International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:5, 2010

Image Segmentation Using the K-means Algorithm for Texture Features

Wan-Ting Lin, Chuen-Horng Lin, Tsung-Ho Wu, and Yung-Kuan Chan

***Abstract*—**This study aims to segment objects using the K-means algorithm for texture features. Firstly, the algorithm transforms color images into gray images. This paper describes a novel technique for the extraction of texture features in an image. Then, in a group of similar features, objects and backgrounds are differentiated by using the K-means algorithm. Finally, this paper proposes a new object segmentation algorithm using the morphological technique. The experiments described include the segmentation of single and multiple objects featured in this paper. The region of an object can be accurately segmented out. The results can help to perform image retrieval and analyze features of an object, as are shown in this paper.

***Keyword*—**k-mean, multiple objects, segmentation, texture features.

I. Introduction

IMAGE segmentation techniques play a key role and serve as the foundation for various types of image processing. In recent years, image segmentation has become an important topic in computer graphics: segmentation of an image into basic units that have certain meanings. These basic units are then further processed.

Since, for some images, objects and texture features obtained by segmentation are required features for image retrieval, image segmentation techniques will be very important in the future. Image segmentation techniques investigate the ways to accurately segment out the needed object. In past research, there were two types of texture segmentation: Feature-based segmentation and Model-based segmentation.

Accurately differentiating objects and backgrounds in digital images relies on the techniques of object shape segmentation [1]–[7].

Various image segmentation techniques have been proposed in the past. Usually these techniques are classified into two types according to the characteristics of the image's individual pixel value. The first method is called the 'discontinuous method'. It segments an image according to changes in pixel value. It distinguishes edges from the rest of the image by detecting sudden pixel value changes. Sudden changes appear in the borders of an image. Algorithms belonging to this type include: gradient method [3], Sobel edge detection [3], Canny edge detection [1, 2], Laplacian edge detection [3], and Laplacian Gaussian edge detection. The second method divides an image into similar regions according to a set of predefined rules. Images with prominent texture features, for example, are suitable for this type of method, as the texture image can be divided into several adequately similar regions according to textures. Algorithms of this type include: Threshold Method [3], Area Growing [4], Region Splitting and Merging [3] and Clustering [3].

The feature extraction method used in this study adopts the method proposed by Tamura et al. [8] to describe texture features. The theoretical basis of this method stems from the psychological studies on human vision. Tamura proposes six features that correspond to the texture feature properties in psychological perspective: coarseness, contrast, directionality, linelikeness, regularity, and roughness. Among these, coarseness, contrast and directionality are the key determinants for image retrieval [9]. Hence, many researchers adopt Tamura's methods for the application of image retrieval. In this paper, extracted texture features serve as the feature variable. Image textures are segmented through clustering.

II. Texture Feature of an Image

The algorithm in this study pre-processes images in order to decrease the number of colors in the image. The feature extraction method used in this study incorporates the method proposed by Tamura et al. [8] to describe texture features.

A. Color Space Conversion

In this paper, the YC_bC_r domain is selected as the operation space. The RGB value of the pixel (x, y) is converted into the YC_bC_r domain through the following formula:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.209 & 0.587 & 0.114 \\ -0.168 & -0.331 & 0.5 \\ 0.5 & -0.418 & -0.081 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}. \quad (1)$$

Wan-Ting Lin is with the Graduate School of Computer Science and Information Technology, National Taichung Institute of Technology, No. 129, Sec. 3, Sanmin Rd., Taichung, Taiwan, R.O.C. (corresponding author to provide phone: 886-4-22196348; e-mail: luckyday100@ gmail.com).

Chuen-Horng Lin is with the Graduate School of Computer Science and Information Technology, National Taichung Institute of Technology, No. 129, Sec. 3, Sanmin Rd., Taichung, Taiwan, R.O.C. (e-mail: linch@ ntit.edu.tw).

Tsung-Ho Wu is with the Department of Computer Science and Engineering, National Chung Hsing University, No. 250, Kuokuang Rd., Taichung, Taiwan, R.O.C. (e-mail: phd9801@ cs.nchu.edu.tw).

Yung-Kuan Chan is with the Management Information Systems, National Chung Hsing University, No. 250, Kuokuang Rd., Taichung, Taiwan, R.O.C (e-mail: ykchan@ nchu.edu.tw).

The formula $f(x,y)=Y(x,y)$ is used to create a grayscale image from the original image, as shown in Fig. 2.


