Proceedings of International Conference on Image
Processing and Character Recognition

Color Texture Segmentation for Clothing Based on Finite Prolate Spheroidal Sequences

Chih-Chung Chien, Jui-Chien Wu, and Ling-Ling Wang
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 30043
Republic of China
E-mail: ling@cs.nthu.edu.tw

Abstract. *One of the most difficult tasks in computer-aided fashion design is to separate the clothing of interest on a model from backgrounds such that changes can be done on the clothing. There exist folds, shadows, diverse textures etc. on the clothing which make the segmentation work difficult. In this paper, a color texture segmentation method for clothing segmentation is proposed. Color quantization is first performed to reduce the number of colors and shadow/highlight effects on the image. The color texture features are then extracted based on the finite prolate spheroidal sequences (FPSS). By these features, a hierarchical coarse-to-fine segmentation method is used to separate the clothing from backgrounds. Finally, post-processing is applied to obtain a smooth clothing boundary. The experimental results achieved are satisfactory.*

Key words. *Finite prolate spheroidal sequences, Color quantization, Texture segmentation, Local centroid clustering.*

1 Introduction

One of the most difficult tasks in computer-aided fashion design is to separate the desired clothing from backgrounds automatically. Few commercial image processing packages can perform the segmentation well. The users either have to specify by hand the clothing boundary or tolerate the unsatisfactory result by an automatic segmentation function. There are folds, shadows, diverse textures etc. on the clothing which make the segmentation difficult. In this study, designing an algorithm that can automatically separate the clothing texture of interest from backgrounds is our aim.

To extract texture features, the statistical based methods have been widely used [1-4]. For example, the gray level co-ocurrence matrix [2], which characterizes the relationship between pixels in the spatial domain, is good for representing the random field textures with different means and variances. However, it can not characterize the structural characteristics of textures. The structural based methods [4] are good for textures which are composed of well-defined texture elements. Since many textures violate this property, the structural based methods are of limited utility. In the last decade, the space/spatial-frequency based methods [4-11] had been found of great use in texture segmentation. For examples, the orientation and frequency selective methods, such as the Gabor and Wavelet transform [6-10], have been widely used and good segmentation results have been shown. Their main drawbacks are the complicated computation and the need of prior determination of parameters. The finite prolate spheroidal sequences (FPSS) [5-7], used in this paper, was presented early in 1978 by Slepian [5], but used on image processing until 1987 by Wilson [6, 7]. Using the FPSS, we can specify intervals of both the spatial and frequency domains simultaneously, and thus can characterize textures easily in both domains. That is, the local information (relationship among pixels within a texture element) and global information (relationship among texture elements) of textures can be characterized.

The earlier methods for color image segmentation or classification used three dimensional histogram clustering techniques to segment color images in a single phase [12-16]. Different color spaces were utilized to compute the color features, such as the (R,G,B) space [12], the Munsell space [13], the (L^*,a^*,b^*) space [14], the (X,Y,I) space [15], and the (L^*,C^*,H^*) space [16]. Since a significant amount of computational effort is required in the above methods, two-phase coarse-to-fine approaches were proposed [18-20]. In the two-phase methods, the histograms were first smoothed for segmenting the color image coarsely, then clustering algorithms were applied in the fine phase to refine the segmentation results. A primary weakness of the above approaches is that they can not overcome the problem of shadows and highlights.

There are rare methods proposed for color texture image segmentation. Panjwani and Healey [21] applied the Markov random fields to model color textures. Caelli and Reye [22] presented a unified scheme to extract features in a single spatial-chromatic space. These methods performed well when no rotation, shadow, highlight, and fold existed on the textures.

In this paper, a color texture segmentation method is proposed to separate or segment the clothing of interest from backgrounds. The clothing to be segmented is specified by the user using a seed point. The clothing where the seed point locates is the one of interest. There are mainly three stages in our segmentation method. In the first stage, we first quantize the input color image to reduce both the computational cost in segmentation and the effect of shadows and highlights on the image. Then

we use the FPSS, which can characterize the textures in both spatial and frequency domains, to extract features on the clothing. In the second stage, a coarse-to-fine segmentation method is applied based on the extracted features. A clustering method (the local centroid clustering method [7]) is performed to coarsely segment the textures, and a hierarchical refinement process is utilized to refine the texture boundaries. After segmentation, each region contains textures which are homogeneous with respect to the extracted features. A simple region growing algorithm is then performed from the given seed point to locate the clothing boundary. Finally, post-processing is applied in the third stage to smooth the clothing boundary. The experimental results indicate that our proposed approach is indeed effective, which can somewhat tolerate shadows, highlights, folds, and texture orientation on the clothing.

In the remainder of this paper, detailed descriptions of the proposed approach is given in Section 2, including introduction of FPSS, the feature extraction method, the color texture segmentation method, and post-processing for boundary smoothing. Experimental results are presented in Section 3. Conclusions appear in the last section.

2 Proposed approach

There are mainly three stages in the proposed approach to separate the clothing of interest from backgrounds, including feature extraction, color texture segmentation, and post-processing.

