Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (5): 772-779, 2008
ISSN 1812-5654

Image Thresholding Using Weighted Parzen-Window Estimation

[1,2]Wang Jun and [1]Wang Shitong
[1]School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing, JiangSu, China
[2]School of Information Technology, Jiangnan University, WuXi, JiangSu, China

Abstract: A novel image thresholding method based on weighted Parzen-window estimation is proposed in this paper. A hierarchical clustering procedure is first performed to obtain the reference pixels and weights before the weighted Parzen-window procedure is used to estimate the corresponding probabilities. The error produced during reference pixels' generation is controlled by the upper bound error. Using the proposed criterion function, the optimal threshold is computed. Compared with the image thresholding method based on Parzen-window estimation. The experimental results here show that the proposed method can effectively reduce the computational burden and storage requirements without degrading final segmentation results a lot.

Key words: Non-parametric classifiers, Parzen-windows, clustering, image processing, training samples

INTRODUCTION

The problem of image segmentation throws great challenges for pattern recognition and image processing community. Image thresholding is the simplest technique for image segmentation and its primary task is to find the optimal threshold which could separate objects from background well.

With the advantage of simple and easy implementation, the global thresholding is still a popular technique over the past years. Several successful thresholding methods such as OTSU methods, histogram-based methods, entropy-based method become popular (Sezgin and Sankur, 2004; Zhang, 2001; de Albuquerque *et al*., 2004; Sarkar and Soundararajan, 2000; Yang and Yang, 2004). Considering the analogy between clustering and image segmentation, some clustering procedures based on pdf estimation have been successfully applied to image segmentation. Recently, a novel thresholding method based on Parzen-window estimation (i.e., PWT) has been proposed by Wang *et al*. (2008). This method effectively integrates spatial information on pixels of different gray levels into the process pdf estimation which also the base of our work here.

In this study, the concept of weighted Parzen-window estimation is presented and a novel image thresholding method based on this concept is accordingly proposed. By combining several pixels which are close to each other, a reference pixel is generated and it will be used in the weighted Parzen-window estimation procedure. In this way, the number of the samples used in Parzen-window estimation decreases greatly, which may further reduce the computational burden and storage requirements of image thresholding.

IMAGE THRESHOLDING USING PARZEN-WINDOW ESTIMATION

With excellent performance and solid theoretical foundation, the Parzen-window estimation is a well-known non-parametric approach for probability estimation. Here, we state a novel thresholding algorithm based on Parzen-window technique in Wang *et al*. (2008) as follows:

Assuming f (x, y) is the gray level of the pixel with its location (x, y) in image, an m×n sized image with L gray levels could be expressed as $\{f(x, y) \mid x \in \{1, 2, ..., m\}, y \in \{1, 2, ..., n\}\}$. Let $\omega_i = \{(x, y) \mid f(x, y) = i, x \in \{1, 2, ..., m\}, y \in \{1, 2, ..., n\}, i \in \{1, 2, ..., L\}\}$ and C_i denotes the number of pixels in ω_i, $i = 1, 2..., L$, we have $\{f(x, y) \mid x \in \{1, 2, ..., n\}\}$, then

$$N = \sum_{i=1}^{L} C_i$$

Obviously, ω_i is defined in a two-dimensional space. According to the Parzen-window estimation, for the point space ω_i, the probability density function $p(x, y, \omega_i)$ can be approximated using the following formulas:

$$p(x, y \mid \omega_i) = \frac{1}{C_i} \sum_{j=1}^{C_i} \frac{1}{V_{C_i}} \varphi(x, y; x_j, y_j; h_{C_i}^2) \quad (1)$$

Corresponding Author: Wang Shitong, School of Information Technology, Jiangnan University, WuXi, JiangSu, China
Tel: +86 510 89512151

$$p(x,y,\omega_i) = p(\omega_i)p(x,y \mid \omega_i)$$
$$= \frac{C_i}{N} \cdot \frac{1}{C_i} \sum_{j=1}^{C_i} \frac{1}{V_{C_i}} \varphi(x,y;x_j,y_j;h_{C_i}^2) \quad (2)$$
$$= \frac{1}{N} \sum_{j=1}^{C_i} \frac{1}{V_{C_i}} \varphi(x,y;x_j,y_j;h_{C_i}^2)$$

