

Orthogonal moment based texture segmentation

Xiao Hua Shu Huazhong Yu Wenxue Li Songyi

(Department of Biological Science and Medical Engineering, Southeast University, Nanjing 210096, China)

Abstract: Texture segmentation is a necessary step to identify the surface or an object in an image. We present a Legendre moment based segmentation algorithm. The Legendre moments in small local windows of the image are computed first and a nonlinear transducer is used to map the moments to texture features and these features are used to construct feature vectors used as input data. Then an RBF neural network is used to perform segmentation. A k -mean algorithm is used to train the hidden layers of the RBF neural network. The training of the output layer is the supervised algorithm based on LMS. The algorithm has been successfully used to segment a number of gray level texture images. Compared with the geometric moment-based texture segmentation, we can reduce the error rates using orthogonal moments.

Key words: Legendre moment; texture; radial basis function neural network

In texture analysis, an important and difficult problem is texture segmentation. Texture segmentation is dividing the textures with different properties into regions which are not intersectant so that the textural properties in one region are more similar while the textural properties in different regions are more different.

Texture analysis has been studied for a long time using various approaches. Various methods are used to perform texture analysis directly upon the gray levels in an image. These include gray level co-occurrence matrix (GLCM)^[1], autocorrelation function analysis^[1], generalized co-occurrence matrices (GCM)^[2], second order spatial averages^[3], and two-dimensional filtering in the spatial and frequency domain^[4-7]. The computation of features that capture textural properties is at the heart of most of these approaches. With the development of the moments, they have been used to extract the textural properties. Tuceryan proposed a geometric-moment-based texture segmentation algorithm^[8]. Because the orthogonal moments have many nice properties, we proposed an orthogonal moment-based texture segmentation. We obtained texture features directly from the gray-level image by computing the orthogonal moments of the image in local regions. An RBF neural network was used to perform segmentation. Section 1 gives the details of the algorithm and experimental results. Section 2 makes some concluding remarks.

Received 2002-08-21.

Foundation item: The National Natural Science Foundation of China (60272045).

Biographies: Xiao Hua (1978—), female, graduate; Shu Huazhong (corresponding author), male, professor, shu.list@seu.edu.cn.

1 Method

The steps of orthogonal moment based texture segmentation are:

- 1) Compute the image moments within a small window around each pixel;
- 2) Get images using these moments and compute the mean value within a small window;
- 3) Compute the texture features by applying a nonlinear transformation;
- 4) Train RBF neural network;
- 5) Segment the image;
- 6) Display the result.

The algorithm is given in Fig.1.

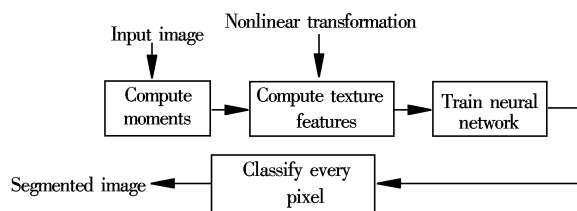


Fig.1 The flowchart of the segmentation algorithm

The details of each step are described as follows.

1.1 Moments and nonlinear transformation

We used Legendre moments to extract the properties. Concretely, the $(p + q)$ -th order Legendre moments of a two-dimensional image are defined as

$$L_{pq} = \frac{(2p + 1)(2q + 1)}{4} \cdot \sum_{i=1}^N \sum_{j=1}^N P_p(x_i) P_q(y_j) f(x_i, y_j)$$

where the n -th order Legendre polynomial is given by

$$P_n(x) = \frac{1}{2^n} \sum_{m=0}^{n/2} (-1)^m \frac{(2n-2m)!}{m!(n-m)!(n-2m)!} x^{n-2m}$$

1.1.1 Feature extraction

The properties were extracted by computing the orthogonal moments of the image. The moments were computed within small local windows around each pixel. Given a window size, the coordinates were normalized to the range $[-1, 1]$, the pixel being at the origin. This permitted us to compare the set of moments computed for each pixel.

Let W be the window width. We always choose the window width W to be odd so that the pixel (i, j) is centered on a grid point. Let (i, j) be the pixel coordinates for which the moments are computed. For a pixel with coordinates (m, n) which fall within the window, the normalized coordinates (x_m, y_n) are given by

$$x_m = \frac{m-i}{\lceil W/2 \rceil}, \quad y_n = \frac{n-j}{\lceil W/2 \rceil}$$

Then the moments are

$$L_{pq} = \frac{(2p+1)(2q+1)}{4} \sum_{m=-W/2}^{W/2} \sum_{n=-W/2}^{W/2} P_p(x_m) P_q(y_n) \cdot f(m, n)$$

Here the size of the window width is very important. As the window size gets larger, more global features are detected. This suggests that the choice of the window size could possibly be tied to the contents of the image. The images with larger texture tokens require larger window sizes whereas finer textures require smaller windows. Here we regard the window size as a space parameter.

1.1.2 Nonlinear transformation

The moments alone are not sufficient to obtain good texture features in certain images. One solution suggested by Caelli is to introduce a nonlinear transducer that maps moments to texture features^[9].