Fig. 1: Original image


Fig. 2: Grayscale image

B. Texture analysis

As mentioned above, coarseness, contrast and directionality have considerable effects on image analysis.

a. Coarseness

Coarseness is the degree of coarseness of the image. In this study, the scale $k=1\sim5$ is used to indicate the coarseness of each pixel. A smaller value indicates a higher degree of coarseness. In other words, when the coarseness of an image is small, the image has more coarse textures.

Coarseness can be calculated through the following procedure. First, a block adjacent to the pixel (x,y) is selected and the average pixel value $A_k(x,y)$ of the block is calculated using the following equation:

$$A_k(x,y)=\frac{1}{2^{2k}}\sum_{i=-2^{k-1}}^{2^{k-1}-1}\sum_{j=-2^{k-1}}^{2^{k-1}-1}f(x+i,y+j)\ , \tag{2}$$

where $k=1\sim5$ and $f(x+i,y+j)$ is the gray-level value of $(x+i,y+j)$. When $k=1$, a 2×2 block is selected to calculate the average pixel value $A_1(x,y)$ of this block. When $k=2$, a 4×4 block is used to calculate the average pixel value $A_2(x,y)$ of the block. This operation is applied to calculate $A_3(x,y)\sim A_5(x,y)$ for $k=3,\ 4,\ 5$.

Later, based on each k value, the average gray-level value differences between each pixel and its neighboring pixels (in horizontal and vertical directions) can be derived as follows:

$$E_{k,h}(x,y)=\left|A_k(x+2^{k-1},y)-A_k(x-2^{k-1},y)\right|, \tag{3a}$$

$$E_{k,v}(x,y)=\left|A_k(x,y+2^{k-1})-A_k(x,y-2^{k-1})\right|. \tag{3b}$$

After the average gray level difference between each pixel (x,y) when $k=1\sim5$ in horizontal and vertical directions, $E_{k,h}(x,y)$ and $E_{k,v}(x,y)$ are derived, the maximum value E and the corresponding k are used to represent the coarseness of the pixel (x,y), denoted as $F_{coarse}(x,y)$:

$$F_{coarse}(x,y)=\max\{E_{k,i}(x,y),i=h \text{ and } v,k=1,2,..,5\}. \tag{4}$$

The results are shown in Fig. 3.

b. Contrast

Contrast refers to the level of contrast between image colors. A larger contrast value indicates higher and more evident variations between colors.

In the calculation of contrast, this study uses a 3×3 block around each pixel (x,y) to estimate the contrast level of the pixel, as shown below:

$$F_{con}(x,y)=\frac{\sigma(x,y)}{[\eta(x,y)/\sigma^4(x,y)]^{1/4}}, \tag{5}$$

where $\mu(x,y)$ denotes the mean of each block; $\sigma(x,y)$ is the standard deviation, and $\eta(x,y)$ is the fourth moment about the mean. Each can be presented as follows:

$$\mu(x,y)=\frac{1}{9}\sum_{j=-1}^{1}\sum_{i=-1}^{1}f(x+i,y+j), \tag{6a}$$

$$\sigma(x,y)=\sqrt{\frac{1}{9}\sum_{j=-1}^{1}\sum_{i=-1}^{1}[f(x+i,y+j)-\mu(x,y)]^2}, \tag{6b}$$

$$\eta(x,y)=\frac{1}{9}\sum_{j=-1}^{1}\sum_{i=-1}^{1}[f(x+i,y+j)-\mu(x,y)]^4. \tag{6c}$$

After the contrast value is normalized, a contrast image can be described with 0–255 gray levels, as shown in Fig. 4. In this figure, lower contrast has a smaller value and is represented by the black color; higher contrast has a larger value and is represented by nearly-white colors.

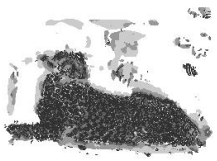


Fig. 3: Coarseness image


Fig. 4: Contrast image

c. Directionality

Directionality can generally describe the degree of directional variation of a certain color. A larger value of directionality indicates a higher degree of directional variation of the color.

This study uses the gradient magnitude $\Delta(x,y)$ to calculate the gradient angle $\theta(x,y)$, as shown in the following equation:

$$\Delta(x,y)=\sqrt{\Delta_x^2(x,y)+\Delta_y^2(x,y)}, \tag{7}$$

$$\theta(x,y)=\tan^{-1}\left(\frac{\Delta_y(x,y)}{\Delta_x(x,y)}\right)+\pi, \tag{8}$$

where Δ_x and Δ_y are the edge variation in the horizontal

direction and in the vertical direction, respectively.

