PROCEEDINGS OF SPIE

SPIE—The International Society for Optical Engineering

Applications of Digital Image Processing XXV

Andrew G. Tescher
Chair/Editor

8–10 July 2002
Seattle, Washington, USA

SPIE P PROCEEDINGS SERIES

Volume 4790

PROCEEDINGS OF SPIE
SPIE—The International Society for Optical Engineering

Applications of Digital Image Processing XXV

Andrew G. Tescher
Chair/Editor

8–10 July 2002
Seattle, Washington, USA

Sponsored and Published by
SPIE—The International Society for Optical Engineering

Cooperating Organizations
The Boeing Company (USA)
Pacific Northwest National Laboratory (USA)
Washington State University/College of Sciences
and College of Engineering and Architecture (USA)
WTC—Washington Technology Center (USA)
University of Washington/College of Engineering (USA)
University of Washington/Center for Nanotechnology (USA)

SPIE
P
PROCEEDINGS SERIES

Volume 4790

Please use the following format to cite material from this book:

Author(s), "Title of paper," in *Applications of Digital Image Processing XXV*, Andrew G. Tescher, Editor, Proceedings of SPIE Vol. 4790, page numbers (2002).

ISSN 0277-786X
ISBN 0-8194-4557-6

Published by
SPIE—The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA
Telephone 1 360/676-3290 (Pacific Time) · Fax 1 360/647-1445

Printed in the United States of America.

Conference Committee

Conference Chair

Andrew G. Tescher, Compression Science, Inc. (USA)

Program Committee

V. Ralph Algazi, University of California/Davis (USA)
Bernard V. Brower, Eastman Kodak Company (USA)
Mohammed Farhang Daemi, University of Nottingham (United Kingdom)
Touradj Ebrahimi, Swiss Federal Institute of Technology Lausanne (Switzerland)
Ali Habibi, The Aerospace Corporation (USA)
Eric Hamilton, Compression Sciences, Inc. (USA)
T. Russell Hsing, Telcordia Technologies, Inc. (USA)
C.-C. Jay Kuo, University of Southern California (USA)
Catherine Lambert-Nebout, CNES (France)
Glen G. Langdon, Jr., University of California/Santa Cruz (USA)
Andre J. Oosterlinck, Katholieke Universiteit Louvain (Belgium)
Sethuraman Panchanathan, Arizona State University (USA)
John A. Saghri, California Polytechnic State University (USA)
Pankaj N. Topiwala, FastVDO Inc. (USA)
Mohan M. Trivedi, University of California/San Diego (USA)

Session Chairs

Image Representation
John A. Saghri, California Polytechnic State University (USA)

Image Classification
John A. Saghri, California Polytechnic State University (USA)

Imaging Security
Touradj Ebrahimi, Swiss Federal Institute of Technology Lausanne (Switzerland)
Benoît M. M. Macq, Université catholique de Louvain (Belgium)

JPEG 2000
Bernard V. Brower, Eastman Kodak Company (USA)

Image Compression
C.-C. Jay Kuo, University of Southern California (USA)
Pankaj N. Topiwala, FastVDO Inc. (USA)

Space Compression
Catherine Lambert-Nebout, CNES (France)

Advanced Application Scenarios
James W. Reid, Xerox Corporation (USA)

Contents

IMAGING SECURITY

ADVANCED APPLICATION SCENARIOS

Using Kernel Principal Components for Color Image Segmentation

Slawo Wesolkowski
Systems Design Engineering
University of Waterloo
Waterloo, Ontario N2L-3G1
Canada
s.wesolkowski@ieee.org

ABSTRACT

Distinguishing objects on the basis of color is fundamental to humans. In this paper, a clustering approach is used to segment color images. Clustering is usually done using a single point or vector as a cluster prototype. The data can be clustered in the input or feature space where the feature space is some nonlinear transformation of the input space. The idea of kernel principal component analysis (KPCA) was introduced to align data along principal components in the kernel or feature space. KPCA is a nonlinear transformation of the input data that finds the eigenvectors along which this data has maximum information content (or variation). The principal components resulting from KPCA are nonlinear in the input space and represent principal curves. This is a necessary step as colors in RGB are not linearly correlated especially considering illumination effects such as shading or highlights. The performance of the k-means (Euclidean distance-based) and Mixture of Principal Components (vector angle-based) algorithms are analyzed in the context of the input space and the feature space obtained using KPCA. Results are presented on a color image segmentation task. The results are discussed and further extensions are suggested.