2.1 *Feature extraction based on FPSS*

A. FPSS

Before introducing FPSS, we first define two operators: the truncation operator and bandlimiting operator. The truncation operator, $\mathbb{T}_{n1,n2}$, is defined as

$$\mathbb{T}_{n1,n2}\mathbf{M} = \begin{cases} \langle \mathbf{M} \rangle_{k\ell} & \text{if } n1 \le k \le n2 \\ 0 & \text{otherwise} \end{cases}, \quad 0 \le n1 \le n2 < n,\ 0 \le \ell < q \tag{1}$$

where $\mathbf{M}$ denotes an $n \times q$ matrix and $\langle \mathbf{M} \rangle_{k\ell}$ represents the entry in row k and column ℓ of matrix $\mathbf{M}$. $\mathbb{T}_{n1,n2}$ can also be considered as an $n \times n$ matrix:

$$\langle \mathbb{T}_{n1,n2} \rangle_{k\ell} = \begin{cases} 1 & \text{if } k = \ell \text{ and } n1 \le k \le n2 \\ 0 & \text{otherwise} \end{cases}, \quad 0 \le n1 \le n2 < n,\ 0 \le \ell < n \tag{2}$$

The output matrix $\mathbb{T}_{n1,n2}\mathbf{M}$ obtained from applying the truncation operator $\mathbb{T}_{n1,n2}$ to the matrix $\mathbf{M}$ is called a truncated matrix. A vector $\mathbf{v}$ is called index limited if it satisfies

$$\mathbb{T}_{n1,n2}\mathbf{v} = \mathbf{v} \tag{3}$$

The bandlimiting operator $\mathbb{B}_{m1,m2}$ is defined as

$$\mathbb{B}_{m1,m2} = \mathbb{F}^{*}\mathbb{T}_{m1,m2}\mathbb{F} \tag{4}$$

where $\mathbb{F}$ is the Discrete Fourier Transform (DFT) matrix [24], which is defined as

$$\langle \mathbb{F} \rangle_{k\ell} = \frac{1}{\sqrt{n}} \exp\left(\frac{-j2\pi k\ell}{n}\right), \quad 0 \le k, \ell < n \tag{5}$$

and $\mathbb{F}^{*}$, the conjugate of $\mathbb{F}$, is the inverse DFT (IDFT) matrix. $\mathbb{B}_{m1,m2}$ can also be regarded as an $n \times n$ matrix. A vector $\mathbf{u}$ is called bandlimited if the following equation is satisfied:

$$\mathbb{B}_{m1,m2}\mathbf{u} = \mathbf{u} \tag{6}$$

When a truncation operator $\mathbb{T}_{n1,n2}$ is applied to a matrix, the values of elements within the specified range (from $n1$ to $n2$) in the matrix are unchanged, but the values of elements outside the range are set to zero. The region formed by the unchanged elements in this matrix is called the truncated region. Since the truncation operation is performed directly on a matrix, it is a spatial-domain operation. The bandlimiting operator is very like the truncation operator. When applying the bandlimiting operation to a matrix (see Equation (4)), we first transform the matrix from the spatial domain to the frequency domain using the DFT matrix $\mathbb{F}$, then perform the truncation operation on the transformed matrix, and finally, transform the truncated matrix from the frequency domain to the spatial domain using the IDFT matrix $\mathbb{F}^{*}$. Note that the bandlimiting operation is a frequency-domain operation whereas the truncation operation is a spatial-domain operation.

We frequently need to specify the interval of time and the interval of frequency in analyzing a signal. We can specify the interval of time by a truncation operator with two parameters $n1$ and $n2$, and specify the interval of frequency by a bandlimiting operator with two parameters $m1$ and $m2$. However, can we use a single operator which can specify both intervals at the same time? As Equations (3) and (6) show, an index limited vector is an eigenvector of $\mathbb{T}_{n1,n2}$ and a bandlimited vector is an eigenvector of $\mathbb{B}_{m1,m2}$. It is clear that $\mathbb{T}_{n1,n2}$ and $\mathbb{B}_{m1,m2}$ are both Hermitian [25], and that in general $\mathbb{T}_{n1,n2}\mathbb{B}_{m1,m2} \neq \mathbb{B}_{m1,m2}\mathbb{T}_{n1,n2}$. Furthermore, it can be shown that no vector exists which is an eigenvector of both $\mathbb{T}_{n1,n2}$ and $\mathbb{B}_{m1,m2}$ [25]. Thus, the answer of the above question is 'no'. That is, we can not find such an operator to specify both intervals at the same time. The question is now changed for an alternative solution, "Can

we find an index limited vector which can approximate the bandlimited vector with the smallest loss of energy, or vice versa." The answer is 'yes' [6], and it is the index limited eigenvector $\mathbf{e}_0$ corresponding to the largest eigenvalue θ_0 of the operator $\mathbf{T}_{n1,n2}\mathbf{B}_{m1,m2}$:

$$\mathbf{T}_{n1,n2}\mathbf{B}_{m1,m2}\mathbf{e}_k = \theta_k \mathbf{e}_k, \quad 0 \le k < n \text{ and } \theta_0 \ge \theta_1 \ge \cdots \ge \theta_{n-1} \tag{7}$$

or the bandlimited eigenvector $\mathbf{g}_0$ corresponding to the largest eigenvalue θ_0 of the operator $\mathbf{B}_{m1,m2}\mathbf{T}_{n1,n2}$:

$$\mathbf{B}_{m1,m2}\mathbf{T}_{n1,n2}\mathbf{g}_k = \theta_k \mathbf{g}_k; \quad 0 \le k < n \text{ and } \theta_0 \ge \theta_1 \ge \cdots \ge \theta_{n-1} \tag{8}$$