Where:

hc_i = Window width of the ith gray level set
(x_j, y_j) = The jth pixel in ω_i,
Vc_i = The volume of the corresponding hypercube with h_{ci} as its length of each edge, i.e., $Vc_i = Hc_i^2$ for an image and

$$h_{C_i} = \left(\frac{1}{C_i}\right)^{\frac{1}{4}}$$

is a good choice; $p(\omega_i)$ can be approximated by C_i/N, $1 \le i \le L$ and $\varphi(\cdot)$ is a kernel function.

Because of the continuous distribution of pixels with same gray level in images, a pixel's possibility of having gray level i is in inverse proportion to its Euclidean distance to other pixels whose gray levels are i. The following Gaussian function for $\varphi(\cdot)$ reflects this fact well:

$$\varphi(x,y;x_j,y_j;h_{C_n}^2) = \frac{1}{2\pi} \exp\left(-\frac{(x-x_j)^2+(y-y_j)^2}{2h_{C_i}^2}\right) \quad (3)$$

Let p(x, y) be the probability density of pixel (x, y) belonging to the background class and r(x, y) be the probability density of pixel (x, y) belonging to the object, we have:

$$p(x,y) = \sum_{i=1}^{t} p(x,y,\omega_i) = \sum_{i=1}^{t} p(\omega_i)p(x,y \mid \omega_i)$$
$$= \frac{1}{N} \sum_{i=1}^{t} \sum_{j=1}^{C_i} \frac{1}{V_{C_i}} \varphi(x,y;x_j,y_j;h_{C_i}^2) \quad (4)$$
$$= \frac{1}{2\pi N} \sum_{i=1}^{t} \sum_{j=1}^{C_i} \frac{1}{h_{C_i}^2} \exp\left(-\frac{(x-x_j)^2+(y-y_j)^2}{2h_{C_i}^2}\right)$$

$$r(x,y) = \sum_{i=t+1}^{L} p(x,y,\omega_i) = \sum_{i=t+1}^{L} p(\omega_i)p(x,y \mid \omega_i)$$
$$= \frac{1}{N} \sum_{i=t+1}^{L} \sum_{j=1}^{C_i} \frac{1}{V_{C_i}} \varphi(x,y;x_j,y_j;h_{C_i}^2) \quad (5)$$
$$= \frac{1}{2\pi N} \sum_{i=t+1}^{L} \sum_{j=1}^{C_i} \frac{1}{h_{C_i}^2} \exp\left(-\frac{(x-x_j)^2+(y-y_j)^2}{2h_{C_i}^2}\right)$$

According to the concept of entropy in information theory, the entropy of pixel (x, y) could be computed as follows:

$$H(x,y) = p(x,y)\log p(x,y) + r(x,y)\log r(x,y) \quad (6)$$

Obviously, H(x, y) is maximized when p(x, y) = r(x, y) is satisfied. It implies that the smaller |p (x, y)-r(x, y)| takes, the larger H(x, y) gets. Therefore, In order to make class p and class r well separated, the following criterion function was used:

$$t_{opt} = \arg\min_t \iint_\Omega (p(x,y) - r(x,y))^2 \, dxdy$$
$$= \arg\min_t \iint_\Omega p^2(x,y)dxdy + \quad (7)$$
$$\iint_\Omega r^2(x,y)dxdy - 2\iint_\Omega p(x,y)r(x,y)dxdy$$

Where, $\Omega = [1, m] \times [1, n]$

The Parzen-window estimation for p(x, y) and r(x, y) reflects the fact that the pixel's possibility of being gray level i is determined by its distance to other pixels with same gray level i. However, in order to compute p(x, y) and r(x, y) for a pixel at (x, y) in an image, all the pixels except itself will be considered and this will bring very big computational burden.