Keywords: Kernel Principal Component Analysis, clustering, nonlinear cluster prototypes, color image segmentation.

1. INTRODUCTION

Image segmentation is the partitioning of an image into a set of non-overlapping regions whose union is the entire image. The purpose of image segmentation is to decompose the image into parts that are meaningful with respect to a particular application. Without good image segmentation, it is not possible to understand what the image represents.

The color image segmentation field can be roughly divided into three categories: global or color clustering methods [6,13,14,15], local or region growing algorithms, and hybrid procedures [5,10]. These algorithms base their principles on physical models of the image [5,13,14,15] or on the statistical nature of images [6,10]. Most approaches have used only one color space for testing (especially RGB) while a handful such as [6] and [15] have been examined within multiple contexts. Given that any color image has three planes and therefore is by definition a multispectral image the following also applies to multispectral image segmentation algorithms.

In this paper, color clustering as a means of image segmentation will be explored [7]. Clustering algorithms segment pixels based on an inherent characteristic or feature (e.g., intensity for grayscale images) rather than on their spatial relationships. Many clustering algorithms can also be enhanced with spatial constraints. However, in this paper this will not be done explicitly. The purpose of discussing color clustering in this context is to illustrate how a change in color distance measure and feature space can help in improving image segmentation results. It would seem that most approaches in the literature use a form of Euclidean distance in RGB or some other standard color space to determine color similarity between two pixels. A brief literature review follows.

Klinker et al. [5] describe a color image segmentation algorithm based on the Dichromatic Reflectance Model. Their algorithm calculates the principal eigenvectors of small non-overlapping image regions (these regions are assumed to be smaller than the size of an object). Next, based on the characteristics of the eigenvalues, they determine whether a region

Applications of Digital Image Processing XXV, Andrew G. Tescher, Editor,
Proceedings of SPIE Vol. 4790 (2002) © 2002 SPIE · 0277-786X/02/$15.00

is homogeneous, a highlight or contains two or more objects/highlights. An algorithm merges spatially close regions that have similar color characteristics. They classify color points lying on object boundaries by using the distance between that point and the regional eigenvector. The use of the eigenvector as an object color descriptor proves to be useful and eliminates the effects of shadows. Klinker et al. further describe ways of dealing with highlights by detecting the point of intersection between the matte and highlight clusters (cf. definitions in Section 2). Their examples are convincing.

Stokman and Gevers [13] also describe a global image segmentation method based on the Dichromatic Reflectance Model. However, multispectral images are used. The cluster prototype vectors are estimated using the least squares line fitting method and color similarity is calculated by the distance between a color pixel and the cluster prototype (distance of a point from a line). Their results are encouraging.

Park et al. depict an approach based on morphological transforms [6]. They first smooth the 3-D RGB histogram using a difference of Gaussians and then apply closing and dilation operators to segment the image. The clusters follow a non-uniform expansion (i.e., a cluster could be greatly extended in the horizontal direction but not in the vertical direction). The cluster formation process is decided based on the Euclidean distance. Results on an artificial image with shadows show promise. However, results on the standard house image are not convincing. Numerous color spaces are tested.

Many clustering algorithms have been devised beginning with k-means [7]. The advantage of using clustering is that color image segmentation could be done simply and quickly. However, it is interesting to note that not much consideration is given to non-Euclidean distance measures between the pattern and the prototype cluster. Duda et al. mention that the choice of similarity measure is very important when clustering data and can drastically change the results [2]. Since the use of the Euclidean distance as a similarity measure assumes that the color space is isotropic, the clusters defined by this distance will be invariant to translations or rotations [2]. However, they will be sensitive to linear and nonlinear transformations in general, especially those that distort the distance relationships. Given that the RGB space is not a perceptually uniform color space, the assumption that it is isotropic fails and, therefore, clustering algorithms using the Euclidean distance will not necessarily yield good results in this color space when real world image scenes are analyzed.