We obtained the texture feature image F_k corresponding to the moment image M_k with mean \bar{M} using the following transformation:

$$F_k(i, j) = \frac{1}{L^2} \sum_{(a, b) \in W_{ij}} \left| \tanh[\sigma(M_k(a, b) - \bar{M}_k(a, b))] \right|$$

where W_{ij} is an $L \times L$ averaging window centered at location (i, j) ; σ controls the shape of the logistic function. We have determined by trial and error the value of σ to be 0.01. The original image is shown in Fig.2. Fig.3 shows its moment images and Fig.4 shows the computed texture feature images after the nonlinear transducer stage. Fig.5 shows gray histograms of the images in Fig.4.

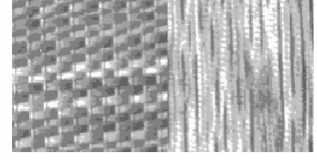


Fig.2 The original

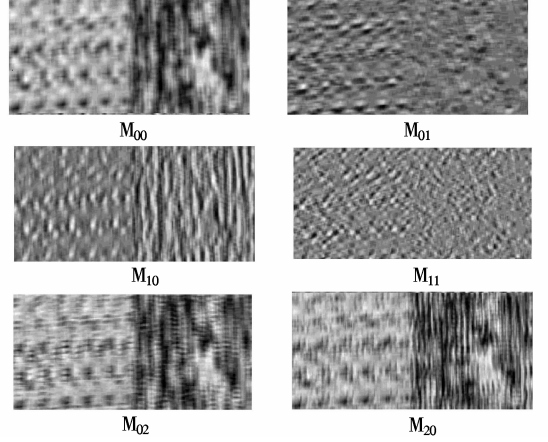


Fig.3 The moment images with the moment mask size 9

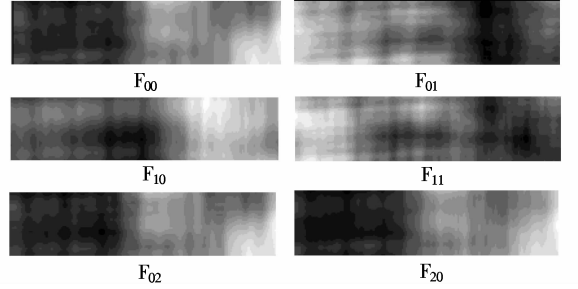


Fig.4 The feature images with the average window of size 49

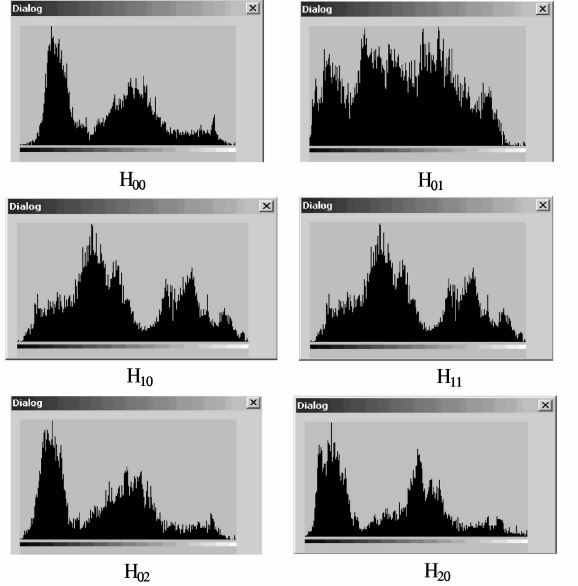


Fig.5 Gray histograms

1.2 Segmentation

1.2.1 RBF neural network

The RBF neural network is used here. The nerve cell of the RBF is defined as

$$u_i = g \left[\sum_{j=1}^N \left(\frac{x_j - w_{ij}}{\sigma_i} \right)^2 \right] \quad i = 1, 2, \dots, s_1$$

where s_1 is the number of the hidden layer; N is the number of the dimension of the vector space; x_j is the vector which was computed by our algorithm; w_{ij} is the center of the nerve cell of the RBF; σ_i is the scaling coefficient usually called the radius or width. The inspiring function of the nerve cell usually takes Gauss function for representing and g represents Gauss function.

The output layer of the RBF is a linear layer defined as

$$y_t = \sum_{i=1}^{s_1} v_{ti} u_i \quad t = 1, 2, \dots, M$$

where M is the number of the output layer; y_t is the output of the t -th nerve cell; v_{ti} is its node weight; u_i is the output of the i -th hidden layer.

1.2.2 Training algorithm of the RBF

As the RBF has the characteristic of local clustering, the nonsupervised algorithm was applied to its hidden layers and the supervised algorithm was applied to its output layers^[10].

