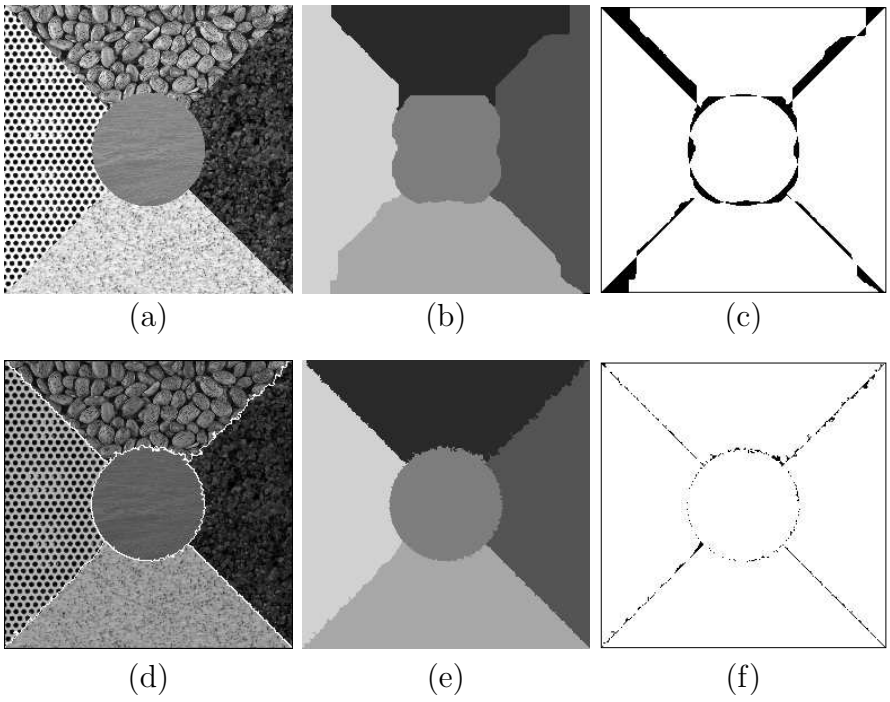


Figure 2.7. (a) A texture image with size 256×256. (b) The segmentation result using spectral histograms. (c) Wrongly segmented pixels of (b), represented in black with respect to the ground truth. The segmentation error is 6.55%. (d) and (e) Refined segmentation result shown by region boundaries and by regions respectively. (f) Wrongly segmented pixels of (e), represented in black as in (c). The segmentation error is 0.95%.

integration scale however results in large errors along texture boundaries due to the uncertainty introduced by large windows [7]. By using asymmetric windows for feature extraction, the uncertainty effect is reduced. However, for arbitrary texture boundaries, the errors along boundaries can be large even when the overall segmentation performance is good. For example, Fig. 2.7(b) shows a segmentation result using spectral histograms. While the segmentation error is only 6.55%, visually the segmentation result is intolerable due to large errors along texture boundaries, as shown in Fig. 2.7(c).

In order to accurately localize texture boundaries, we propose the following measure to refine the obtained segmentation result. As for segmentation, a refined probability model is built for given m pixels from a texture region. To capture the spatial relationship, we choose for each texture region a window as a template. Here, the template is the same window from which the region feature $\mathcal{F}$ is extracted. For the selected m pixels, we define the distance

Figure 2.8. (a) A synthetic image with size 128×128. (b) Segmentation result. (c) and (d) Refined segmentation result shown as regions and as the region boundary respectively.

between those pixels and a texture region as the minimum mean square distance between those pixels and the template. Based on the obtained result, we build a probability model for each texture region with respect to the proposed distance measure. Intuitively, if the m pixels belong to a texture region, it should match the spatial relationship among pixels when the m pixels are aligned with the texture structure. These probability models are sensitive to alignments and thus should produce more accurate region boundaries than those based on the spectral histograms.

After the probability models are derived, we use the local updating equation given in (2.5) by replacing $W^{(s)}(x, y)$ by the m pixels in a texture region along its boundary based on the current segmentation result and $\chi^2(H_{W^{(s)}(x,y)}, H_i)$ by the new distance measure. Figure 2.7(e) shows the refined segmentation result with $m = 11$ pixels and Fig. 2.7(d) shows the corresponding region boundaries. The segmentation error is reduced to 0.95% and visually the segmentation result is improved significantly. Figure 2.7(f) shows the wrongly segmented pixels of the refined segmentation.

Figure 2.8(a) shows an intensity image with a curvy boundary between two regions with the same mean but different variance. Figure 2.8(b) shows the segmentation result, Fig. 2.8(c) shows the refined result, and Fig. 2.8(d) shows the region boundary. It is clear that the boundary between two regions is improved significantly, especially at the top and bottom borders of the image.

Similar to segmentation, a special case for boundary localization is for pixels along boundaries between the background region and a texture region because there is not a probability model for the background region. To localize these pixels, we assume that the

background region is locally homogeneous along the boundary. Based on this assumption and using the χ^2-statistic measure instead of the probability model, the updating rule (2.5) becomes:

$$\pi_i(x,y) \;=\; (1-\lambda_\Gamma)(\chi^2(H_{|m_i|},H_i)) + \lambda_\Gamma \textstyle\sum_{(x_1,y_1)\in N(x,y)} \frac{L^i_{N(x_1,y_1)}(x_1,y_1)}{|N(x,y)|}. \tag{2.7}$$

where, m_i is the m pixels from region R_i in the template.

2.4 Automated Selection Algorithms

In the segmentation algorithm presented above, we assume that several representative pixels, filters and an integration scale for segmentation are given. This assumption can limit the use of the proposed method in autonomous systems. In this section, we develop algorithms for identifying seed points, selecting optimal filters, and estimating integration scales automatically for a given image.

2.4.1 Automated Seed Selection

The basic idea for seed selection algorithm is to identify homogeneous texture regions within a given image. As they are naturally normalized, spectral histograms defined on image patches of different sizes can be compared. Within a texture region relative to an integration scale, spectral histograms calculated at different windows should be similar. Based on this observation, we try to identify homogeneous texture regions based on distance measures with respect to two integration scales. Let $W^{(a)} = \lambda_s * W^{(s)}$ be an integration scale larger than $W^{(s)}$, the integration scale for segmentation. In other words, we require that $\lambda_s > 1$ and in this chapter, we set λ_s to be 1.20. We define two distance measures at pixel (x,y) with respect to $W^{(s)}$ and $W^{(a)}$, $\psi^B_{W^{(s)}}$ and $\psi^I_{W^{(s)}}$, given by:

$$\psi^B_{W^{(s)}}(x,y) = D(H_{W^{(s)}(x,y)}, H_{W^{(a)}(x,y)}), \tag{2.8}$$

and

$$\psi^I_{W^{(s)}}(x,y) = \max_{(x_1,y_1)} D(H_{W^{(s)}(x,y)}, H_{W^{(s)}(x_1,y_1)}). \tag{2.9}$$

Here $\psi^B_{W^{(s)}}$ measures the distance between the spectral histograms of two integration scales $W^{(s)}$ and $W^{(a)}$. Within a homogeneous texture region, $\psi^B_{W^{(s)}}$ should be small because

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

Figure 2.9. Texture image segmentation with representative pixels identified automatically. $W^{(s)}$ is 29×29, $W^{(a)}$ is 35×35, $\lambda_C = 0.1$, $\lambda_A = 0.2$, $\lambda_B = 2.0$, $\lambda_\Gamma = 0.2$, and $T_A = 0.08$. (a) Input texture image. (b) Initial classification result. Here the representative pixels are detected automatically. (c) and (d) Segmentation result and the resulting region boundary.

$H_{W^{(s)}}(x, y)$ and $H_{W^{(a)}}(x, y)$ should be similar. Similarly, $\psi^I_{W^{(s)}}$ measures the variation among different windows at scale $W^{(s)}$ within $W^{(a)}$ and it should be also small within a homogeneous region. In this chapter, $\psi^I_{W^{(s)}}$ is approximated in implementation using four corner windows within $W^{(a)}$. Finally, we want to choose features that are as different as possible from all those already chosen. Suppose we have chosen n features already, where, $\mathcal{F}_i = H_{W^{(s)}(x_i,y_i)}$, for $i = 1, \ldots, n$, we define

$$\psi^F(x, y) = \max_{1 \le i \le n} D(H_{W^{(s)}(x,y)}, \mathcal{F}_i), \tag{2.10}$$

which gives a distance measure between the candidate feature at (x, y) and all other chosen features. Combining these together, we have the following saliency measure:

$$\psi(x, y) \; = \; (1 - \lambda_C)(\lambda_A \times \psi^B_{W^{(s)}}(x, y) + (1 - \lambda_A) \times \psi^I_{W^{(s)}}(x, y)) - \lambda_C \times \psi^F(x, y) \tag{2.11}$$

Here λ_A and λ_C are parameters to determine the relative contribution of each term. Intuitively, $\psi(x, y)$ should be large in a region that has not been represented. Therefore we select the seed windows according to $\psi(x, y)$ until $\psi(x, y) < T_A$, where T_A is a threshold. To save computation, we compute $\psi(x, y)$ on a coarse grid.

Figure 2.9 shows a seed selection example, where the image with two texture regions is shown in Fig. 2.9(a). Figure 2.9(b) shows the initial classification result, where the feature vectors are detected automatically. Figure 2.9(c) shows the segmentation result. The texture boundary is localized well even though the two textures are similar in local intensity values.

2.4.2 Automated Filter and Integration Scale Selection

Similar to seed selection, we also can estimate the optimal filters and the optimal integration scale for each homogeneous region. While a larger integration scale tends to give smaller $\psi^B_{W^{(s)}}$ and $\psi^I_{W^{(s)}}$, a smaller integration scale is preferred in term of computational efficiency and boundary accuracy given that it is sufficient to characterize the region. Note that the optimal integration scale depends on the filters that are used in spectral histogram calculation and the choice of filters also affects the effectiveness of the spectral histogram representation. If implemented directly, this requires an iterative procedure of choosing filters and estimating the optimal integration scale.

To simplify the procedure, we adopt a simpler one that seems to be sufficient for segmentation. Given a large set of candidate filters, we estimate the optimal integration scale for each filter. Then we choose a fixed number of filters whose estimated optimal integration scale is the smallest. After the filters are chosen, we use the same procedure to estimate the final optimal integration scale but using the chosen filters.