The eigenvectors $\mathbf{e}_k$, $0 \le k < n$, form the FPSS. The relation between $\mathbf{e}_k$ and $\mathbf{g}_k$ [6] is

$$\mathbf{g}_k = \theta_k^{-1/2}\mathbf{B}_{m1,m2}\mathbf{e}_k, \quad 0 \le k < n \tag{9}$$

The eigenvector $\mathbf{e}_k$ is index limited and can approximate $\mathbf{g}_k$ with the minimum loss, so as to the bandlimited vector $\mathbf{g}_k$ to approximate the $\mathbf{e}_k$. Fig. 1 shows the spatial response and the frequency response of $\mathbf{e}_0$ and $\mathbf{g}_0$. Using the FPSS, we can specify both intervals in the spatial and frequency domains.

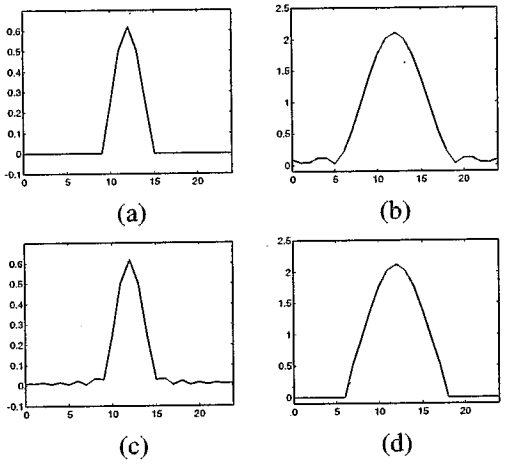


Fig. 1. Spatial and frequency response of $\mathbf{e}_0$ and $\mathbf{g}_0$ with (n=25, $n1$=10, $n2$=14, $m1$=7, $m2$=17): (a) spatial response of $\mathbf{e}_0$; (b) frequency response of $\mathbf{e}_0$; (c) spatial response of $\mathbf{g}_0$; (d) frequency response of $\mathbf{g}_0$.

B. 2-D FPSS

The FPSS introduced above is of one-dimensional (1-D) form. To process two-dimensional (2-D) images, we extend it to the 2-D form. The FPSS of 2-D form is just the combination of two 1-D FPSS's in respectively the horizontal (x) and vertical (y) directions of the Cartesian coordinate system. The truncation and bandlimiting operators of 2-D form are therefore the Kronecker products [25] of those of 1-D form. Thus, the 2-D form of Equation (7) is

$$\begin{aligned}\mathbf{TBe}_k &= \mathbf{T}_x \otimes \mathbf{T}_y \mathbf{B}_x \otimes \mathbf{B}_y \mathbf{e}_{xk} \otimes \mathbf{e}_{yk} \\ &= \theta_{xk}\theta_{yk}\mathbf{e}_{xk} \otimes \mathbf{e}_{yk}, \quad 0 \le k < n\end{aligned} \tag{10}$$

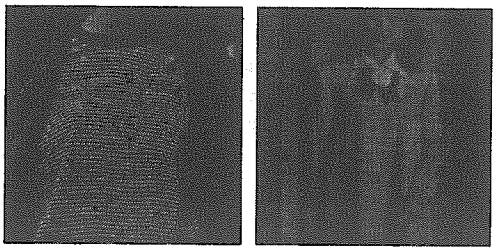
where $\mathbf{T}$ and $\mathbf{B}$ are both $n \times n$ matrices, denoting respectively the 2-D truncation operator and the bandlimiting operator. $\mathbf{T}_x$ and $\mathbf{B}_x$ are respectively the 1-D truncation operator and bandlimiting operator in the x direction, and $\mathbf{T}_y$ and $\mathbf{B}_y$ are respectively the 1-D truncation operator and bandlimiting operator in the y direction. $\mathbf{e}_k$ are the eigenvectors (2-D FPSS) of $\mathbf{TB}$. $\mathbf{e}_{xk}$ and θ_{xk} are respectively the eigenvectors (1-D FPSS) and eigenvalues of $\mathbf{T}_x\mathbf{B}_x$ with parameters ($xn1$, $xn2$, $xm1$, $xm2$), and $\mathbf{e}_{yk}$ and θ_{yk} are respectively the eigenvectors (1-D FPSS) and eigenvalues of $\mathbf{T}_y\mathbf{B}_y$ with parameters ($yn1$, $yn2$, $ym1$, $ym2$). Given the parameters ($xn1$, $xn2$, $xm1$, $xm2$) and ($yn1$, $yn2$, $ym1$, $ym2$), we can use two 1-D FPSS's $\mathbf{e}_{xk}$ and $\mathbf{e}_{yk}$ to obtain the 2-D FPSS $\mathbf{e}_k$. With these parameters, the truncated region in the spatial domain and that in the frequency domain are both rectangular. The regions truncated to rectangular shapes are of Cartesian separable form [7].