1) The algorithm of the hidden layers — k -mean algorithm

Its details are:

- ① Initialize the clustering center \mathbf{w}_j ($j = 1, 2, \dots, s_1$);
- ② Search for the clustering center \mathbf{w}_j^* which is the nearest to \mathbf{x}_i ;
- ③ Include \mathbf{x}_i into the clustering C_j^* which is represented by \mathbf{w}_j^* ;
- ④ Take the mean of all the swatches in the clustering C_j as the new center. It is defined as

$$\mathbf{w}_j = \frac{1}{M_j} \sum_{\mathbf{x}_i \in C_j} \mathbf{x}_i$$

where M_j is the number of swatches;

- ⑤ Change the center until the clustering situation does not change any more.

The parameter σ_i was defined after k -mean algorithm. It was defined as

$$\sigma_i^2 = \frac{1}{M_j} \sum_{\mathbf{x}_i \in C_j} \|\mathbf{x}_i - \mathbf{w}_j\|^2$$

2) The algorithm of the output layers

The training of the output layer is the supervised algorithm based on LMS. The change of weight after each iteration should be proportional to the value of the negative gradient. It is defined as

$$\Delta \mathbf{v}_t = -\alpha \frac{\partial E}{\partial \mathbf{V}_t} = \alpha \sum_{k=1}^Q \mathbf{u}^k (y_t^k - c_t^k)$$

where Q is the number of the swatch; y_t^k is the expected output of the k -th swatch of the t -th output cell; c_t^k is the real output; \mathbf{u}^k is the vector of the output of the hidden layers.

In the experiment, α is 0.0001.

The result after the output layers is shown in Fig. 6(a). And the result using geometric moments is shown in Fig. 6(b). The error rate in Fig. 6(a) is 1.7750%. And the error rate in Fig. 6(b) is 1.9792%.

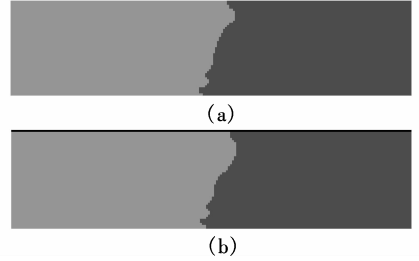


Fig. 6 The results. (a) The segmentation result using orthogonal moments; (b) The segmentation result using geometric moments

2 Discussion and Conclusion

We provided a method to segment the textures based on orthogonal moment here and we used RBF neural network to finish the segmentation.

When we extracted the features, we computed the orthogonal moments within a small window. Then we used a nonlinear transformation to get more information of the textures. The results of the experiment show that it is successful to extract the texture features using our algorithm. And it is better than the method using geometric moments to extract the texture features.

In the experiment we find that the window width is important. The images with larger texture tokens would require larger window sizes whereas finer textures would require smaller windows.

The result shows that our method is effective. The future work is to consider the influence of the orders of the moments.

References

- [1] Haralick R M. Statistical and structural approaches to texture [J]. *Proceedings of the IEEE*, 1979, **67**(5):786 – 804.
- [2] Davis L S, Clearman M, Aggarwal J K. An empirical evaluation of generalized cooccurrence matrices[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1981, **3** (2):214 – 221.
- [3] Gagalowicz A. Blind texture segmentation[A]. In: *Proc of the 9th International Conference on Pattern Recognition* [C].

Rome, Italy, 1988. 46 – 50.

[4] Coggins J M, Jain A K. A spatial filtering approach to texture analysis[J]. *Pattern Recognition Letters*, May, 1985, PRL(3):195 – 203.

[5] Clark M, Bovik A C, Geisler W S. Texture segmentation using Gabor modulation/demodulation[J]. *Pattern Recognition Letters*, Sept, 1987, PRL(6):261 – 267.

[6] Turner M R. Texture discrimination by Gabor functions[J]. *Biological Cybernetics*, 1986, 55:71 – 82.

[7] Voorhees H, Poggio T. Detecting textons and texture boundaries in natural images[A]. In: *Proc of the First International Conference on Computer Vision* [C]. London, 1987. 250 – 258.

[8] Mihran Tuceryan. Moment based texture segmentation[J]. *Pattern Recognition Letters*, July, 1994, PRL(15):659 – 668.

[9] Caelli T, Oguztoreli M N. Some tasks and signal dependent rules for spatial vision[J]. *Spatial Vision*, 1987, 2: 295 – 315.

[10] Moody J, Darken C J. Fast learning in networks of locally-tuned processing units[J]. *Neural Computation*, 1989, 1: 281 – 293.

基于正交矩的纹理分割

肖 华 舒华忠 於文雪 李松毅

(东南大学生物科学与医学工程系, 南京 210096)

摘 要 在识别一幅图像中的界面或者物体时,一般先要进行纹理分割.本文提出了基于勒让得矩的纹理分割方法.首先在图像的小窗口中计算矩值,然后用一个非线性转换器把它转化成纹理特征.再用这些特征组成特征向量作为输入数据.接着采用 RBF 人工神经网络对提取的特征进行分割.用 k 均值算法训练 RBF 人工神经网络的隐层.输出层的训练是采用基于 LMS 的监督式数学模型.该算法成功地分割了许多灰度级图像.和基于几何矩的纹理分割相比,用正交矩可以降低分割错误率.

关键词 勒让得矩; 纹理; RBF 人工神经网络

中图分类号 TP391.4