Wesolkowski et al. [14] describe a color clustering method based on Stokham's model [12] which effectively adapted the use of eigenvectors (such as that found in Klinker et al. [5] and similar to the principles used in [13]) for the color clustering problem. They propose the use of the Mixture of Principal Components (MPC) algorithm to perform clustering in a way similar to k-means [7], but using the vector angle as a similarity measure and the principal eigenvector of the cluster as the cluster prototype. The algorithm was tested in the RGB space and compared it to k-means results in RGB and CIELUV (found to produce the best color differences using the Euclidean Distance [10]). Their qualitative and quantitative results indicate the superiority of this algorithm over k-means which uses the Euclidean distance. Most objects of one color are shown as separate regions in the image. Furthermore, this clustering method has been shown to be intensity invariant [15] with respect to the Dichromatic Reflectance Model [11] in RGB. Its effectiveness was also tested extensively on a multitude of color spaces [15].

One way of dealing with color image segmentation is by applying various color image transformations to remove the effect of shading and highlights and then applying a clustering algorithm [17]. In the case of color images, this type of approach is very appropriate. However, this type of transformation would only apply to color RGB images (which are composed of inherently correlated three spectral planes) and not multispectral images in general. Therefore, in this paper no inherent relationships between spectral planes will be assumed. The colors in the image will first be transformed into a higher dimensional space using Kernel Principal Component Analysis [8,9], a nonlinear principal component transformation. Calculating the principal components in the transformed or feature space would implicitly determine the nonlinear principal components in the input space. It will be shown that there is some benefit in using this nonlinear transformation to more effectively cluster color image data.

This paper is organized as follows. The Mixture of Principal Components algorithm is presented in Section 2. In Section 3, the Kernel Principal Components Analysis algorithm for feature extraction is described. Section 4 presents the color segmentation results by combining both approaches. Section 5 concludes this paper.

2. MIXTURE OF PRINCIPAL COMPONENTS CLUSTERING

For completeness, the Mixture of Principal Components (MPC) algorithm is presented here [1]. It has been presented earlier for the purpose of color image segmentation in [14,15]. This algorithm was adapted for color image segmentation since for color clustering the vector angle cannot be substituted directly for the Euclidean distance in the k-means algorithm [15].

It is not appropriate to substitute the vector angle for the Euclidean distance in the k-means algorithm. For k-means, the cluster prototype is the average of all the vectors in the cluster. From an intuitive point of view, the contribution of a color point to the vector mean is directly proportional to the distance between those points. The analogous notion of a cluster prototype for the vector angle is the principal eigenvector of the covariance matrix of the cluster [15]. The principal eigenvector indicates the most prominent direction of the data cluster in the given space [5]. Intuitively, the principal eigenvector is the *angular* analogy of the vector mean, just as the vector angle is analogous to the Euclidean distance in a spherical domain. Therefore, the MPC effectively unites the vector angle measure and principal component extraction into a new clustering algorithm.

Figure 2 illustrates the MPC algorithm structure for a single basis vector per cluster. Each module or cluster prototype consists of a basis vector and represents a cluster of the input data. A single color vector used for clustering within each module will be made up usually of three values. For multispectral pixel vectors, there will be a corresponding increase in size of the basis vector.

The principal eigenvector of the covariance matrix of a given cluster corresponds to a basis vector. The input vector, $\vec{x} \in \Re^N$, is linearly transformed by each of the K modules each with a defined basis vector, $\vec{w}_i$, resulting in K coefficients, $y_1, y_2, \ldots, y_K$. So, if $\vec{w}_i$ is of dimension N, the coefficient y_i is calculated as

$$y_i = \vec{w}_i^T \vec{x} \qquad (1)$$

Given the normalization of the result is done implicitly by the classifier, the raw inner vector product is used instead of the vector angle form to compute the distance between the cluster centers and the data. The vector angle formula reduces to equation 6 given $\| \vec{w}_i \| = 1$ and $\| \vec{x} \|$ is constant across all i,. The classifier chooses the output of the winning module based on the subspace classifier

$$\arg\{\max_i (y_i)\} \qquad (2)$$

The difference with [1] and [14] is that the norm of y_i is not calculated since we are now interested in differences in the sign of the coefficient. For a network with unit norm basis vectors, this classifier is equivalent to minimizing the vector angle between $\vec{w}_i$ and $\vec{x}$. The color spaces are considered to have their origin fixed at (0,0,0). The learning algorithm finds the principal eigenvector for each cluster with respect to the origin [1,4].