C. Color texture features

Selection of texture features is important for color image segmentation. Without good features, no matter how good the segmentation scheme is, the result will not be satisfactory. Hence, many approaches have paid emphasis on selecting or extracting promising features for texture segmentation.

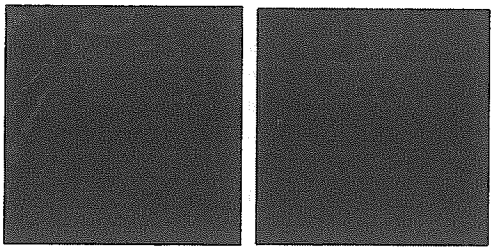
In this paper, the FPSS is used to extract texture features on the clothing. Using the FPSS, we can characterize textures in both the spatial and frequency domains. Hence, the local and global information of textures can be obtained. That is, we can use the FPSS to characterize the relationship among pixels within a texture element (local information) and that among texture elements (global information). Before extracting texture features, we first transform the input image from the RGB space to the HSI space. Next, we apply the circular hue histogram approach [27] to quantize the hue image such that both the number of colors and the influence of shadows/highlights on the image can be reduced. Fig. 2 shows two color model images, and Fig. 3 shows their quantized results.

The texture features will be obtained by convoluting the FPSS with the quantized image. Let the truncated region of a FPSS in the spatial domain be S, and that in



(a) (b)

Fig. 2. Two color model images.



(a) (b)

Fig. 3. Quantized results of images in Fig. 2: (a) quantized to 7 colors; (b) quantized to 6 colors.

the frequency domain be Ω. Assume that the areas of S and Ω are $A(S)$ and $A(\Omega)$, respectively. If they satisfy

$$A(S)A(\Omega) = MN \tag{11}$$

where M and N are the width and height of the image, respectively. The first eigenvector $\mathbf{e}_0$ of the FPSS is an effective basis for the whole spatial and frequency domains [6, 7]. Consequently, if we choose the parameters of the FPSS such that the Equation (11) holds, the texture features generated from the first eigenvector of each FPSS will be promising for segmentation.

We can truncate the frequency domain to a set of disjoint regions, which constitute a tessellation. Different FPSS's can be generated when we use different tessellations. In the spatial domain, the truncated region is located in the center, and its size is the same as that of the convolution window. In the frequency domain, the Cartesian separable tessellation [7], as shown in Fig. 4, is used.

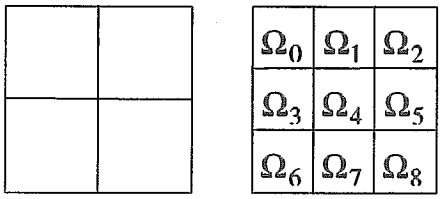


Fig. 4. Tessellations used in the frequency domain: Cartesian separable tessellation.

In the paper, the 9-regions Cartesian separable tessellation is used. Let the 9 regions of the tessellation be denoted as $\Omega_0, \Omega_1, \cdots, \Omega_8$. Given a quantized image $\mathbb{I}$ and the set of eigenvectors $\mathbf{e}_0(\Omega_i)$ of FPSS's, $0 \le i \le 8$, we define the texture feature vector $\mathbb{F}(x, y)$ for each point (x, y) in the image as follows:

$$\mathbb{F}(x, y) = (f_0(x, y), f_1(x, y), \cdots, f_8(x, y))$$

with

$$f_i(x, y) = \left| \mathbb{I}(x, y) * \mathbf{fpss}_i \right|, \quad 0 \le i \le 8$$

and

$$\langle \mathbf{fpss}_i \rangle_{k\ell} = \langle \mathbf{e}_0(\Omega_i) \rangle_{k \times n + \ell} \tag{12}$$

where * is the convolution operator. These feature vectors will be used in segmentation in the next section.

2.2 Color texture segmentation

The texture feature vectors $\mathbb{F}(x, y)$ are used as features here for texture segmentation. The segmentation method is a multi-dimensional hierarchical coarse-to-fine approach. In the method, we first build a hierarchical structure based on the feature vectors in a bottom-up manner. Next, we segment the top level (the coarsest resolution level) of the structure by a local centroid clustering algorithm [7]. Finally, using the coarse segmentation results, we proceed downwards the hierarchical structure to refine the boundaries of texture regions level by level until the bottom level (the finest resolution level) is reached.