More specifically, for one filter or a set of filters, we initially set $s = s^0$ and compute $\psi^B_{W^{(s)}}$, where $W^{(s^0)}$ is a 5×5 window. Then we do the following steps iteratively: 1) compute $\psi^B_{W^{(\lambda_s s)}}$, at a larger scale $W^{(\lambda_s s)}$, where λ_s is set to 1.20. 2) If $\psi^B_{W^{(s)}} - \psi^B_{W^{(\lambda_s s)}} < T_{\psi^B}$, stop the iteration and return $W^{(s)}$ as the optimal integration scale. Here T_{ψ^B} is a threshold. 3) Otherwise, set $s = \lambda_s \times s$ and go to step 1). Similarly, an optimal integration scale is chosen based on $\psi^I_{W^{(s)}}$. Here, a linear combination of two optimal integration scales is used as the estimated optimal integration scale.

Figure 2.10 shows an example for selecting filters and estimating the optimal integration scales on an image with four texture regions. Here 8 filters with the smallest optimal integration scale are selected from 74 candidates. Then for each region, the optimal integration scale is computed based on the eight chosen filters using the procedures given above. Figure 2.10(b) shows the wrongly segmented pixels using selected filters and the estimated optimal integration scale. The estimated integration scale for each texture region is consistent with the texture pattern scale. For example, the estimated integration scale for the top-right region is 23×23 while the one for the top-left is 13×13. For comparison, Figures 2.10(c)-(f) show the corresponding results using manually specified scales and the

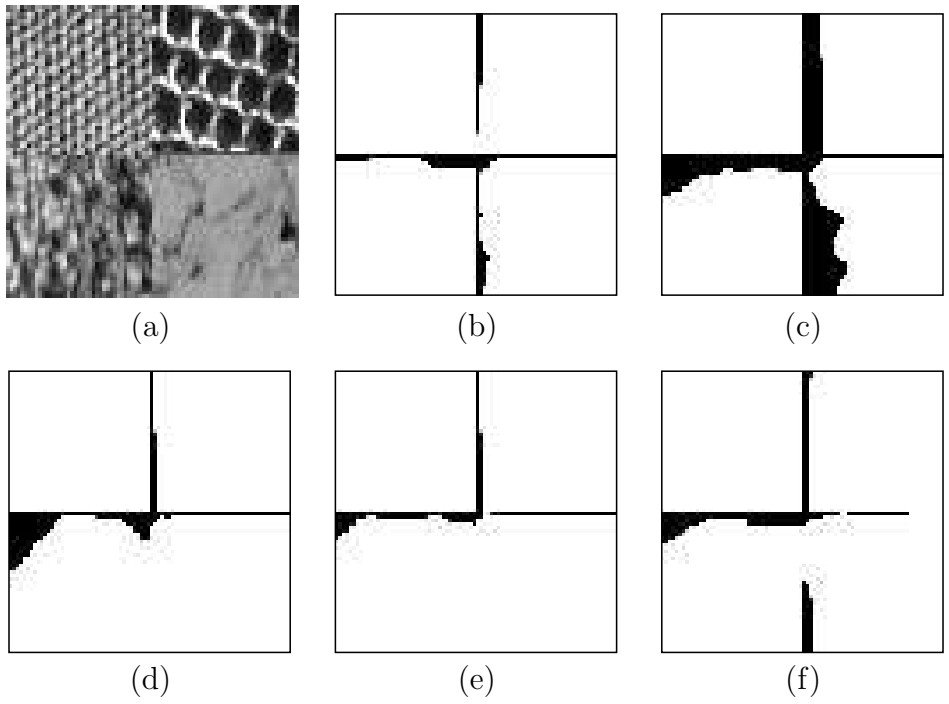


Figure 2.10. (a) A texture image of size 128×128. (b) Wrongly segmented pixels (black) using automatically selected filters and the estimated optimal integration scale, where the error rate is 4.04%. Here the integration scale is 13×13 for the top-left and bottom-right region, 23×23 for top-right region, and 15×15 for bottom-left region. (c)-(f) Wrongly segmented pixels (black) using manually specified integration scale and eight fixed filters. The integration sale is 19×19 with error rate 12.73%, 23×23 with error rate 4.99%, 29×29 with error rate = 3.15%, and 35×35 with error rate = 5.11% respectively.

eight fixed filters given in Section 2.1. This example shows that the segmentation result using automatically selected filters and the integration scale gives an accurate segmentation result with much smaller integration scales, therefore computationally more efficient.

2.5 Experimental Results and Comparison

2.5.1 Segmentation results

This section shows some additional experimental results. For all results shown here, the optimal filters and integration scales are selected automatically. Figures 2.11 and 2.12 show the segmentation results for a set of texture images. First the feature vectors are identified automatically and an initial result is then obtained using a minimum distance classifier with

the found region features. The final result is obtained by applying the boundary localization algorithm. As shown in these examples, the texture boundaries are localized well and all the homogeneous texture regions are identified by the seed selection algorithm. The inaccuracy of the boundaries is mostly due to the similarity of the textures along the boundaries as well as large texture structures and variations in these examples.

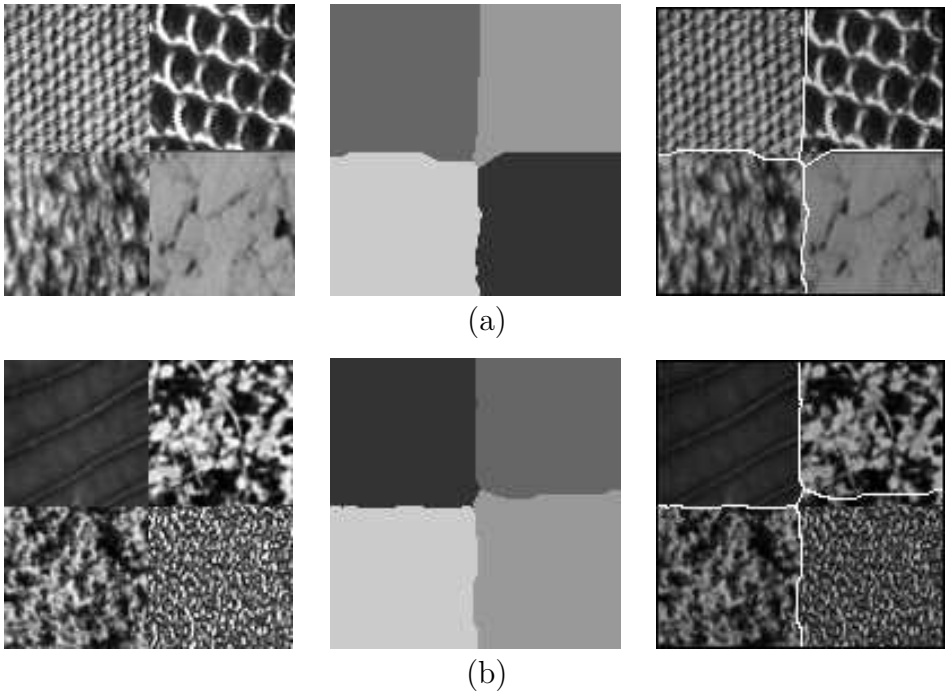
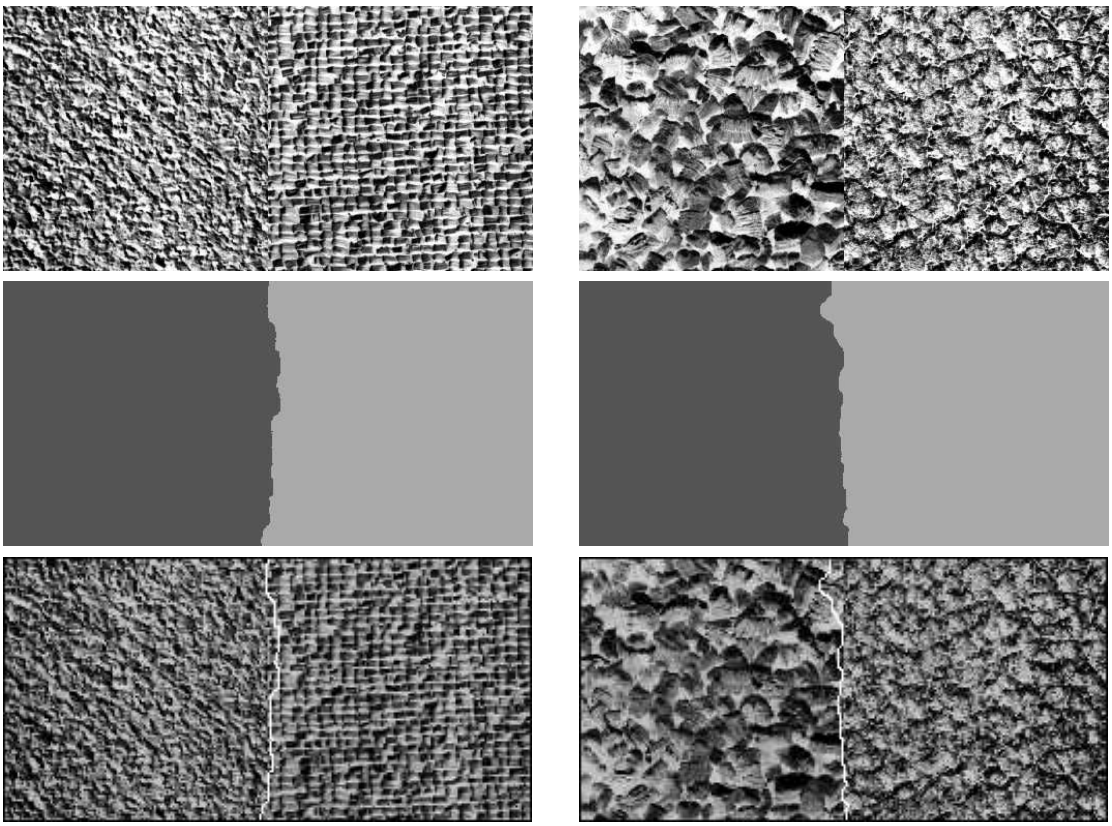


Figure 2.11. Texture image segmentation examples. In each panel, the left is the input image, the middle the final segmentation result, and the right the final region boundaries. All representative pixels are detected automatically. $\lambda_C = 0.1$, $\lambda_A = 0.2$, $\lambda_\Gamma = 0.8$, $\lambda_B = 5.0$, and $T_A = 0.20$. (a) $m = 11$. (b) $m = 15$.

Two examples shown in Fig. 2.12 are also used by Randen and Husoy [29] in their comparative study of 100 texture classification methods. Their experiments are set up as supervised classification, where training features are provided first. Thus the segmentation problem studied here is essentially more difficult[1]. Figures 2.13(a) and (b) show the classification error of the 100 texture classification methods for Fig. 2.12(a) and (b) respectively. In both cases, the segmentation error of our results is 1.0%. For Fig. 2.12(a), only four methods perform slightly better (one 0.9% and three 0.7%) than ours. However,

[1]For texture classification comparison, see [23].