With $\Delta(x,y)$ and $\theta(x,y)$, the average gradient angle $\bar{\theta}(x,y)$ of the pixel (x,y) and its neighboring pixels can be calculated. A 3×3 block is used in this study. The equation is:

$$\bar{\theta}(x,y)=\frac{1}{9}\sum_{j=-1}^{1}\sum_{i=-1}^{1}\theta(x+i,y+j)\cdot \tag{9}$$

To calculate directionality, the gradient angle $\theta(x,y)$ of the gradient magnitude $\Delta(x,y)$ greater than the threshold T_θ is adopted. The threshold T_θ can be defined by users according to different image features. In this study, the threshold value is defined as the average gradient magnitude $\Delta(x,y)$ of the entire image. Thus, the gradient angle $\theta(x,y)$ for the calculation of directionality can be expressed as follows:

$$\theta(x,y)=\begin{cases}\theta, & \Delta\geq T_\theta\\ 0, & \Delta<T_\theta\end{cases}. \tag{10}$$

Finally, the difference between $\theta(x,y)$ and $\bar{\theta}(x,y)$ is calculated and divided by 9 to derive the directionality. The directionality $F_{dir}(x,y)$ of the pixel (x,y) can be expressed as:

$$F_{dir}(x,y)=\frac{1}{9}[\theta(x,y)-\bar{\theta}(x,y)]. \tag{11}$$

After directionality $F_{dir}(x,y)$ is normalized, directionality can be described with gray levels 0–255 in an image, as shown in Fig. 5. The black region indicates smaller color variations, while the nearly-white regions indicate larger color variations.

III. Image Segmentation

The algorithm segments an image according to similar features among pixels. The first step extracts texture features of an image. The second step performs the K-means algorithm for clustering. Lastly, image segmentation is achieved when the grouped regions are further processed by adopting methods from morphology.

A. Multiple Features Classification

In consideration of the fact that these three features have different degrees of influence, the algorithm multiplies feature value by a scaling factor in this paper, as shown in the following equation:

$$\begin{aligned}\bar{F}_1(x,y)&=w_1F_{coarse}(x,y)\\ \bar{F}_2(x,y)&=w_2F_{con}(x,y)\\ \bar{F}_3(x,y)&=w_3F_{dir}(x,y)\end{aligned}, \tag{12}$$

The algorithm then calculates the mean filter of each weighted value by using 9×9 blocks. The result serves as the basis for data grouping. The K-means algorithm is as follows:

Step 1: Select a value for c.
Step 2: Select the cluster center.
Step 3: Determine the features for every pixel (mean feature $\bar{F}_i$).
Step 4: Define a similarity measure between the cluster center and features (Euclidean Distance).
Step 5: Calculate the cluster center of the new cluster.
Step 6: Repeat steps 4 and 5 until the cluster center stops changing.

The algorithm divides the coarseness, contrast and directionality of each pixel into two groups (0 or 1) by using the K-means algorithms expressed in this paper. The result is then processed by using the method of erosion in morphology. The processed image is shown in Fig. 6.

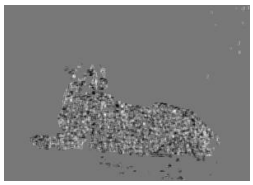


Fig. 5: Directionality image

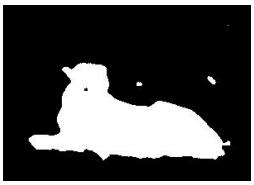


Fig. 6: Result after K-means clustering

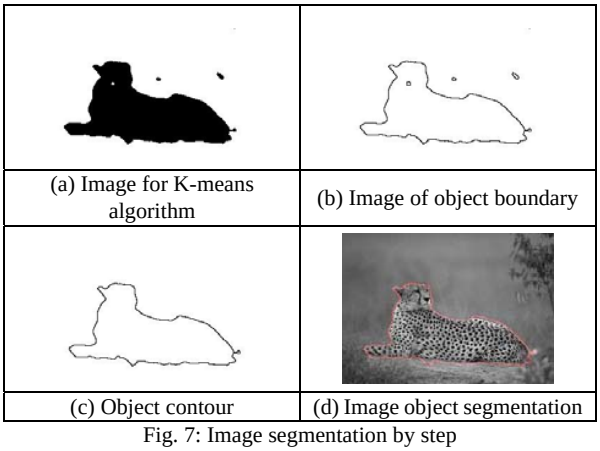
B. Object Segmentation

The image has been segmented using the K-means algorithm. The clustering can be improved by assuming that neighboring pixels have a high probability of falling into the same cluster. In image segmentation application, the observations are based on the pixels in the image plane. Therefore, after K-means segmentation, the image is segmented into K non-continuous regions. The K-means segmentation algorithm is as follows:

Step 1: Select a value for K.
Step 2: Apply the K-means Algorithm.
Step 3: Apply the Connected Components Algorithm.
Step 4: Merge any components of size less than a given threshold to an adjacent component.
Step 5: Segmentation of the object and the background.