A. Building a hierarchical structure

The hierarchical structure that we build for segmentation is the so-called quadtree [7]. Each feature vector $\mathbb{F}(x, y)$ extracted from the FPSS is considered as a point in the bottom level of the quadtree. Each level, except the bottom level, of the quadtree is constructed from its lower level. The value of each point in the ℓ-th level is computed from the values of four points in the $(\ell-1)$-th level in an averaging manner. Assume that the bottom level is of $2^n \times 2^n$ points. Let the value of a point located at (x, y) in the ℓ-th level of the quatree be denoted as $\mathbf{q}(x, y, \ell)$. Then

$$\mathbf{q}(x, y, \ell) = \frac{1}{4} \sum_{u=0}^{1} \sum_{v=0}^{1} \mathbf{q}(2x + u,\ 2y + v,\ \ell - 1), \quad 0 \le x, y < 2^{n-\ell} \tag{13}$$

Note that $\mathbf{q}(x, y, 0) = \mathbb{F}(x, y)$. The size of each level (i.e., the number of points in each level) is one-fourth of that of its lower level. When a level of size 16×16 points is obtained, we terminate the quadtree construction process. That is, the top level is of size 16×16 points. The size of the top level is obtained by experience. The noise is reduced in the higher level after the averaging operation is performed, but the boundaries of texture regions are blurred at the same time. Therefore, segmenting the top level of the quadtree, we obtain a coarse segmentation

result. Proceeding downwards the quadtree, we can refine the segmentation results to locate the correct texture boundaries.

B. Segmenting the top level of the structure

The segmentation work starts from the top level of the quadtree. The segmentation result of the top level will have great influence on the entire segmentation process. A misclassified point $\mathbf{q}(x, y)$ in the top level will result in the misclassification of four points (which are used to generate $\mathbf{q}(x, y)$ in quadtree construction) in its lower level. Furthermore, when segmenting downwards, the number of misclassified points grows four times when its lower level is processed. Consequently, segmentation of the top level is of importance. At the top level there are two steps to be performed:

1. Local centroid clustering: segmenting the texture image.
2. Insignificant regions removal: removing regions whose number of points is small.

The local centroid clustering is an iterative process [7]. The feature set used here is the set of features $\mathbf{q}(x, y, L)$ (L denotes the top level of the quadtree). For each point $\mathbf{q}$ in the feature set, we compute the local center $LC_{\mathbf{q}}$ from its neighboring points $\mathbf{p}$ in a window W_R of a specified radius R as follows:

$$LC_{\mathbf{q}} = \frac{\sum_{\mathbf{p}\in W_R} \mathbf{p}}{\sum_{\mathbf{p}\in W_R} 1} \tag{14}$$

Each point will be moved to a new local center or stay unchanged if its location is the same as that of the local center. Then a new iteration is performed again using the new locations of all points. The process terminates when all points stay unmoved. The points which occupy the same location form a class. In the paper, weights are added to Equation (14) for computing the local center as follows:

$$LC_{\mathbf{q}} = \frac{\sum_{\text{all p}} w_{\mathbf{qp}}\mathbf{p}}{\sum_{\text{all p}} w_{\mathbf{qp}}}$$

with

$$w_{\mathbf{qp}} = \frac{1}{\exp\left(d_{\mathbf{qp}}/10\right)} \tag{15}$$

where $d_{\mathbf{qp}}$ is the Euclidean distance from $\mathbf{q}$ to $\mathbf{p}$. By using Equation (15), we need not to specify the window of $\mathbf{q}$ but use all the points in the feature space to compute its local center. Its advantage is that no a priori information on the number of classes or the class centers is required.

The purpose of this paper is to separate the clothing of interest from backgrounds. We assume that the clothing to be segmented occupies a significant percentage in the image. There may be buttons, pockets or folds on the clothing whose textures are different from the clothing texture. They are usually classified into classes with small areas in the top level. Consequently, we remove these insignificant classes or regions and reassign them such that the whole clothing region can be obtained.

The insignificant regions are removed using the following steps [23]. We first find all regions whose number of points are small. Second, for each point $\mathbf{q}(x, y, L)$ in these regions, determine its neighboring classes C_k $(0 \le k \le 7)$ in its eight directions (see Fig. 5). If the class of $\mathbf{q}(x+u, y+v, L)$, $-1 \le u, v \le 1$, is different from that of $\mathbf{q}(x, y, L)$, then the class of $\mathbf{q}(x+u, y+v, L)$ is stored as a neighboring class of $\mathbf{q}(x, y, L)$. Otherwise, we consider the next point $\mathbf{q}(x+2u, y+2v, L)$ in the same direction and compare the classes of $\mathbf{q}(x, y, L)$ and $\mathbf{q}(x+2u, y+2v, L)$. If they are different, the class of $\mathbf{q}(x+2u, y+2v, L)$ is stored as a neighboring class of $\mathbf{q}(x, y, L)$. If they are the same, we consider the next point $\mathbf{q}(x+3u, y+3v, L)$ in the same direction. The process is repeated until a neighboring class of $\mathbf{q}(x, y, L)$ is found. After we obtain eight neighboring classes for the point $\mathbf{q}(x, y, L)$, we reassign $\mathbf{q}(x, y, L)$ to one of the neighboring classes whose center has minimum Euclidean distance [24] to $\mathbf{q}(x, y, L)$. For example, in Fig. 5, the eight neighboring classes of $\mathbf{q}$ are $(C_0, C_1, \cdots, C_7) = (1,3,3,3,3,2,2,2)$. Then we reassign $\mathbf{q}$ to class 1, assuming the distance from the center of class 1 to $\mathbf{q}$ is less than those from class 2 and class 3.