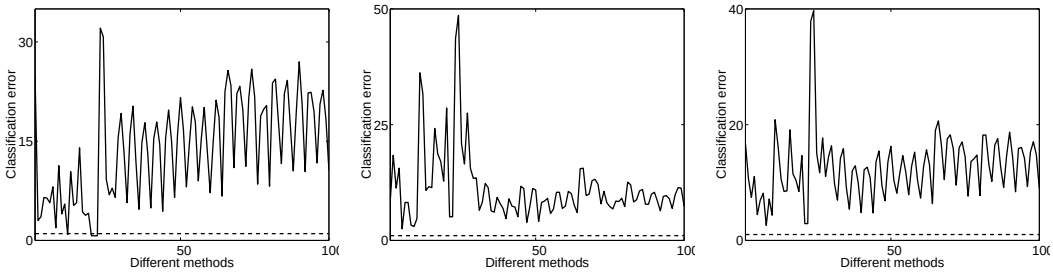


(a) (b)

Figure 2.12. Texture image segmentation examples. In each column, the top is the input image, the middle the segmentation result and the bottom the region boundaries. Here $\lambda_C = 0.1$, $\lambda_A = 0.2$, $\lambda_\Gamma = 0.2$, $T_A = 0.20$, and $m = 11$. All representative pixels are detected automatically. (a) $\lambda_B = 5.0$. (b) $\lambda_B = 4.0$.

these four methods along with others perform significantly worse than ours on Fig. 2.12(b). To show this, Fig. 2.13(c) shows the average performance on both images and clearly our method gives the lowest error rate. This significant improvement in performance is due to the desirable properties of the spectral histogram representation and the derived probability models.

Natural images in general consist of many regions that are not homogeneous regions and we are interested in some meaningful regions, called *region of interest*. This can be achieved in our system by identifying only a few region features. As mentioned before, no assumption is made in our algorithm regarding the distributions and properties of background regions, and thus we avoid building models for them. We apply the same algorithm but with one



(a) (b) (c)

Figure 2.13. Classification error of 100 methods used in [29]. In each plot, the dashed line is the corresponding segmentation error of our results. (a) For image in Fig. 2.12(a). (b) For image in Fig. 2.12(b). (c) The average performance on the two images.

region identified and Figs. 2.14 and 2.15 show some examples. Here we use the special case of the localization algorithm described in Section 4 because there is only one region of interest in each image. The left row shows the input images and the middle shows the segmentation result before boundary localization. The final boundary-localized result is shown in the right row. To show the accuracy of segmented boundaries, they are embedded in the original image. As these examples show, our algorithm gives accurate region boundaries. Some parts of the boundaries are not localized well due to the similarity between the region of interest and the background. If we had models for recognition, the cheetah in Fig. 2.15(a) and (b) could be recognized due to its distinctive skin pattern and then the segmentation result could be further improved using top-down information. Given that our system is generic and there is no image specific training, our results are comparable with the best available results.

2.5.2 Comparison with Normalized Cut

To provide additional empirical justifications of our method, we have compared our segmentation method with the normalized cut algorithm proposed by Shi and Malik [34] and further analyzed by Fowlkes et al. [12]. Here an implementation provided by Shi[2] is employed. For each case shown below, we have tried different combinations of the parameters to obtain the best performance we can. First we apply the normalized cut on the gray level

[2]Obtained from http://www.hid.ri.cmu.edu/Hid/software_ncutPublic.html

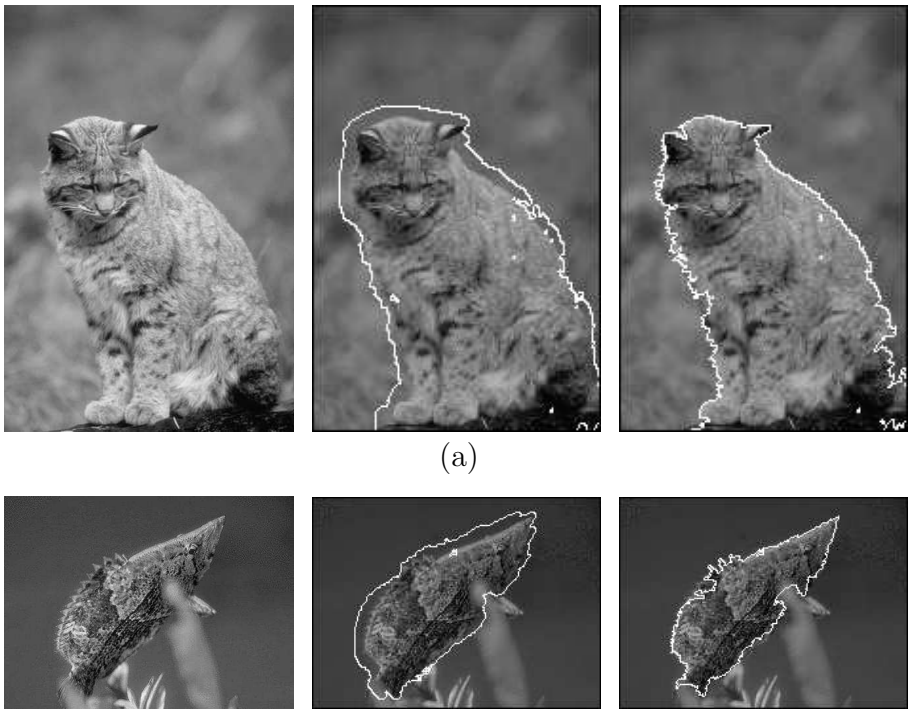


Figure 2.14. Natural image segmentation examples. In each panel, the left is the input image, the middle segmentation result before boundary localization and the right the result after boundary localization. Here $m = 25$, $\lambda_\Gamma = 0.2$. (a) A cat image with size 305×450, $\lambda_B = 5.0$. (b) A fish image with size 438×321, $\lambda_B = 9.0$.

image shown in Fig. 2.8(a) and Fig. 2.16(b) shows the segmentation result. Following Shi and Malik, Fig. 2.16(b) also shows the segmented components from normalized cut and Fig. 2.16(c) shows our corresponding segmentation and components. In this case, the normalized cut successfully segments two regions out. However, the region boundary is not well localized because the normalized cut does not exploit the spatial connectivity in segmented regions and does not have a refined model for boundary localization.

Figure 2.17 compares the normalized cut method and ours on the cheetah image shown in Fig. 2.15(b). Here we compare the segment corresponding to the cheetah region. While both methods are able to segment the cheetah out, our segment is localized more accurately. Note that the normalized cut gives the cheetah region as one of its segments and does not localize its boundary.

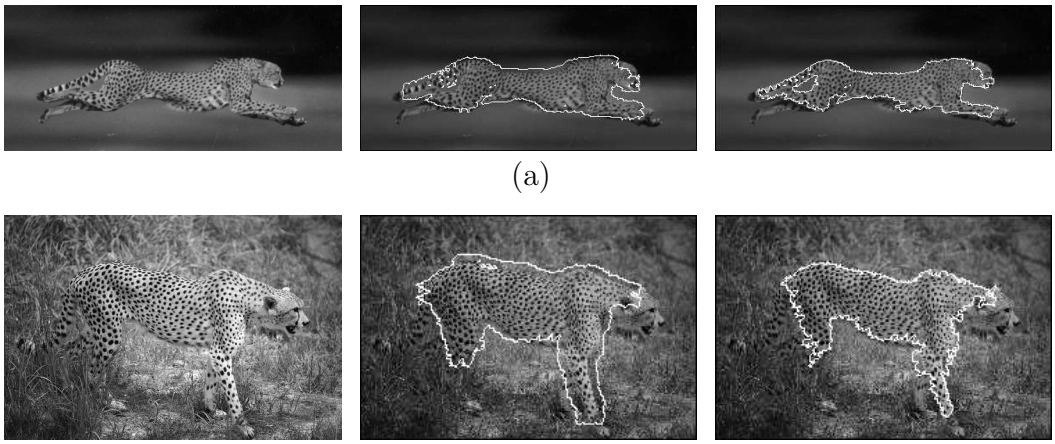


Figure 2.15. More natural image segmentation examples. See Fig. 2.14 for figure legend. Here $m = 25$, $\lambda_\Gamma = 0.2$. (a) A cheetah image with size 795×343, $\lambda_B = 2.2$. (b) Another cheetah image with size 486×324, $\lambda_B = 1.2$.

Figure 2.18(b) shows the normalized cut segmentation and its major components for the image shown in Fig. 2.7(a). For comparison, we have shown our segmentation result in the same format in Fig. 2.18(c). As in the previous example, the major regions are segmented out by their algorithm. While boundaries between very different regions are localized well, boundaries are not accurate when neighboring regions are similar. In several cases, regions from different textures are mixed. On the other hand, our segmentation gives accurate region boundaries. We attribute the difference to the derived probability models and to the fact that the spectral histogram is more reliable to characterize texture regions than filter responses, which are used directly in the normalized cut method.

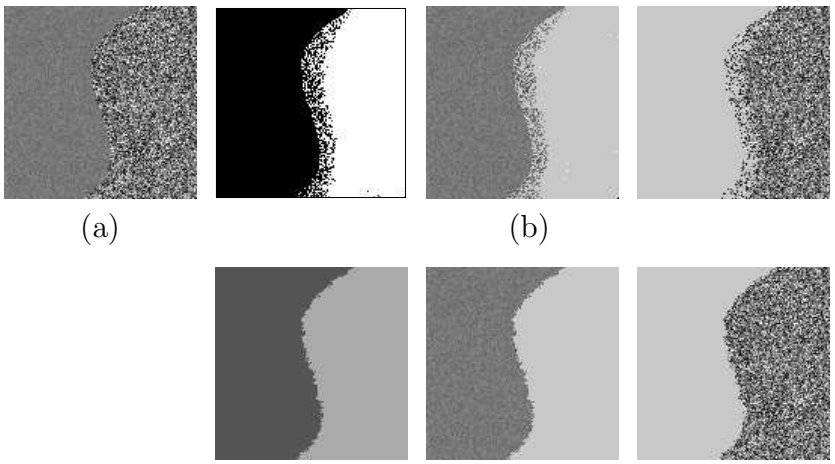


Figure 2.16. Comparison between normalized cut and proposed method on a gray-level image. (a) Input image, as shown in Fig. 2.8(a). (b) and (c) Segmentation result from normalized cut and our method. In each panel, the left is the segmentation result and the rest are the components. Here a gray color is used to indicate pixels not in the current component.