WEIGHTED PARZEN-WINDOW METHOD FOR IMAGE THREDSHOLDING

In order to accelerate the PWT procedure and reduce the computational burden, we will derive its enhanced version i.e., image thresholding WPWT based on weighted Parzen-window estimation. In the proposed method WPWT, a training procedure is performed to obtain a set of the reference pixels and weights at each gray level, which undoubtly yields errors into the computation of t_{opt}. However, by controlling the upper bound of errors we can make thresholding results acceptable.

The set of the reference pixels at gray level i is denoted as:

$$R_i = \{r_{i1}, r_{i2}, \ldots, r_{im}\}, r_i = \{x_{ij}^{(r)}, y_{ij}^{(r)}), j = 1,2,\ldots m$$
$$x_{ij}^{(r)} \in \{1,2,\ldots,m\}, y_{ij}^{(r)} \in \{1,2,\ldots,n\}, i \in \{1,2,\ldots,L\}\}$$

Accordingly, the set of the weights for the reference pixels at gray level i is denoted as:

$$W_i = \{w_{i1}, w_{i2}, \ldots, w_{im}\}, m \le n$$

So, as an approximation of (1), p (w, y | ω_i) in (1) can be computed as:

$$p(x,y \mid \omega_i) = \frac{1}{C_i} \sum_{j=1}^{m_i} \frac{w_{ij}}{V_{C_i}} \varphi(x,y;x_{ij}^{(r)}, y_{ij}^{(r)};h_{C_i}^2) \quad (8)$$

Where:

w_{ij} = Weight of the jth reference pixel at gray level i, which is equal to the number of the original training samples that were collapsed into $(x_{ij}^{(r)}, y_{ij}^{(r)})$

The training procedure for a single gray-level: In order to find a set of appropriate reference pixels for effective weighted Parzen-window estimation, we derive a training procedure based on the idea of hierarchical clustering. We state the training procedure as the following pseudo-code:

n : Number of the pixels at each gray level
m : Number of the reference pixels at each gray level
R_i : Vector of the reference pixels at gray level i
r_{ij} : The jth reference pixel of gray level i
W_i : Vector of weights for the reference pixels at gray level i
w_{ij} : Weight of the jth reference pixel at gray level i
DM : Distance matrix of the reference pixels at each gray level

For the above training procedure, we give more details as follows.

This procedure initializes R_i with the pixels of gray level i in the image. Then R_i is updated by combining two nearest pixels until the error e exceeds e_{max} or only one reference pixel remains.

The sub-routine getNearestPair () returns two nearest pairs r_1, r_2 and store their weights into w_1 and w_2, respectively. Then r_1, r_2 are merged in the sub-routine mergePair ().

The sub-routine mergePair () involves the following key issues. Firstly,

$$r_0 = \frac{\omega_1 r_1 + \omega_2 r_2}{\omega_1 + \omega_2} \text{ and } \omega_0 = \omega_1 + \omega_j$$

are inserted into R_i and W_i, respectively. Meanwhile, r_1, r_2 are removed from R_i and so ω_i and ω_j are done from W_i. Then, m is decreased by 1. Finally, the distance matrix DM is updated by calculating the distance from the newly generated reference pixel r_0 to each reference pixel in DM except r_0.

According to the newly generated reference vector r_0 and ω_0, Pm is updated in UpdatePm () and the error e is recalculated using

$$e = \frac{1}{n}\sum_{k=1}^{n}\frac{|p_n(x_k) - p_m(x_k)|}{p_n(x_k)}$$

in the sub-routine ComputeError ().

The temporary vector R_0, W_0 and variable m_0 are used to back up the past values of R_i, W_i and m. As we can see from above pseudo-code, they take the last values of R_i, W_i and m, which have been updated in the sub-routine mergePair (). R_i, W_i and m will recover to their past values from R_0, W_0 and m_0 if e exceeds e_{max}.