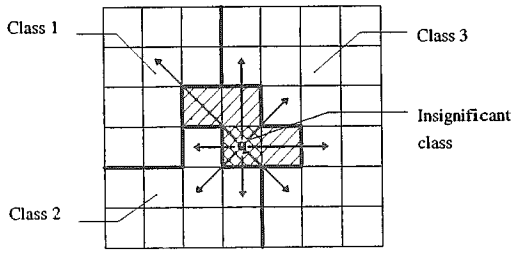


Fig. 5. Eight directions of the point $\mathbf{q}$.

After removing insignificant regions, segmentation of the top level is completed.

C. Segmenting downwards from the top level of the structure

The class boundaries obtained in the top level are blurred or rough. Proceeding downwards the quadtree level by level from the top, we can refine the boundaries. The following steps are used to compute the texture boundaries at the ℓ-th level of the quadtree:

1. Initialize the set of boundary points, $B(\ell)$, in the ℓ-th level to empty set.
2. Apply the following rule to each point $\mathbf{q}(x, y, \ell)$ in the ℓ-th level for finer classification:

   if the classes of $\mathbf{q}(\frac{x}{2}+u, \frac{y}{2}+v, \ell+1)$ are the same for all intergers u and v, $-1 \le u, v \le 1$,
   then assign $\mathbf{q}(x, y, \ell)$ to the class of $\mathbf{q}(x/2, y/2, \ell+1)$
   else $B(\ell) = B(\ell) \cup (x, y)$.
3. For each point $\mathbf{q} \in B(\ell)$, assign it to one of the classes of its corresponding 9 points in the $(\ell+1)$-th level (see Fig. 6) whose center has minimum distance to $\mathbf{q}$.

The above steps are repeated for each level until the bottom level is reached.

By performing the refinement process level by level from top to bottom, the boundaries will be located more precisely. Finally, we obtain the class of each point on the image. We then apply a simple region growing algorithm [24] starting from the given seed point to locate the whole clothing boundary. Fig. 7 shows the segmentation results of the model images in Fig. 3.

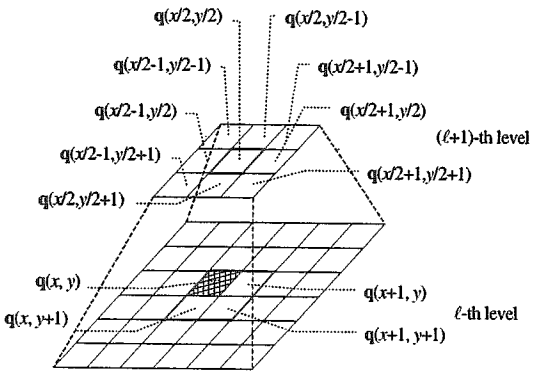
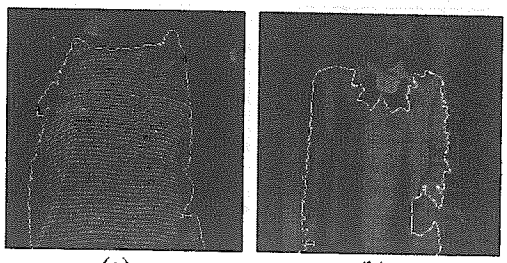


Fig. 6. The corresponding 9 points in the $(\ell+1)$-th level for the point $\mathbf{q}(x,y)$ in the ℓ-th level.

2.3 Post-processing

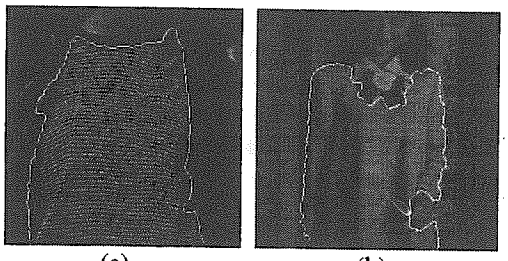
The extracted clothing boundary obtained in the previous stage is frequently jagged. Hence, post-processing is necessary to obtain a smooth clothing boundary. There are two steps in post-processing, including the morphological filtering [24] and the Gaussian smoothing [26].



(a) (b)

Fig. 7. Segmentation of images in Fig. 3 without post-processing.

The morphological filtering process consists of the opening and closing operations. The opening operation can eliminate the protrusions, while the closing operation can fill the gaps. We use the filter on the clothing region to remove small protrusions and gaps on the clothing boundary. The filtering results of images in Fig. 7 are shown in Fig. 8.



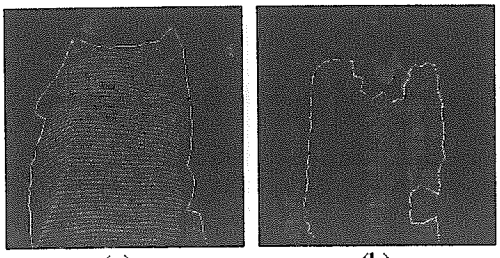
(a) (b)

Fig. 8. Morphological filtering of images in Fig. 7.