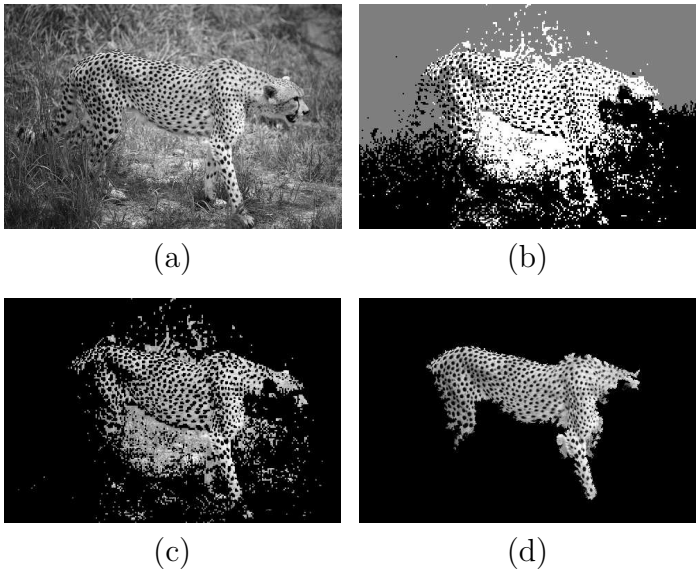


Figure 2.17. Comparison between normalized cut and the proposed method on the cheetah image (Fig. 2.15(b)). In (c) and (d), black is used to indicate pixels not in the current component. (a) Input image. (b) Segmentation result from normalized cut. (c) The cheetah segment from (b). (d) The cheetah segment from our method.

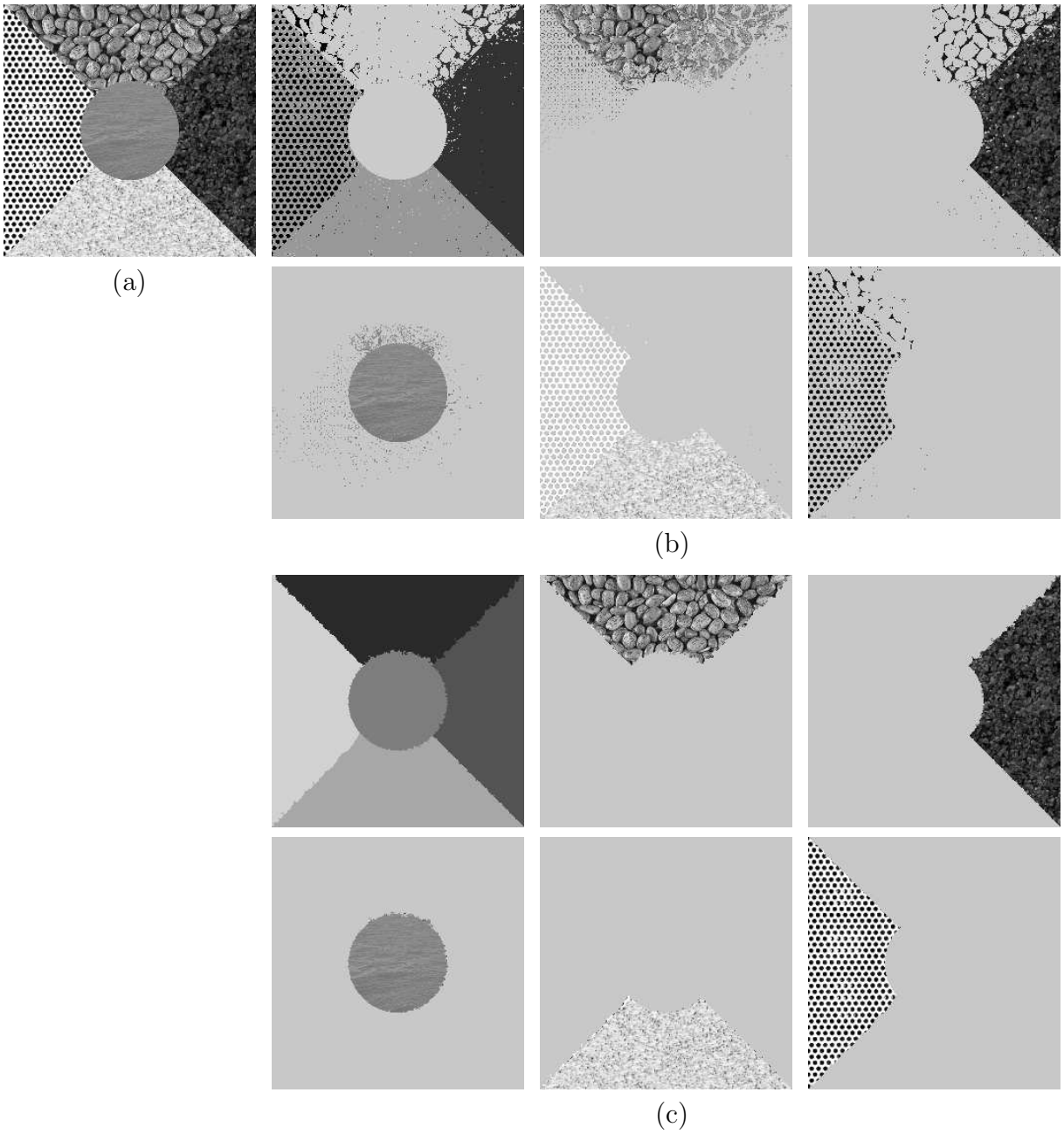


Figure 2.18. Normalized cut segmentation result and its major components for a texture image. (a) Input image as shown in Fig. 2.7(a). (b) and (c) Segmentation result and major segmented components of normalized cut and the proposed method respectively. Here a distinctive gray color is used to indicate pixels not in the current component.

CHAPTER 3

SEMANTICS ANALYSIS OF IMAGE REPRESENTATIONS FOR CONTENT-BASED IMAGE RETRIEVAL

Due to the curse of dimensionality [3], low dimensional representations for images are widely used in image processing, including content-based image retrieval. However, a low dimensional representation imposes equivalence relations in the image space and causes meaningless responses in image retrieval applications. An induced equivalence class consists of all the images that are identical under an adopted representation and is called intrinsic generalization. In this chapter, we investigate the semantics of a few commonly used low dimensional representations by sampling their intrinsic generalizations and reveal that they tend to include semantically dissimilar images in same equivalence class. We then propose a new representation called Spectral Subspace Analysis (SSA) and demonstrate improved retrieval performance using SSA representations by experimental results.

3.1 Low Dimensional Representations Analysis

In this chapter, an image $\mathbf{I}$ is defined on a finite lattice $R \subset \mathbf{Z}^2$, the intensity at pixel location $\vec{v} \in R$ is denoted by $\mathbf{I}(\vec{v}) \in \mathcal{G} = [r_1, r_2]$, where r_1, r_2 bound the dynamic range of the imaging sensor, and Ω the set of all images on R. A representation is a mapping defined as $f : \Omega \to R^K$. For a low dimensional representation, it is required that $K \ll |R|$. Before analyzing the semantics of low dimensional representations, we first introduce the equivalence relations imposed by these representations.

3.1.1 Equivalence Class of Low Dimensional Representation

Here, to introduce the equivalence relations, Bishop's framework is chosen which is laid out to formulate generalization problem in recognition [5]. Conceptually, this framework is valid for content-based image retrieval applications as the retrieval problem can be viewed as a recognition problem, where the answer is a few most similar images instead of only the most similar one. Let the query image $\mathbf{I}$ be generated from an unknown probability density P on Ω and the underlying true retrieval mapping be denoted by $h : \Omega \mapsto \mathcal{A}$, where $\mathcal{A}$ is the set of all classes. Each class consists of semantically similar images and there is no overlapping between two classes. For any retriever function g, in case of using a low dimensional representation f, its average retrieval ability is defined as the probability that $g(f(\mathbf{I})) = h(\mathbf{I})$, i.e.

$$\begin{aligned} G(g, f) &= Pr\{\mathbf{I}|\mathbf{I} \in \Omega, g(f(\mathbf{I})) = h(\mathbf{I})\} \\ &= \sum_{f(\mathbf{I})} Pr\{\mathbf{J}|\mathbf{J} \in \Omega, f(\mathbf{J}) = f(\mathbf{I})\} \end{aligned} \tag{3.1}$$

here $f(\mathbf{I})$ denotes the range of f on Ω. From (3.1), it is clear that f has a significant effect on the performance of g. Ideally, all the images from each class should be grouped as a single equivalence class. While this is generally not possible for real applications, we want to group images from each class into a small number of equivalence classes, with each class having a large cardinality, as emphasized by Vapnik [37]. However, when making each equivalence class as large as possible, we do not want to include images from other classes, as this will lead to meaningless responses. This makes it necessary to analyze the semantics through equivalence classes of low dimensional representations to achieve a good retrieval performance.

3.1.2 Semantics Analysis by Sampling Intrinsic Generalization

The previous analysis shows that the equivalence class structures of low dimensional representations are essential for a good retrieval performance. This chapter focuses on studying the semantics of a particular equivalence class through statistical sampling. First, we given a definition of intrinsic generalization.

Intrinsic Generalization : Given a representation f, the intrinsic generalization of an image $\mathbf{I}$ under f is defined as

$$S_I(\mathbf{I}) = \{\mathbf{J}|\mathbf{J} \in \Omega, f(\mathbf{J}) = f(\mathbf{I})\} \ \subset \Omega. \tag{3.2}$$

In other words, intrinsic generalization of image $\mathbf{I}$ includes all the images that cannot be distinguished from $\mathbf{I}$ under representation f. Define $S_I^0(\mathbf{I})$ as the set of images having semantically similar contents with $\mathbf{I}$. Ideally, $S_I(\mathbf{I})$ should be as close as possible to $S_I^0(\mathbf{I})$. As $S_I^0(\mathbf{I})$ is generally not available, to explore $S_I(\mathbf{I})$, we employ a statistical sampling through the following probability model:

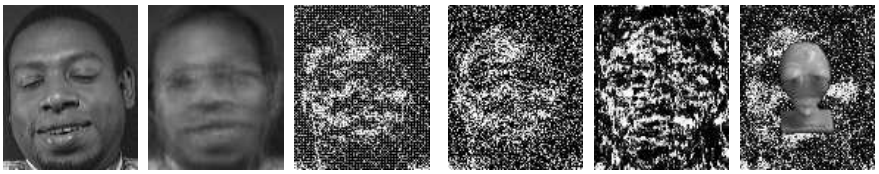
$$q(\mathbf{J}, T) = \frac{1}{Z(T)} \exp\{-D(f(\mathbf{J}), f(\mathbf{I}))/T\}. \tag{3.3}$$

Here T is a temperature parameter, $D(.,.)$ a Euclidean or other distance measure, and $Z(T)$ is a normalizing function, given as $Z(T) = \sum_{\mathbf{J}\in\Omega} \exp\{-D(f(\mathbf{J}), f(\mathbf{I})/T\}$. This model has been used for texture synthesis [39] and we generalize it to any representation. It is easy to see that as $T \to 0$, $q(\mathbf{J}, T)$ defines a uniform distribution on $S_I(\mathbf{I})$ [39]. The advantage of using a sampler is to be able to generate typical images in $S_I(\mathbf{I})$ so that $S_I(\mathbf{I})$ under f can be examined in a statistical sense.