The proposed algorithm WPWT: To derive the new version of p (x, y) using weighted Parzen-window estimation, the following Gaussian kernel function is introduced:

$$\begin{aligned} &G(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2) \\ &= \frac{1}{h_{C_i}^2}\varphi(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2) \end{aligned} \tag{9}$$

For Gaussian Kernel function, the following two important results are used (Torkkola, 2003):

$$\begin{aligned} &\int_D G(x, x_i, \sigma_1^2)G(x, x_j, \sigma_2^2)dx = G(x_i, x_j, \sigma_1^2 + \sigma_2^2) \\ &\int_D \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} G(x, x_i, \sigma_1^2)G(x, x_j, \sigma_2^2)dx \\ &= \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} G(x_i, x_j, \sigma_1^2 + \sigma_2^2) \end{aligned} \tag{10}$$

Thus with the obtained reference pixels and weights, we can have:

$$\begin{aligned} p(x, y) &= \sum_{i=1}^{t} p(x, y, \omega_i) = \sum_{i=1}^{t} p(\omega_i)p(x, y|\omega_i) \\ &= \frac{1}{N}\sum_{i=1}^{t}\sum_{j=1}^{m_i}\frac{w_{ij}}{V_{C_i}}\varphi(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2) \\ &= \frac{1}{N}\sum_{i=1}^{t}\sum_{j=1}^{m_i} w_{ij}G(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2) \end{aligned} \tag{11}$$

Furthermore, we have:

$$\begin{aligned} &\iint_\Omega p^2(x, y)dxdy = \frac{1}{N^2} \\ &\iint_\Omega \left(\sum_{i=1}^{t}\sum_{j=1}^{m_i}\frac{w_{ij}}{V_{C_i}}\varphi(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2)\right)^2 dxdy \\ &= \frac{1}{N^2}\iint_\Omega\left(\sum_{i=1}^{t}\sum_{j=1}^{m_i} w_{ij}G(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2)\right)^2 dxdy \\ &= \frac{1}{N^2}\iint_\Omega \sum_{i=1}^{t}\sum_{j=1}^{m_i}\sum_{l=1}^{t}\sum_{k=1}^{m_l} w_{ij}G(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2) \\ &w_{lk}G(x, y; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_l}^2)dxdy \\ &= \frac{1}{N^2}\sum_{i=1}^{t}\sum_{j=1}^{m_i}\sum_{l=1}^{t}\sum_{k=1}^{m_l} w_{ij}w_{lk}\iint_\Omega G(x, y; x_{ij}^{(r)}, y_{ij}^{(r)}; h_{C_i}^2)\cdot \\ &G(x, y; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_l}^2)dxdy \\ &= \frac{1}{N^2}\sum_{i=1}^{t}\sum_{l=1}^{t}\sum_{j=1}^{m_i}\sum_{k=1}^{m_l} w_{ij}w_{lk}G(x_{ij}^{(r)}, \\ &y_{ij}^{(r)}; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_i}^2 + h_{C_l}^2) \end{aligned} \tag{12}$$

Similarly, we have:

$$\begin{aligned} &\iint_\Omega r^2(x, y)dxdy = \frac{1}{N^2}\sum_{i=t+1}^{L}\sum_{l=t+1}^{L}\sum_{j=1}^{m_i}\sum_{k=1}^{m_l} w_{ij}w_{lk} \\ &G(x_{ij}^{(r)}, y_{ij}^{(r)}; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_i}^2 + h_{C_l}^2) \end{aligned} \tag{13}$$

$$\iint_{\Omega} p(x,y)r(x,y)dxdy = \frac{1}{N^2}\sum_{i=1}^{t}\sum_{l=t+1}^{L}\sum_{j=1}^{m_i}\sum_{k=1}^{m_l} w_{ij}w_{lk} \quad (14)$$
$$G(x_{ij}^{(r)}, y_{ij}^{(r)}; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_i}^{\ 2} + h_{C_l}^{\ 2})$$

For ease of computation, let:

$$M(i,l) = \sum_{j=1}^{m_i}\sum_{k=1}^{m_l} w_{ij}w_{lk} \quad (15)$$
$$G(x_{ij}^{(r)}, y_{ij}^{(r)}; x_{lk}^{(r)}, y_{lk}^{(r)}; h_{C_i}^{\ 2} + h_{C_l}^{\ 2})$$