After morphological filtering, the clothing boundary is still a little jagged though the protrusions and gaps are removed. We use the Gaussian smoothing algorithm on the clothing boundary to smooth the boundary. The Gaussian smoothing algorithm uses the Gaussian function to convolute an input 1-D signal for smoothing the signal. In the algorithm, the spread parameter (or the standard deviation) of the Gaussian function is automatically determined by an iterative process [26]. To apply the algorithm, we represent the boundary as a sequence of points with their (x, y) coordinates: (x_0, y_0), (x_1, y_1), $\cdots$, (x_{n-1}, y_{n-1}), where n is the number of boundary points. The x and y coordinates of the points form two 1-D signals $(x_0, x_1, \cdots, x_{n-1})$ and $(y_0, y_1, \cdots, y_{n-1})$, respectively. Then we apply the automatic Gaussian smoothing algorithm on these two 1-D signals to smooth the signals. Finally, The clothing boundary can be polygonally approximated based on the two smoothed 1-D signals [26]. The smoothing results of images in Fig. 8 are shown in Fig. 9.

3 Experimental results

In the experiments, the truncated region in the spatial domain is of size 11×11 points. Each of the 9 truncated regions in the frequency domain is of size 23×23 points and the whole frequency domain is of size 69×69 points.



(a) (b)

Fig. 9. Gussian smoothing of images in Fig. 8.

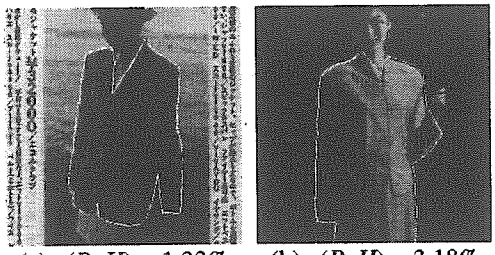
Some experimental results are shown in Fig. 10-13. Fig. 10 shows the segmentation results of two non-textured clothing. The shadows and highlights on the clothing are evident. Fig. 11 shows the segmentation results of two textured clothing. In these images, shadows, highlights, and folds are apparent on the clothing. Fig. 12 shows the segmentation results of two textured clothing with larger pattern prints. Fig. 13 shows the segmentation results of two textured clothing with serious folds. It takes about two minutes in total for each 256×256 image to separate the clothing of interest from backgrounds.

To evaluate the experimental results quantitatively, an error function is defined. Let P be the clothing region whose boundary is found by the proposed method, and H be that whose boundary is specified by hand. The error function $e(P, H)$ is defined as

$$e(P, H) = \frac{A(P \cup H) - A(P \cap H)}{A(P \cup H)}$$

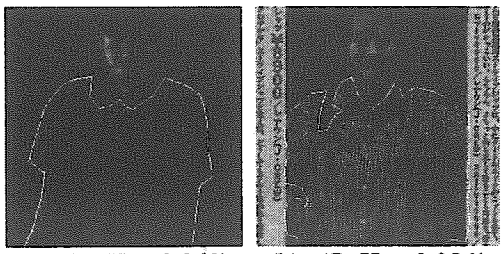
where $A(\cdot)$ denotes the area. The value of the error function for each segmentation result is given in Figs. 10-13. The average value of the error function in Figs. 10-13 is about 3.41%.

From these figures, some defects can be found in the segmentation results which still remain to be solved. First, when the clothing to be segmented and the background have similar hue attributes, some points on the background or on the clothing will be misclassified. Second, when there are serious shadows or highlights on the clothing, the points on which may also be misclassified. Finally, if there is a hole which is surrounded by the clothing region, it may be considered as an insignificant region and be reassigned to the class of the clothing. Figure 11(b) shows an example.



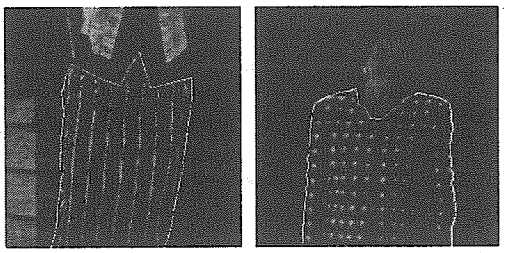
(a) $e(P, H) = 1.33\%$ (b) $e(P, H) = 3.18\%$

Fig. 10. Segmentation results of non-textured clothing.



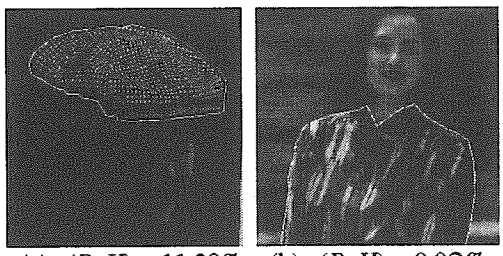
(a) $e(P, H) = 2.26\%$ (b) $e(P, H) = 3.83\%$

Fig. 11. Segmentation results of textured clothing.



(a) $e(P, H) = 3.06\%$ (b) $e(P, H) = 1.43\%$

Fig. 12. Segmentation results of clothing with larger pattern prints.



(a) $e(P, H) = 11.28\%$ (b) $e(P, H) = 0.92\%$

Fig. 13. Segmentation results of clothing with serious folds.

4 Conclusion

In this paper, we have proposed a color texture segmentation method for segmenting the clothing of interest from backgrounds. The texture features extracted based on FPSS is promising for segmentation. Texture characteristics in both the spatial and frequency domains can be captured adequately by FPSS. The hierarchical coarse-to-fine segmentation process shows its ability on noise reduction and boundary information preserving. The proposed approach is tolerant to shadows, highlights, folds and orientation variations on the textures and satisfactory experimental results are achieved.