To illustrate the effectiveness of the sampling methods, we have used a linear subspace representation PCA on the ORL face dataset[1], which consists of 40 subjects with 10 images each; we have obtained similar results using other linear subspace representations. In this case, we first calculate the eigen faces corresponding to the 50 largest eigenvalues. Under PCA, given an image $\mathbf{I}$, $f(\mathbf{I})$ is the projections of $\mathbf{I}$ along eigen faces. We define the reconstructed image of $\mathbf{I}$ as $\pi(\mathbf{I}) = \sum_{i=1}^{K} < \mathbf{I}, \mathbf{V}_i > \mathbf{V}_i$, where $\mathbf{V}_i$ is the ith eigen face and $< .,. >$ is the inner product. Fig. 3.1(a) shows a face image in the dataset and Fig. 3.1(b) shows the reconstructed image with $K = 50$. Then a Gibbs sampler is used to generate samples from $S_I(\mathbf{I})$. Fig. 3.1(c)-(f) show four samples of $S_I(\mathbf{I})$ (For Fig. 3.1(f), the object in the middle is used as boundary condition, i.e., pixels on the object are not updated). In other words, these images have the same 50 eigen decompositions, which means that these images should have the same semantics under PCA representation. Obviously, they actually do not. Note that $S_I(\mathbf{I})$ is defined on Ω and these images are far from each other in Ω.

Because $S_I(\mathbf{I})$ consists of images with semantically different contents, we argue that linear low dimensional representations of images group semantically different images into same

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



(a) (b) (c) (d) (e) (f)

Figure 3.1. (a) A face image. (b) Reconstructed image using $K = 50$ principal components. (c)-(f) Four random samples from $S_I(\mathbf{I})$, with $\pi(\mathbf{I})$ identical to the one shown in (b).

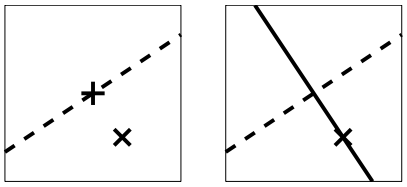
equivalence class. Therefore, these representations are not semantically meaningful, and they can make the subsequent retrieval intrinsically sensitive to noise and other deformations. To show this, Fig. 3.2(a) gives three different face images which share the exactly same eigen representation (bottom row). On the other hand, Fig. 3.2(b) shows three similar images whose eigen representations correspond to three different faces.

(a) (b)

Figure 3.2. Examples of different images with identical eigen decompositions and similar images with different eigen decompositions. The top row shows the images and the bottom reconstructed. (a) Three different images with the same eigen representations. (b) Three similar images with different eigen representations.

It needs to be emphasized here that the sampling is very different from reconstruction. In the PCA case, the sampling is to draw a typical sample from the set of all the images with a particular low dimensional representation while reconstruction gives one in the set whose coefficients are zero along the dimensions complement to the given subspace. To illustrate

this, Fig. 3.3 shows an example of a one-dimensional subspace in a two-dimensional space. In this case, the reconstructed "image" of "x" is the point given by "+" in Fig. 3.3(a) while the sampling can return any point with equal probability along the solid line shown in Fig. 3.3(b). This shows clearly that the reconstructed image may not provide much information about all the other images having the same low dimensional representation.



(a) (b)

Figure 3.3. An illustration example of the difference between sampling and reconstruction. Here the dashed line represents a one-dimensional subspace in a two-dimensional space. For a training example (marked as 'x'), the sampling is to draw a random point along the solid line in (b) while the reconstructed image is a single point given by '+' in (a).

These results, while generated based on PCA, are valid to an arbitrary linear subspace since the sampling tries to match the representation. The main problem of linear subspace representations, as revealed here, is that these representations can not take into account that most images in the image space are white noise images.

3.2 Spectral Subspace Analysis

3.2.1 Spectral Representation of Images

As discussed earlier, an ideal representation f for $\mathbf{I}$ will be such that $S_I(\mathbf{I}) = S_I^0(\mathbf{I})$. There are two important limitations of the linear methods that need to be addressed: (i) As the vast majority images in Ω are white noise images, a good approximation of $S_I^0(\mathbf{I})$ for an image of object(s) must handle white noise images effectively; otherwise, $S_I(\mathbf{I})$ will concentrate on white noise images. Earlier experiments show that linear representations suffer this problem. (ii) Another issue is the linear superposition assumption, where each basis contributes independently to the image. In contrast, pixels on objects are dependent and efficient models should exploit this dependency.

The issue of white noise images can be dealt with effectively through the method of types [10] as the white noise images are grouped together under types. However, the direct use of types does not provide enough constraints as only the histogram of images is used. We generalize the type definition by including marginals of filter responses (of the input image) with respect to a set of filters, which also incorporates local pixel dependence through filtering.

The representation of using marginals of filtered images can be justified in many ways: (i) by assuming that *small disjoint regions in the frequency domain are independent*. That is, partition the frequency domain into small disjoint regions and model each region by its marginal distribution. The partitioning of the frequency also leads to spatial filters. (ii) Wavelet decompositions of images are local in both space and frequency, and hence, provide attractive representations for objects in the images. We convolve an image with the filters and compute the marginals. Each image is then represented by a vector consisting of all the marginal distributions. This representation is called *spectral representation*, each of these vectors a *spectral histogram*, and the set of all valid vectors the *spectral space*. Elements of a spectral histogram relate to the image pixels in a nonlinear fashion, and hence, avoid the linearity issue mentioned earlier.

This representation has also been suggested through psychophysical studies on texture modeling [8], and has been used in the texture modeling and synthesis [15, 40, 22] and texture classification [23]. Both the histogram of input images [36] and joint histograms of local fields [32] have been used for object recognition.

3.2.2 Spectral Subspace Analysis

In this method, the strategy is to first represent each image in the spectral space and then apply a linear subspace method, such as PCA, ICA or FDA, in the spectral histogram space[2]. Name these corresponding methods as SPCA, SICA, and SFDA, and call them collectively as spectral subspace analysis (SSA).

[2]Note a reconstructed spectral histogram may be outside the spectral space and here we ignore this complication.

To demonstrate the effectiveness of SSA representations, we explore their intrinsic generalizations through sampling. As in the linear subspace case, SPCA is used for experiments; similar results have been obtained using other linear spectral subspaces.

First, bases in the spectral space are computed based on training images. Given an image, its spectral representation is computed and then projected onto a spectral subspace. We use a Gibbs sampling procedure to generate images that share the same spectral representation. Fig. 3.4 shows four examples; Fig. 3.4(a)-(b) show two texture images and Fig. 3.4(c)-(d) show one object image and one face image. These examples show that the spectral subspace representation captures photometric features as well as topological structures, which are important to characterize and recognize images. We have applied SSA representations to a large dataset and obtained improved retrieval performance compared to the corresponding linear subspace representations. Experimental results are shown in the next section.

3.3 Experimental Results

In this section, experimental results are presented to demonstrate the retrieval performance using linear subspace representations and the proposed SSA representations. To demonstrate convincingly, a large dataset is created by combining ORL face dataset, a texture dataset, and the COIL-100 dataset. The resulting dataset consists of 180 different classes with 40 textures, 100 objects, and 40 faces and a total of 10160 images, selected images of which are shown in Fig. 3.5 to show the variability in the dataset. This dataset is divided to two parts. The first one consists of 8060 images to be retrieved; the remaining images are used for querying.

To evaluate the performance of a particular representation, first the 8,060 images are represented under the given representation. Then for each query image, its representation is computed and compared with that of the images to be retrieved. For a given number of images retrieved, the corresponding precision and recall measures [27] are used for performance evaluation. The performance of a representation is taken as the average precision and recall of all the query images.

In our experiments, for all the linear and spectral subspace representation, we use the precision-recall curves to evaluate the retrieval performance. These curves are generated by varying the number of images retrieved. Here, the number of principal components is

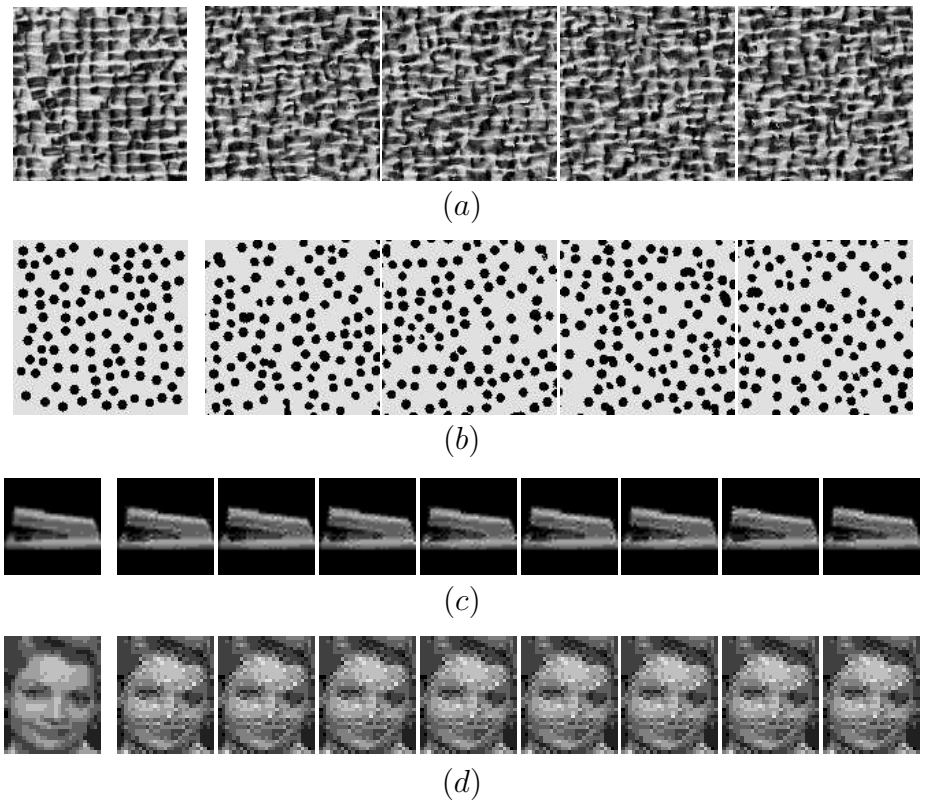


Figure 3.4. Samples from SPCA intrinsic generalization. In each row, the first column shows the input image and others samples from the intrinsic generalization. (a)-(b) Two textures. (c)-(d) One object and one face images. Boundary conditions need to be taken with care when sampling from $S_I(\mathbf{I})$.

fixed to 54 for all representations. ICA is calculated using the FastICA algorithm [17] and FDA is based on an algorithm by Belhumeur et al. [2]. To calculate the spectral histogram, a fixed set of 21 filters are used. These filters were chosen automatically from a large set using a filter selection algorithm [21] for the ORL dataset. To separate the effectiveness of a representation from that of the choice of training data, we have also used (uniformly) randomly generated bases, which we call random component analysis (RCA) and the spectral random component analysis (SRCA).