To compute the principal components while refining the cluster prototypes, an iterative algorithm is employed:

1. Choose the number of clusters, K.
2. Initialize the K principal components, $\vec{w}_1, \vec{w}_2, \ldots, \vec{w}_K$, to some appropriate set of values.
3. Classify a data point using the subspace classifier (minimum vector angle) using equations (1) and (2)
4. Modify the cluster prototypes, $\vec{w}_i$, using a Hebbian learning algorithm which extracts the cluster principal component [4] according to the following equation

$$\vec{w}_{i,j+1} = \vec{w}_{i,j} + m \cdot y_{i,j} \cdot \left(\vec{x}_j - y_{i,j} \cdot \vec{w}_{i,j} \right) \qquad (3)$$

where m is a learning parameter and $\vec{x}_j$ is a training vector at iteration j (the cluster prototypes $\vec{w}_i$ and the coefficient y_i are also iteration dependent).

5. If no convergence, go to 3, else stop

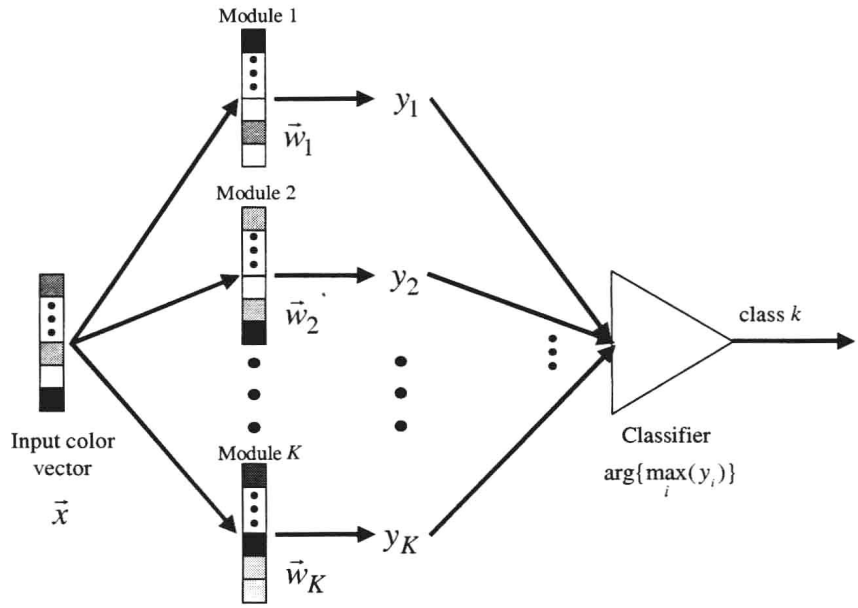

Figure 1: One pass through the MPC algorithm K clusters

This algorithm could also be represented in batch form just like the k-means algorithms. Instead of computing the vector means, the algorithm would compute the covariance matrix of each cluster and then obtain their corresponding principal eigenvectors.

3. KERNEL PRINCIPAL COMPONENT ANALYSIS

The idea of a mixture of principal components can be extended further to the idea of a mixture of nonlinear principal components. There are several ways of computing nonlinear principal components [4]. An explicit method would involve computing principal curves for each of the clusters in the input space [18]. The idea of principal curves for line fitting was first presented by Hastie and Stuetzle [3]. The authors define a principal curve as a smooth one-dimensional curve that passes through the middle of a N-dimensional data set providing a nonlinear summery of the data. Such a curve is non-parametric and it's shape is suggested by the data. The authors give a simple algorithm for determining principal curves. They suggest that a first order approximation such as a principal component be used as the initial curve. The distance from the data points to the curve is then calculated and the shape of the curve is adjusted accordingly (i.e., moving the curve towards the mean of the data that is perpendicular to it at that point). The intent is to make the curve fit the data as closely as possible. However, instead of computing principal curves in input or data space, one can compute principal components in a feature space. This idea is called kernel principal component analysis [4,8,9] and it lets one transform the input data into a high dimensional space.