We have:

$$\iint_{\Omega} p^2(x,y)dxdy = \frac{1}{N^2}\sum_{i=1}^{t}\sum_{l=1}^{t} M(i,l) \quad (16)$$

$$\iint_{\Omega} r^2(x,y)dxdy = \frac{1}{N^2}\sum_{i=t+1}^{L}\sum_{l=t+1}^{L} M(i,l) \quad (17)$$

$$\iint_{\Omega} p(x,y)r(x,y)dxdy = \frac{1}{N^2}\sum_{i=1}^{t}\sum_{l=t+1}^{L} M(i,l) \quad (18)$$

Due to the symmetry of elements in M, we only show the upper triangular matrix in Fig. 2. As we can readily see, for some t, (16), (17) and (18) corresponds to the sums of elements in area A, B and C, respectively.

Thus, t_{opt} can be computed as:

$$t_{opt} = \arg\min_{t}\left(\begin{array}{l}\iint_{\Omega} p^2(x,y)dxdy + \\ \iint_{\Omega} r^2(x,y)dxdy - 2 \\ \iint_{\Omega} p(x,y)r(x,y)dxdy\end{array}\right) \quad (19)$$
$$= \arg\min_{t}\left(sum(A) + sum(B) - 2\times sum(C)\right)$$

Based on the above, we summarize the novel image thresholding method WPWT as follows:

- Construct the histogram for an N = m×n image with L gray levels.
- For each gray level, compute the reference pixels using the training algorithm in Fig. 1.
- Construct the matrix M using (15).
- For each gray level, compute the corresponding t and find t_{op}
- Segment the image with t_{opt} and output the segmented image accordingly

```
BEGIN
For i=0:L-1
    m = n;
    If (m<=0) continue;
    e = 0;
    R_i=[r_i1, r_i2, …, r_im];
    W_i=[w_i1, w_i2, …, w_im]=1;
    Compute Pn using (8) at gray level i;
    Pm=Pn;
    Construct the distance matrix DM for the pixels at
    gray level i;
    While (m>1 && e<emax)
        [r_1, r_2, w_1, w_2]=getNearestPair(DM);
        R_0= R_i; W_0= W_i; m_0=m;
        [r_0, w_0]=mergePair(r_1, r_2, w_1, w_2);
        UpdatePm(r_0, w_0);
        e = ComputeError(Pn, Pm);;
        If (e>=emax)
            R_i =R_0; W_i =W_0; m=m_0;
                break;
            End if
        End while
        Store R_i and W_i as the reference pixels and
        weights for gray level i;
End for
END
```

Fig. 1: The training procedure

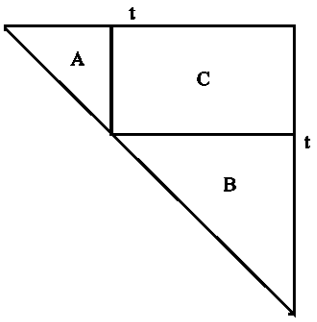

Fig. 2: Upper triangular matrix of M

RESULTS

Experiments were conducted with several images provided by MATLAB and these images have been widely used in literature. We implemented the proposed algorithm WPWT with MATLAB 7 and run it on the computer with P4 2.66GHz CPU and 512M memory. In order to observe how e_{max} values affect the performance of our algorithm, we used different e_{max} values varying from 0.1 to 0.4 with step length 0.1.

The first experiment is used to show how different e_{max} values affect the numbers of the generated reference pixels (Table 1).