Figure 3.6 shows the precision-recall curves for three linear subspace representations vs. their corresponding SSA representations. From the results, it is easy to see that the SSA representations outperform the corresponding linear subspace ones. Note that the

Figure 3.5. Selected images from the dataset used in the retrieval experiments.

dataset consists of different kinds of images and the performance of SFDA is the best and is good especially when the number of images returned is not very large, demonstrating the effectiveness of the proposed representation for image retrieval applications.

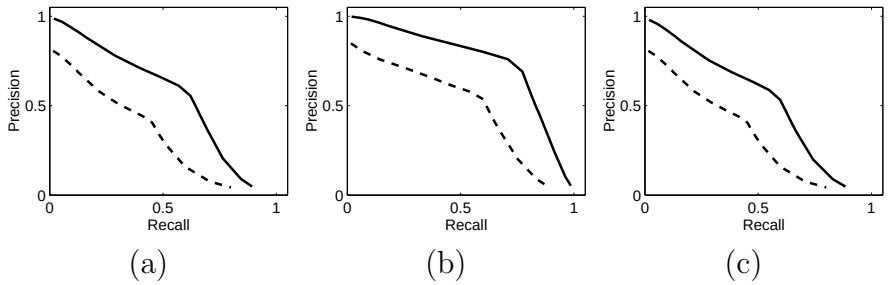


Figure 3.6. Precision-recall curves for linear subspace representations vs. corresponding SSA representations. The solid lines stand for SSA representations and the dash lines stand for linear subspace ones. (a) PCA vs. SPCA. (b) FDA vs. SFDA. (c) RCA vs. SRCA.

CHAPTER 4

LEARNING OPTIMAL REPRESENTATIONS FOR IMAGE RETRIEVAL APPLICATIONS

This chapter presents an MCMC stochastic gradient algorithm for finding representations with optimal retrieval performance on given image datasets. For linear subspaces in the image space and the spectral space, the problem is formulated as that of optimization one on a Grassmann manifold. By exploiting the underlying geometry of the manifold, a computationally effective algorithm is developed. The feasibility and effectiveness of the proposed algorithm are demonstrated through extensive experimental results.

4.1 Optimal Linear Subspace for Retrieval

We adopts an example-based learning methodology to compute representations that provide optimal retrieval performance. This assumption can be generally satisfied for image retrieval applications, as labeled images can be generated or collected interactively using some existing retrieval systems.

4.1.1 Image and Spectral Spaces

Before we introduce the main algorithm for finding optimal linear subspaces, we briefly describe two spaces that are used in this chapter, namely the image space and the spectral space. In first case, each image is viewed as one point in a high dimensional vector space. This framework has been widely used in recognition, where principal component analysis and Fisher discriminant analysis are derived based on this formulation. It is easy to see that in this representation all the images need to have the same length in order to perform dimension reduction using subspace methods. The other space is called the *spectral* space [22], which has been introduced in Chapter 3. In this space, each image is represented by a vector formed

by concatenating histograms of filtered images obtained using a set of filters. The filters can be designed based on some mathematical criteria or can be learned from images [20]. This representation has been shown to be effective for texture classification as well as face and object recognition. Recently, it has been shown systematically through sampling [20] that it is sufficient to synthesize faces and objects. See [22] and references therein for details on the spectral representation.

4.1.2 Problem Formulation

We start with a formulation of the problem for finding optimal linear representations [24, 25], where the performance can be estimated. Mathematically, let $U \in \mathbf{R}^{n \times d}$ be an orthonormal basis of an d-dimensional subspace of $\mathbf{R}^n$, where n is the size of an image (or the length of the spectral representation) and d is the required dimension of the optimal subspace ($n >> d$). For an image I (or its spectral representation), considered as a column vector of size n, the vector of coefficients is given by $a(I, U) = U^T I \in \mathbf{R}^d$. In case of spectral representation, $a(I, U) = U^T H(I)$, where $H(I)$ represents the histograms of filtered images. Let $\mathcal{G}_{n,d}$ be the set of all d-dimensional subspaces of $\mathbf{R}^n$; it is called a Grassmann manifold [6]. Let U be an orthonormal basis in $\mathbf{R}^{n \times d}$ such that $span(U)$ is the given subspace and let $F(U)$ be a retrieval performance measure associated with a system that uses U as the linear representation. That is, $F : \mathcal{G}_{n,d} \mapsto \mathbf{R}_+$ is the performance function and we want to search for the optimal subspace defined as:

$$\hat{U} = \underset{U \in \mathcal{G}_{n,d}}{\operatorname{argmax}} F(U) \ . \tag{4.1}$$

We perform the search in a probabilistic framework by defining a probability density function

$$f(X) = \frac{1}{Z(T)} \exp(F(X)/T) \ , \tag{4.2}$$

where $T \in \mathbf{R}$ plays the role of temperature and f is a density with respect to the Haar measure on the set $\mathcal{G}_{n,d}$.

4.1.3 Optimization via Simulated Annealing

We have chosen a Monte Carlo version of simulated annealing process to estimate the optimal subspace $\hat{U}$. Since the Grassmann manifold $\mathcal{G}_{n,d}$ is a curved space, the gradient

process has to account for its intrinsic geometry. We first describe a deterministic gradient process (of F) on $\mathcal{G}_{n,d}$ and then generalize it to a Markov chain Monte Carlo (MCMC) type simulated annealing process.

The performance function F can be viewed as a scalar-field on $\mathcal{G}_{n,d}$. A necessary condition for $\hat{U}$ to be a maximum is that for any tangent vector at $\hat{U}$, the directional derivative of F, in the direction of that vector, should be zero. The directional derivatives on $\mathcal{G}_{n,d}$ are defined as follows. Let E_{ij} be an $n \times n$ skew-symmetric matrix such that: for $1 \leq i \leq d$ and $d < j \leq n$,

$$E_{ij}(k,l) = \begin{cases} 1 & \text{if } k = i,\ l = j \\ -1 & \text{if } k = j,\ l = i \\ 0 & \text{otherwise .} \end{cases} \tag{4.3}$$

There are $d(n-d)$ such matrices and they form an orthogonal basis of the vector space tangent to $\mathcal{G}_{n,d}$ at identity. The gradient vector of F at any point U is defined to be a skew-symmetric matrix given by:

$$\begin{array}{l} A(U) = (\sum_{i=1}^{d} \sum_{j=d+1}^{n} \alpha_{ij}(U) E_{ij}) \quad \in \mathbf{R}^{n \times n}, \\ \text{where } \ \alpha_{ij} = \frac{F(Q_t^T e^{\epsilon E_{ij}} \tilde{I}_d) - F(U)}{\epsilon} \ . \end{array} \tag{4.4}$$

where α_{ij} is the finite approximation of the directional derivative of F in the direction given by E_{ij}, $e^{\epsilon E_{ij}}$ is an $n \times n$ rotation matrix, and $Q_t \in \mathbf{R}^{n \times n}$ is any orthogonal matrix such that $Q_t U = \begin{bmatrix} I_d \\ 0 \end{bmatrix} \equiv \tilde{I}_d \in \mathbf{R}^{n \times d}$. For numerical implementation, given a step size $\Delta > 0$, the discrete gradient process is denoted by X_t. Then, a discrete updating along the gradient direction is given by:

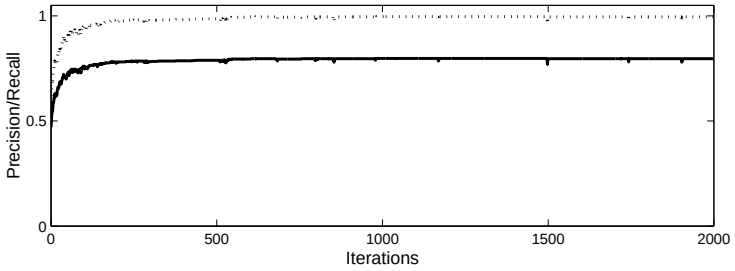
$$\begin{array}{l} X_{t+1} = Q_t^T \exp(\Delta A_t) Q X_t \ , \\ \text{where } A_t = \sum_{i=1}^{d} \sum_{j=d+1}^{n} \alpha_{ij}(X_t) E_{ij} \ . \end{array} \tag{4.5}$$

The gradient process X_t given by (4.5) can be stuck in a local maximum. To alleviate the local maximum problem, a stochastic component is often added to the gradient process to form a diffusion [14]. Both simulated annealing and stochastic gradients have [30] frequently been used to seek global optimizers [13]. To obtain stochastic gradients, we add a random component to (4.4) according to

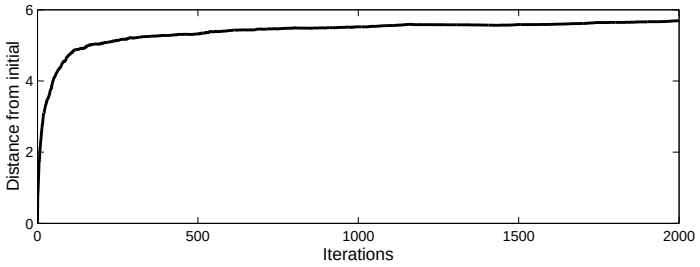
$$\tilde{A}(X_t)\Delta = A(X_t)\Delta + \sqrt{2\Delta T} \sum_{i=1}^{d} \sum_{j=d+1}^{n} w_{ij}(t) E_{ij}, \tag{4.6}$$

where $w_{ij}(t)$'s are $i.i.d$ standard normals. Under this setting, the discrete time update of the stochastic process becomes the following:

$$\begin{aligned} X_{t+1} &= Q_t^T \exp(\tilde{A}(X(t))\Delta)\tilde{I}_d, \\ Q_{t+1} &= \exp(-\Delta\, dX_t)Q_t\ . \end{aligned} \tag{4.7}$$



(a)



(b)

Figure 4.1. Temporal evolution of the optimization algorithm. Here $d = 20$, $R = 10$, $k_{db} = 8$, and $k_{query} = 2$. (a) Plots of retrieval precision (solid line) and the corresponding recall (dotted line). (b) Distance of X_t from X_0.

In case of MCMC simulated annealing, we use this stochastic gradient process to generate a candidate for the next point along the process but accept it only with a certain probability. That is, the right side of the second equation in (4.7) becomes a candidate Y that may or may not be selected as the next point X_{t+1}.

Algorithm 1 MCMC Simulated Annealing: Let $X(0) = U_0 \in \mathcal{G}_{n,d}$ be any initial condition. Set $t = 0$.