IV. Experimental Results

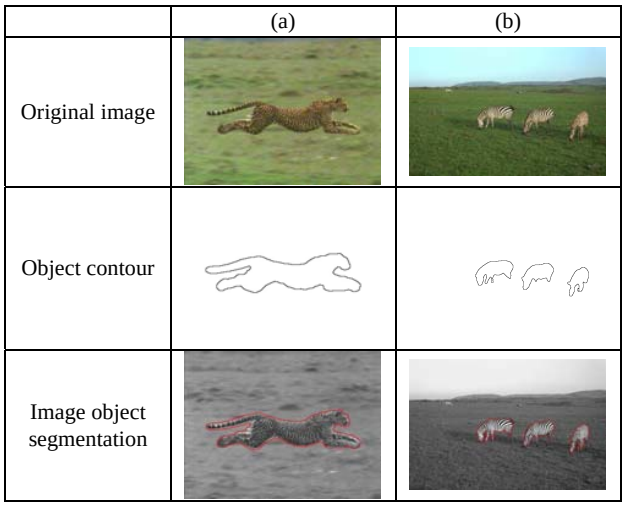
The experiment uses the result of K-means clustering in this paper. First, the resulting image (Fig. 6) is complemented and a new image obtained, as shown in Fig. 7(a). Then the area size of every closed region is calculated, as shown in Fig. 7(b). The objects which are smaller in size than the threshold value (in this case, $T_A=51\times51$ pixel) are removed, as shown in Fig. 7(c). Finally, the contours of every object are treated as edges and segmented. The result is shown in Fig. 7(d).



(a) Image for K-means algorithm | (b) Image of object boundary
(c) Object contour | (d) Image object segmentation

Fig. 7: Image segmentation by step

The next experiment is on an image of a leopard running across a field, as shown in Fig. 8(a). The algorithm segments the leopard out from the image. The result is shown in Fig. 8(a). Although the edges are not prominent, due to the fast movement of the leopard, this experiment indicates that the algorithm is able to accurately segment out the animal.



	(a)	(b)
Original image		
Object contour		
Image object segmentation		

Fig. 8: Image segmentation for single and multiple objects

Finally, this paper proceeds to the multiple object segmentation experiment. The original image is shown in Fig. 8(b). The result is also shown in Fig. 8(b). The experiment indicates that the algorithm is able to separate the animal from the background even when the edges of the animal are blurred due to uneven textures and shadows.

V. Conclusions

The algorithm extract texture features of an image and regions with similar texture features are grouped together to obtain the shape of the object for the purpose of segmentation.

The experiment's results indicate that the algorithm is able to accurately segment out the regions belonging to the object in an image under various inferior conditions. Such conditions include: the case where the edges of even textured objects are not prominent due to fast movement, the case where background color and object color are similar and difficult to distinguish, and the case where the edges of objects are blurry due to shadows.

In addition to accurately and completely segmenting out an object, the algorithm in this paper records the texture features of this object. In the future, the properties of that object can be analyzed by an analysis of its texture features.

References

[1] J. F. Canny, "A Computational Approach to Edge Detection," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, 1986, pp. 679–698.
[2] L. Ding and A. Goshtasby, "On the Canny Edge Detector," Pattern Recognition, Vol. 34, 2001, pp. 721–725.
[3] Rafael C. Gonzalez, and Richard E. Woods, "Digital Image Processing", Prentice-Hall, 2002.
[4] Pellegrino, F.A. Vanzella, W. Torre, and V., "Edge Detection Revisited," IEEE transactions on systems, man, and cybernetics-part B: CYBERNETICS, Vol. 34, NO. 3, 2004, pp.1500–1518.
[5] Z. Hou, Q. Hu and W. L. Nowinski, "On minimum variance thresholding," Pattern Recognition Letters, Vol. 27, 2006, pp. 1732–1743.
[6] F. Y. Shih and S. Cheng, "Automatic seeded region growing for color image segmentation," Image and Vision Computing, Vol. 23, 2005, pp. 877–886.
[7] Mmford, D. and Shah, J., "Optimal approximations by piecewise smooth function and associated variational problems," Commun.Pure Appl. Math.42, 1898, pp.577–684.
[8] H. Tamura, S. Mori, and T. Yamawaki, "Texture features corresponding to visual perception," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 8, 1978, pp. 460–473.
[9] H. C. Lin, C. Y. Chiu, and S. N. Yang, "Finding textures by textual descriptions, visual examples, and relevance feedbacks," Pattern Recognition Letters, vol. 24, 2003, pp. 2255–2267.