Further research may be directed to the following topics. First, use more color information rather than only the hue attribute to improve the quantization effect. Second, find efficient algorithms for convolution operation and for solving eigensystem problem (compute the eigenvectors and eigenvalues of a matrix) to improve the computing speed. Third, achieve better performance by resolving the serious shadow/highlight problem.

References

[1] R. W. Conners and C. A. Harlow, "A Theoretical Comparison of Texture Algorithms," *IEEE Trans.*

on PAMI, 2(3), 204-222 (1980).

[2] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *IEEE Trans. on SMC*, 3(6), 610-621 (1973).

[3] D. C. He and L. Wang, "Unsupervised Textural Classification of Images Using the Texture Spectrum," *Pattern Recognition*, 25(3), 247-255 (1992).

[4] T. R. Reed and J. M. Hans du Buf, "A Review of Recent Texture Segmentation and Feature Extraction Techniques," *CVGIP: Image Understanding*, 57(3), 359-372 (1993).

[5] D. Slepian, "Prolate Spheroidal Wave Functions, Fourier Analysis, and Uncertainty - V: The Discrete Case," *The Bell System Technical Journal*, 57(5), 1371-1430 (1978).

[6] R. Wilson, "Finite Prolate Spheroidal Sequences and Their Application I: Generation and Properties," *IEEE Trans. on PAMI*, 9(6), 787-795 (1987).

[7] R. Wilson and M. Spann, "Finite Prolate Spheroidal Sequences and Their Application II: Image Feature Description and Segmentation," *IEEE Trans. on PAMI*, 10(2), 193-203 (1988).

[8] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel Texture Analysis Using Localized Spatial Filters," *IEEE Trans. on PAMI*, 12(1), 55-73 (1990).

[9] A. Teuner, O. Pichler, and B. J. Hosticka, "Unsupervised Texture Segmentation of Images Using Tuned Matched Gabor Filters," *IEEE Trans. on Image Processing*, 4(6), 863-870 (1995).

[10] D. Dunn and W. E. Higgins, "Optimal Gabor Filters for Texture Segmentation," *IEEE Trans. on Image Processing*, 4(7), 947-964 (1995).

[11] M. Unser, "Texture Classification and Segmentation Using Wavelet Frames," *IEEE Trans. on Image Processing*, 4(11), 1549-1560 (1995).

[12] Y. Ohta, T. Kanade, and T. Sakai, "Color Information for Region Segmentation," *Computer Graphics and Image Processing*, 13(2), 224-241 (1980).

[13] S. Tominaga, "Color Image Segmentation Using Three Perceptual Attributes," *Proc. of IEEE Intl. Conf. on Computer Vision and Pattern Recognition*, Miami Beach, Florida, 628-630 (1986).

[14] S. Tominaga, "Color Classification of Natural Color Images," *Color Research and Application*, 17(4), 230-239 (1992).

[15] A. Sarabi and J. K. Aggarwal, "Segmentation of Chromatic Images," *Pattern Recognition*, 13(6), 417-427 (1981).

[16] M. Celenk, "A Color Clustering Technique for Image Segmentation," *CVGIP*, 52(2), 145-170 (1990).

[17] J. M. Kasson and W. Plouffe, "An Analysis of Selected Computer Interchange Color Spaces," *ACM Trans. on Graphics*, 11(4), 373-405 (1992).

[18] C. L. Huang, T. Y. Cheng, and C. C. Chen, "Color Images' Segmentation Using Scale Space Filter and Markov Random Field," *Pattern Recognition*, 25(10), 1217-1229 (1992).

[19] Y. W. Lim and S. U. Lee, "On the Color Image Segmentation Algorithm Based on the Thresholding and the Fuzzy c-means Techniques," *Pattern Recognition*, 23(9), 935-952 (1990).

[20] J. Liu and Y. H. Yang, "Multiresolution Color Image Segmentation," *IEEE Trans. on PAMI*, 16(7), 689-700 (1994).

[21] D. K. Panjwani and G. Healey, "Unsupervised Segmentation of Textured Color Image Using Markov Random Field Models," *Proc. of Intl. IEEE Conf. on Computer Vision and Pattern Recognition*, New York, USA, 776-777 (1993).

[22] T. Caelli and D. Reye, "On the Classification of Image Regions by Colour Texture and Shape," *Pattern Recognition*, 26(4), 461-470 (1993).

[23] P. Schroeter and J. Bigün, "Hierarchical Image Segmentation by Multi-dimensional Clustering and Orientation-Adaptive Boundary Refinement," *Pattern Recognition*, 28(5), 695-709 (1995).

[24] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley (1992).

[25] B. Noble and J. W. Daniel, *Applied Linear Algebra, 3rd Edition*, Prentice-Hall International (1988).

[26] H. C. Lin, L. L. Wang, and S. N. Yang, "Automatic Determination of the Spread Parameter in Gaussian Smoothing," submitted for publication.

[27] Y. F. Li and D. C. Tseng, "Color image segmentation using circular histogram thresholding," *Proc. of IPPR Conf. on CVGIP*, Taiwan, R.O.C, 175-183 (1994).