1. Calculate the gradient matrix $A(X_t)$ according to (4.4).

2. Generate $d(n-d)$ independent realizations, w_{ij}'s, from standard normal density. With X_t, calculate the candidate value Y as X_{t+1} according to (4.6) and (4.7).

3. Compute $F(Y)$, $F(X_t)$, and set $dF = F(Y) - F(X_t)$.

4. Set $X_{t+1} = Y$ with probability $\min\{\exp(dF/T_t), 1\}$, else set $X_{t+1} = X_t$.

5. Modify T, set $t = t + 1$, and go to Step 1.

The resulting process X_t forms a Markov chain. This algorithm is a particularization of Algorithm A.20 (p. 200) in the book by Robert and Casella [30]. Please consult that text for the convergence properties of X_t.

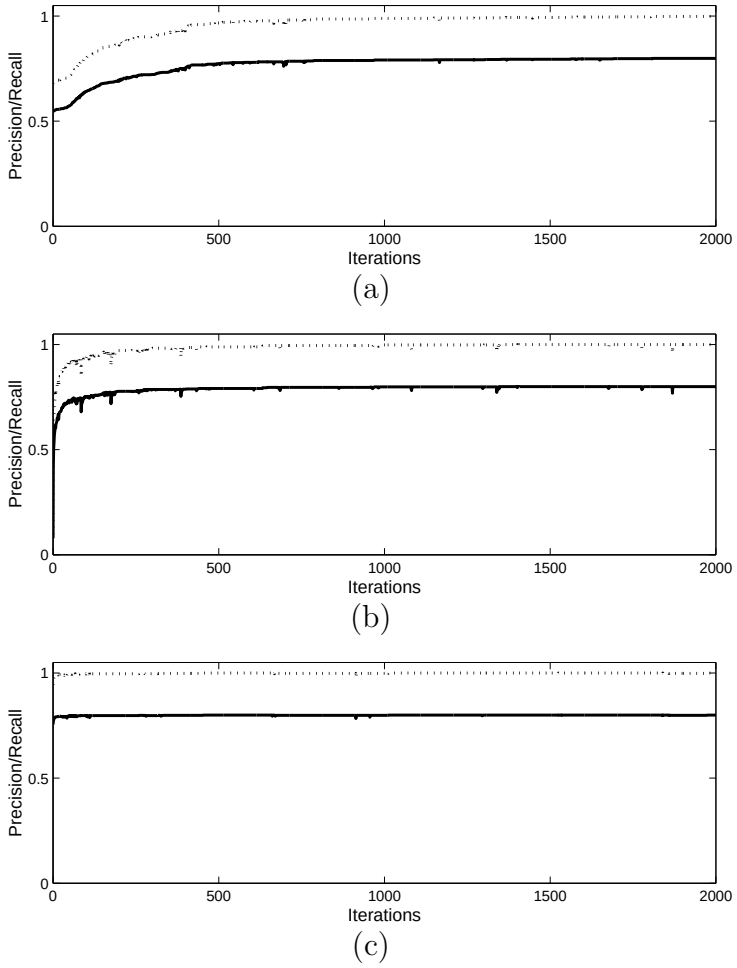


Figure 4.2. Performance of X_t versus t for different initial conditions. In each plot, the solid line represents the precision measure and the dashed line corresponding recall measure. (a) $X_0 = U_{PCA}$. (b) $X_0 = U_{ICA}$. (c) $X_0 = U_{FDA}$.

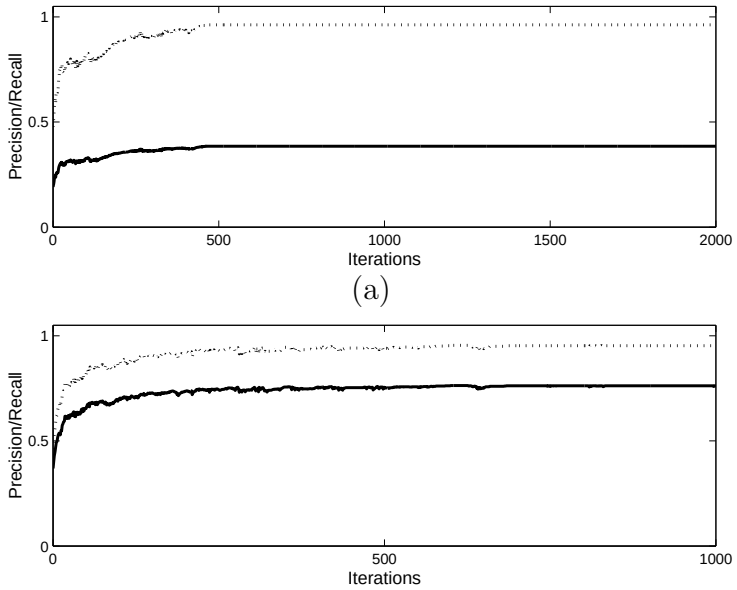


Figure 4.3. Performance of X_t versus t for different values of d and R. In each plot, the solid line represents the precision measure and the dashed line corresponding recall measure. (a) $d = 5$ and $R = 20$. (b) $d = 10$ and $R = 10$.

4.2 Experimental Results

We have applied the proposed algorithm to the search for optimal linear bases in the context of content-based image retrieval in the image space and the spectral space. Note that the algorithm requires evaluation of F (the performance measure) for any linear representation U. Here we use the retrieval precision as F with a fixed number of retrieved images [27]. To be more specific, let there are C classes in an image dataset; each class has k_{db} images (denoted by $I_{c,1}, \ldots, I_{c,k_{db}}$) to be retrieved and k_{query} query images (denoted by $I'_{c,1}, \ldots, I'_{c,k_{query}}$). Here for simplicity, we assume that each class has the same k_{db} and k_{query}, which can be modified easily to allow different numbers of images in different classes. To evaluate the precision measure, let R denote the number of images to be retrieved for each query image, we define F as the average retrieval precision for all the query images, given by

$$F(U) = \frac{1}{Ck_{query}} \sum_{c=1}^{C} \sum_{i=1}^{k_{query}} \frac{\text{No. of relevant images retrieved}}{R}. \tag{4.8}$$

Because the total number of relevant images is known in this setting, for each $F(U)$, the corresponding average *recall* measure is given by $F(U) * R/k_{db}$. Note that the optimal performance of $F(U)$ is given by $\min\{1, k_{db}/R\}$.

Before we proceed further, we briefly describe the two image datasets that have been used in our experiments: the ORL face recognition dataset[1] and a Brodatz texture dataset[2]. The ORL dataset consists of faces of 40 different subjects with 10 images each. The texture dataset consists of textures of 40 textures with 16 images in each class.

Figures 4.1 - 4.4 show the results on the ORL database with different initial conditions in both the image space and the spectral space. Figure 4.1 shows a case with a random initial condition. Fig. 4.1(b) (the distance plot) highlights the fact that the algorithm moves effectively on the Grassmann manifold going large distances along the chain. Together with Fig. 4.1(a), it shows multiple subspaces that lead to perfect performance.

Figure 4.2 shows three cases when X_0 is set to U_{PCA}, U_{ICA}, or U_{FDA}. FDA was calculated using a procedure given in [2] and ICA was calculated using a FastICA algorithm proposed by Hyvärinen [17]. In these experiments, $d = 20$, $R = 10$, $k_{db} = 8$, and $k_{query} = 2$. While these commonly used linear bases provide a variety of performances, the proposed algorithm converges to subspaces with the best retrieval precision performance regardless of the initial condition. While U_{FDA} in this particular case gives a performance close to the optimal one, however the optimality of U_{FDA} depends on the assumptions that the underlying distributions are Gaussian and linear discriminant function is used [24]. Therefore, theoretically, U_{FDA} produces only suboptimal performance (see [24] for examples). Similar to the earlier result, these results also point to the effectiveness of this optimization approach. In fact, for any chosen initial condition, the search process converges to a perfect solution (in that it gives the best achievable performance) and moves effectively along the Grassmann manifold. Also these solutions are quite different from each other, allowing additional constraints to be imposed.

We have studied the variation of optimal performance versus the subspace rank denoted by d and the number of images retrieved denoted by R. Fig. 4.3 shows two cases with $d = 5$ and $d = 10$. While the optimal solution does not achieve the best achievable performance,

[1] http://www.uk.research.att.com/facedatabase.html
[2] http://www-dbv.cs.uni-bonn.de/image/texture.tar.gz

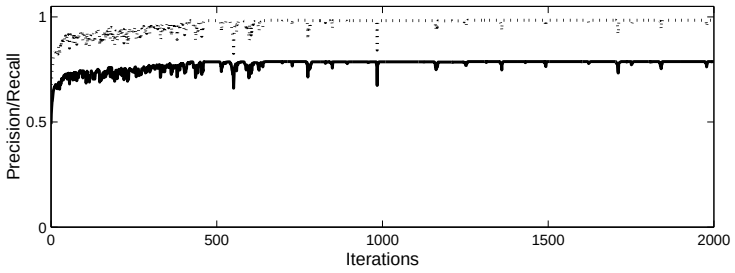


Figure 4.4. Temporal evolution of X_t on the ORL dataset in the spectral space. Here $d = 20$, $R = 10$, $k_{db} = 8$, and $k_{query} = 2$. Here solid line shows the retrieval precision and dotted line the corresponding recall.

it is very close to that as shown by the precision and recall curves. In some image retrieval applications, the computation time may be more important than the performance. The algorithm can be used to find the best compromise between accuracy and computation.

The previous three figures show different cases in the image space. As face images used here are roughly aligned, the linear representations in the image space work well on the ORL dataset. Fig. 4.4 shows a case in the spectral space on the ORL dataset. It shows that the performance using spectral representation is comparable with that in the image space. The significance of the result is that it shows the spectral representation is sufficient to characterize different faces. In addition, it can be used to characterize textures, making it a good representation for image retrieval applications where images are not confined to particular types in general.

Figures 4.5 and 4.6 show the results on the texture dataset. Fig. 4.5 shows a typical case in the image space, where Fig. 4.5(a) shows the performance and Fig. 4.5(b) the corresponding distance from X_0. As Fig. 4.5(b) shows, the MCMC algorithm moves effectively in the image space as the distance is constantly increasing. However, the performance, (shown in Fig. 4.5(a)), while improved significantly compared to the initial performance (the precision is improved from 0.139 to 0.544), is still not satisfactory. The main reason is that texture models must be translation invariant while the subspaces of the image space are not. In contrast, the subspaces in the spectral space are very effective. Fig. 4.6 shows two typical cases. The algorithm converges quickly to representations that give the optimal achievable performance. Note that while the performance does not change, the

representations are constantly evolving, which shows there are multiple solutions that have the perfect performance.