Why would one want to use a nonlinear transformation of the data to determine cluster prototypes? The reason is quite simple. Not all problems can be easily clustered in the input space using an algorithm such as or similar to k-means. It might be a lot easier to work in the feature space given that the principal components would be linear in that environment. However, working *directly* in the feature space might not be tractable [8]. Therefore, an algorithm such as kernel PCA is needed to simplify the transformation of the data into a high dimensional feature space where this data would become linearly separable (or at least more so than before).

The kernel PCA algorithm presented here was described in [4]. Just like the PCA algorithm finds the eigenvectors defining the input data, the kernel PCA algorithm finds the eigenvectors defining the input data in a high dimensional feature space. The strength of the approach lies in the avoidance of the computation of the explicit form of the high dimensional feature space [9]. Kernel PCA as the name implies relies on the use of a kernel function to compute the elements of the kernel matrix. Several functions have been used in the past including polynomials, radial basis functions and hyperbolic tangents [8]. In this paper, a polynomial function of degree three is used which is defined as

$$K(\vec{x}, \vec{x}_i) = (\vec{x}^T \vec{x}_i)^p$$

The reader is referred to [9] for a more detailed derivation. The algorithm is comprised of the following steps:

1. Given the training examples $\{\vec{x}_i\}_{i=1}^N$, compute the N-by-N kernel matrix $K = \{K(\vec{x}_i, \vec{x}_j)\}$, where
$$K(\vec{x}_i, \vec{x}_j) = \varphi^T(\vec{x}_i) \cdot \varphi(\vec{x}_j)$$
and $\varphi(\cdot)$ defines the kernel which characterizes the nonlinear transformation.
2. Solve the eigenvalue problem:
$$K\vec{\alpha} = \lambda\vec{\alpha}$$
where λ is an eigenvalue of the kernel matrix $\boldsymbol{K}$ and $\vec{\alpha}$ is the associated eigenvector.
3. Normalize the eigenvectors obtained by Step 2 by requiring that
$$\vec{\alpha}_k^T \vec{\alpha}_k = \frac{1}{\lambda_k}, \qquad k = 1,2,..., p$$
where λ_k is the smallest nonzero eigenvalue of matrix $\boldsymbol{K}$, assuming that the eigenvalues are arranged in decreasing order.
4. Center the data in feature space [9].
5. For the extraction of principal components of test point $\vec{x}$, compute the projections
$$\begin{aligned} a_k &= \vec{q}_k^T \varphi(\vec{x}) \\ &= \sum_{j=1}^{N} \alpha_{k,j} K(\vec{x}_j, \vec{x}) \quad k = 1,2,..., p \end{aligned}$$
where $\alpha_{k,j}$ is the jth eigenvector of $\vec{\alpha}_k$.

The values a_k are the new feature vector elements. Typically, the N elements will not be used but only a subset p of those elements. In other words, typically one would only use the projections on a subset of eigenvectors. Furthermore, the accuracy of these eigenvectors depends on the number of data points used in the computation of the kernel matrix. If the number of training samples is high, the problem can become computationally expensive.

4. RESULTS

Experiments were conducted on a color image of a staged scene (see Figure 2). The image was previously hand segmented to identify regions of interest (see Figure 2 for illustration). Three clustering algorithms were tested on the color image segmentation problem. The algorithms used in this paper were the k-means, the MPC [14] and Kernel PCA followed by the MPC algorithm. The k-means algorithm showed no effective improvement when used on a kernel PCA transformed image and therefore will not be discussed further.

Given that this paper examines a change in similarity measure, the number of clusters for the MPC and KM approaches is varied between 2 and 24 (i.e., there is no automatic selection of the number of clusters). The initial cluster centers are chosen randomly according to a uniform distribution. KM was implemented in a batch-based approach. MPC was implemented in an online training fashion based on Oja's rule [4].