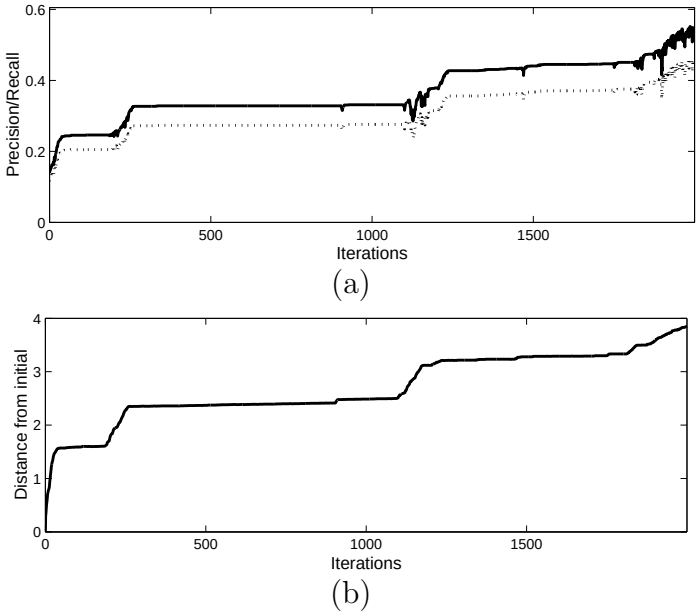


Figure 4.5. Temporal evolution of X_t on the texture dataset in the image space. Here $d = 20$, $R = 10$, $k_{db} = 12$, and $k_{query} = 4$. (a) Plots of retrieval precision (solid line) and the corresponding recall (dotted line). (b) Distance of X_t from X_0.

These results underscore two important points about Algorithm 1: (i) the algorithm is consistently successful in seeking optimal linear basis from a variety of initial conditions, and (ii) the algorithm moves effectively on the manifold $\mathcal{G}_{n,d}$ with the final solution being far from the initial condition. We have also compared empirically the performances of these optimal subspaces with the frequently used subspaces, namely U_{PCA}, U_{ICA}, and U_{FDA}. Fig. 4.7 shows the precision/recall performance for the ORL dataset in the image space. The plots are obtained by varying R, the number of retrieved images. This comparison confirms the effectiveness of the optimal representation and shows a potential significant performance improvement.

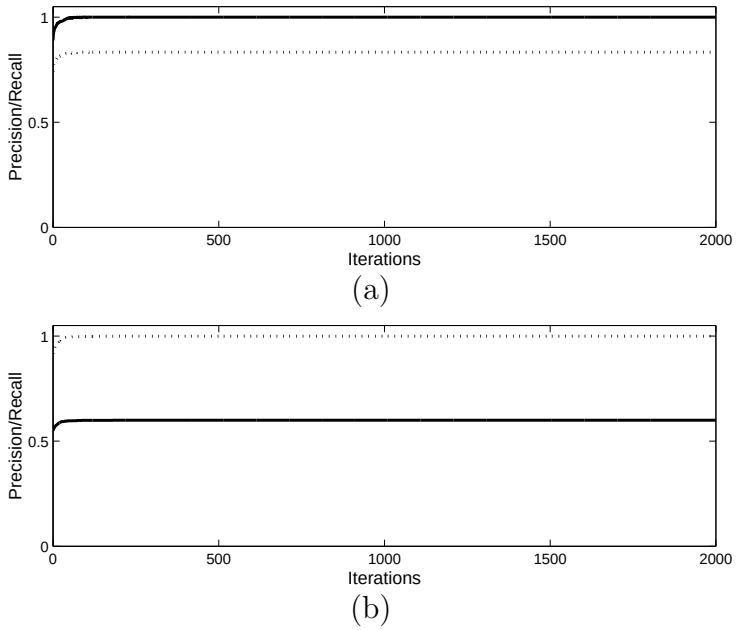


Figure 4.6. Performance of X_t versus t for the texture dataset with different values of R in the spectral space. the solid line represents the precision measure and the dashed line corresponding recall measure. Here $d = 20$, $k_{db} = 12$, $k_{query} = 4$. (a) $R = 10$. (b) $R = 20$.

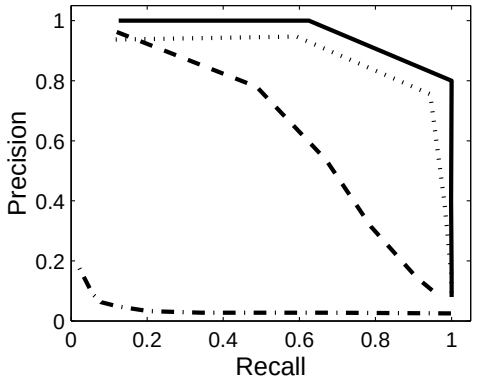


Figure 4.7. The precision/recall performance of different linear subspaces on the ORL dataset. Here solid line is the optimal basis from the gradient search process, dotted line FDA, dashed line PCA, and dash-dotted line ICA. The results are obtained by varying the number of images retrieved (R).

CHAPTER 5

CONCLUSION

In this thesis, we have presented an image segmentation algorithm using local spectral histogram representations. Related methods for automated feature selection, automated filter and integration scale selection are also developed. The experimental results show that the algorithm is effective and robust, and is comparable to other segmentation algorithms. The proposed algorithm can be combined with other representations for content-based image retrieval applications. This needs to be studied further in the future.

We have also provided a novel way, namely sampling the intrinsic generalization, to analyze the semantics of representations for content-based image retrieval applications. Using this tool, we have shown that linear subspace representations of images cannot semantically characterize images well since semantically dissimilar images tend to be grouped into the same equivalence class under these representations. A new representation, Spectral Space Analysis, is proposed to improve the intrinsic generalization by implementing linear subspaces in the spectral space, and substantial improvement in retrieval performance has been obtained.

In addition, we have proposed a simulated annealing algorithm on Grassmann manifolds for finding the optimal linear subspaces in the image space and the spectral space for image retrieval applications. The experimental results demonstrate that the algorithm provides an effective tool for improving retrieval performance. To our best knowledge, this algorithm is the first attempt to systematically find optimal representations for image retrieval applications. While the used datasets are limited compared to typical image retrieval datasets, they consist of representative natural images and therefore the experimental results are convincing and significant.

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, no. 2, pp. 148-153, 1997.

[2] 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.

[3] R. Bellman, *Adaptive Control Processes: A Guided Tour*. (Princeton University Press, New Jersey, 1961).

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

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

[6] W. M. Boothby, *An Introduction to Differential Manifolds and Riemannian Geometry*, Academic Press, 2003.

[7] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[8] C. Chubb, J. Econopouly, and M. S. Landy, "Histogram contrast analysis and the visual segregation of IID textures," *J. Opt. Soc. Am. A*, vol. 11, pp. 2350–2374, 1994.

[9] P. Comon, "Independent component analysis, A new concept?" *Signal Processing*, vol. 36(4), pp. 287-314, 1994.

[10] I. Csiszar and J. Korner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, Academic Press, New York, 1981.

[11] R. A. Fisher, "The use of multiple measures in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179-188, 1936.

[12] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nyström method", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[13] S. Geman and C.-R. Hwang, "Diffusions for global optimization," *SIAM J. Control and Optimization*, vol. 24(24), pp. 1031–1043, 1987.

[14] U. Grenander and M. I. Miller, "Representations of knowledge in complex systems," *Journal of the Royal Statistical Society*, vol. 56(3), pp. 549–603, 1994.

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

[16] H. Hotelling, "Analysis of a complex of statistical variables in principal components," *Journal of Educational Psychology*, vol. 24, pp. 417-441, 498-520, 1933.

[17] A. Hyvärinen, "Fast and robust fixed-point algorithm for independent component analysis," *IEEE Transactions on Neural Networks*, vol. 10(3), pp. 626–634, 1999.

[18] M. Lesk, "What you see is what you get, but not what you want," *Workshop on Shape-Based Retrieval and Analysis of 3D Models*, 2001 (available at http://www.cs.princeton.edu/gfx/shape01/abstracts.html).

[19] X. Liu, *Computational Investigation of Feature Extraction and Image Organization*, Ph.D. Dissertation, The Ohio State University, Columbus, Ohio, USA, 1999 (available at http://www.cis.ohio-state.edu/∼liux).

[20] 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.

[21] X. Liu and D. L. Wang, "Appearance-based recognition using perceptual components," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 3, pp. 1943–1948, 2001.

[22] 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.

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

[24] X. Liu, A. Srivastava, and Kyle Gallivan, "Optimal linear representations of images for object recognition," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 26, 2004 (Available at http://www.stat.fsu.eud/∼anuj).

[25] X. Liu and A. Srivastava, "Stochastic search for optimal linear representations of images on spaces with orthogonality constraints," in *Proceedings of the International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2003.

[26] J. M. Morel and S. Solimini, *Variational Methods for Image Segmentation*, Birkhauser, Boston, 1995.

[27] H. Muller, W. Muller, D. M. Squire, S. Marchand-Maillet, and T. Pun, "Performance evaluation in content-based image retrieval: overview and proposals," *Pattern Recognition Letters*, vol. 22, pp. 593–601, 2001.

[28] D. Mumford and J. Shah, "Optimal approximations of piecewise smooth functions and associated variational problems," *Communications on Pure and Applied Mathematics*, vol. XLII, no. 4, pp. 577–685, 1989.

[29] T. Randen and J. H. Husoy, "Filtering for texture classification: A comparative study," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 21, no. 4, pp. 291-310, 1999.

[30] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer, 1999.

[31] S. Santini and R. Jain, "Beyond Query by Example," *Proceedings of the IEEE 1998 Workshop on Multimedia Signal Processing*, Los Angelers, USA, Dec. 1998.

[32] 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.

[33] M. Sharma, *Performance Evaluation of Image Segmentation and Texture Extraction Methods in Scene Analysis*, M.S. Thesis, University of Exeter, 2001.

[34] J. Shi and J. Malik, "Normalized cut and image segmentation," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[35] A. W.M. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain "Content-based image retrival at the end of the early years," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

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

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

[38] D. L. Wang and D. Terman, "Image segmentation based on oscillatory correlation," *Neural Computation*, vol. 9, pp. 805–836, 1997.

[39] S. C. Zhu, X. Liu, and Y. Wu, "Exploring Julesz ensembles by efficient Markov chain monte carlo," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 22, no. 6, pp. 554-569, 2000.

[40] 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

Donghu Sun

Donghu Sun was born in Jiangsu, China in January, 1975. He received his B.S. in Automatic Control from Huazhong University of Science and Technology, China in 1997. From 1997 to 2001, he worked as a software engineer in Shenzhen and Shanghai, China respectively. He received his M.S. in Computer Science in 2004 from The Florida State University.