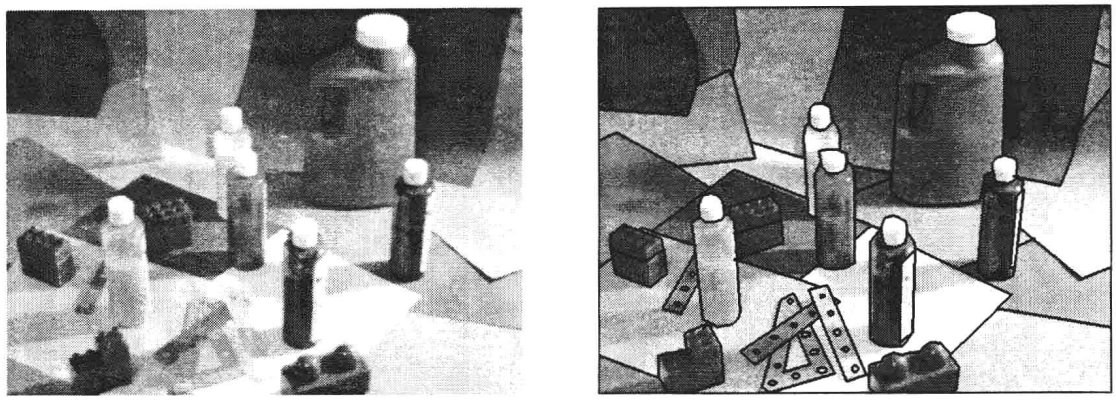

Figure 2: Original test image and its hand-segmented version

Two variants of the clustering algorithms were used. The first utilized the RGB pixel for clustering while the other used a 3x3 neighborhood of the pixel (i.e., a 27 element vector of consecutive RGB pixel values from the top left corner to the bottom right corner).

In this paper, the number of points to construct the kernel matrix was set at 500. The representative pixels were chosen randomly. This number of points required 24 hours of computation on a Dec Alpha 400 MHz machine to transform a color image. Others have used 3000 points, but for this problem this was found to be a very high number.

The number of training patterns used was 200 times the number of clusters for the 1x1 problem and 1,800 times the number of clusters for the 3x3 clustering. More pixels were used for the 3x3 problem since training in a higher dimensional space requires more data for the data to converge properly. Therefore, the total number of randomly selected training patterns used for an 8 cluster 3x3 MPC clustering (assuming no overlapping training sets) would have been (8+4+2+1)*1,800 = 27,000. That's roughly speaking 10% of the data since the image contains 283162 pixels in total. This would go up to about 20% for the 16 cluster 3x3 MPC clustering. The number of total training color vectors varied from image to image, but was always less than 50% of all the pixels in the image. However, there could be duplicates since the pixels are chosen at random from a uniform distribution. The results indicate a mixture of training and test data thus reflecting the test data when the number of clusters is small. Finally, only three projection points were used to represent the new space. The choice of making the feature space low dimensional was to show that the data could be clustered better using the same number of dimensions as in RGB.

To obtain the best possible image segmentation results, the algorithms were assessed using a quantitative image segmentation evaluation procedure [16]. In this method, the region separability and region consistency measures are plotted one against each other. The region consistency measure is calculated by determining the number of cluster-consistent pixels within a thruthed region (i.e., as determined by the hand segmentation). The region separability measure (i.e., the more the regions are separable the less overlap there is in segmented regions) is determined based on the consistency of thruthed regions falling within a segmented region. It is clear that the better algorithms will have curves approaching the top right corner of the graph (i.e., where both measures are unity). If a performance curve for algorithm A is closer in general to the point (1,1) than the curve for algorithm B, it could be said that algorithm A has better segmentation performance than algorithm B. This type of assessment can be quite arbitrary especially if the curves overlap. In general, if the curves overlap significantly, it can be said that the algorithms are equivalent.

The quantitative results are presented in Figure 3. It is clear that results obtained using the MPC either with or without KPCA are better than the best results obtained using the k-means algorithm. To illustrate this several examples of the best possible results of color clustering with those algorithms are shown in Figure 4. The results were selected on the basis of how close the points on the curves were to the point (1,1) on the graph. The results for k-means have a lot more artifacts and especially shadow effects than the results obtained with the other two methods. The MPC result does not have many shadow regions. MPC has been shown to be an intensity-invariant color segmentation algorithm for the RGB color space [15] with respect to Shafer's Dichromatic Reflectance Model [11].