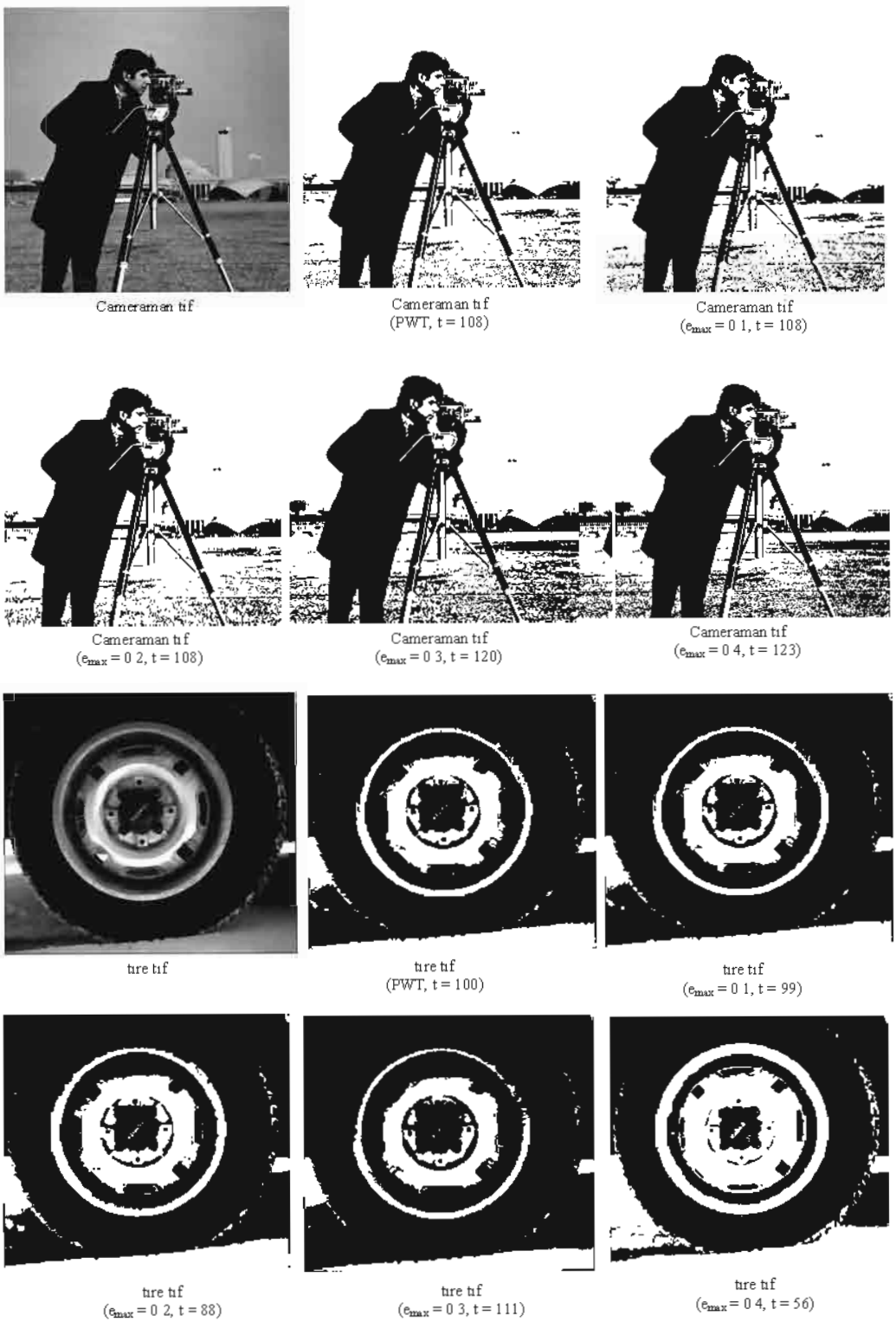

Fig. 3: Continued

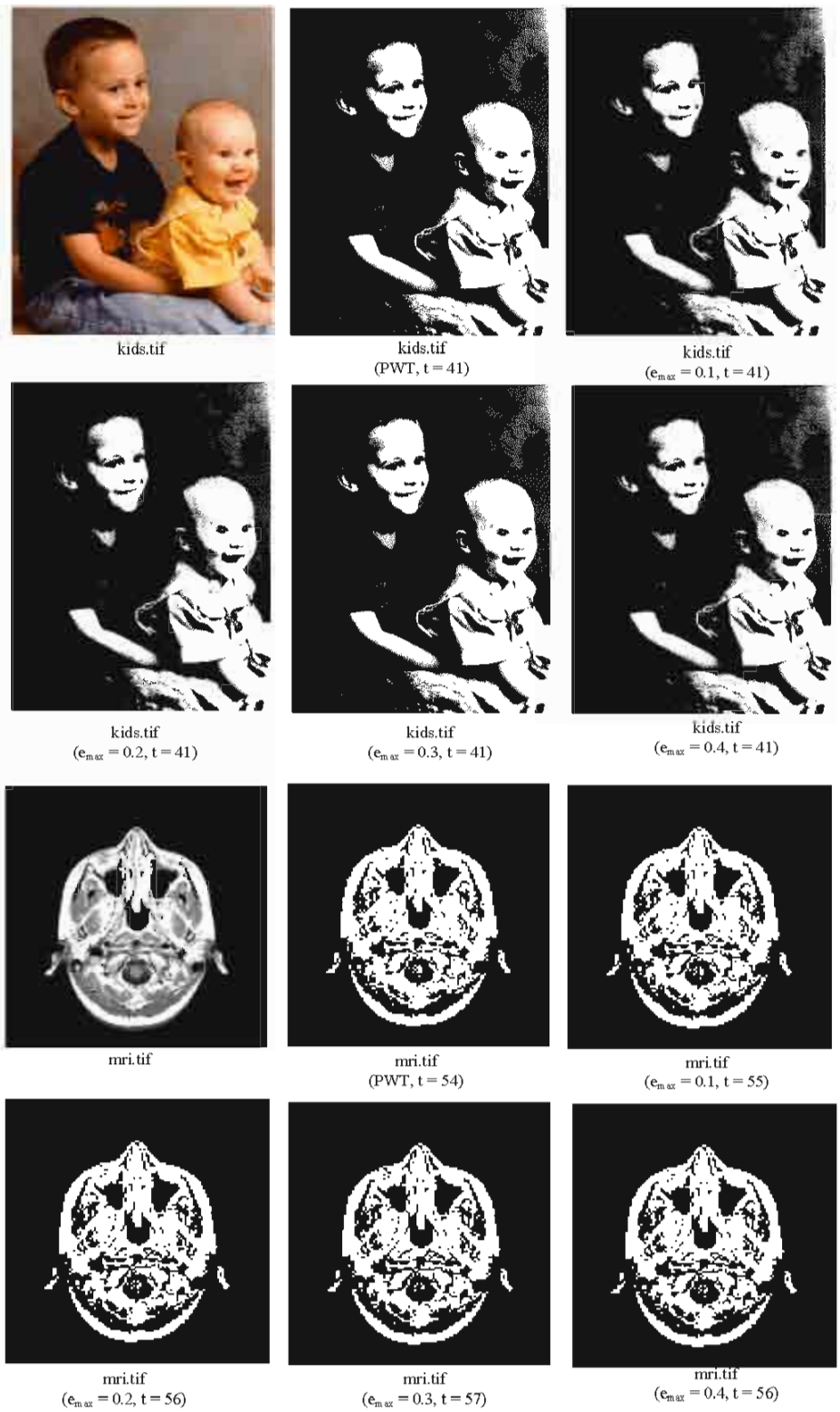

Fig. 3: Segmentation results of WPWT and PWT in Wang *et al.* (2008)

Cameraman tif with white noise Power = 23

Cameraman tif with noise (e_{max} = 0 3, t = 130)

Cameraman tif with noise (e_{max} = 0 3, t = 131)

Fig. 4: Segmentation results on noisy images

Table 1 The number of the reference pixels obtained by the training procedure

	PWT (Wang *et al*, 2008)	e_{max} = 0 1	e_{max} = 0 2	e_{max} = 0 3	e_{max} = 0 4
cameraman tif	65536	61560	57970	54367	50325
tire tif	47560	44244	41640	39045	36305
kids tif	127200	122274	117317	112352	107351
mri tif	16384	15050	14156	13293	12368

Table 2 Running time of WPWT and PWT (seconds)

	PWT (Wang *et al*, 2008)	e_{max} = 0 1	e_{max} = 0 2	e_{max} = 0 3	e_{max} = 0 4
cameraman tif	147 5	131 1	118 1	104 8	91 3
tire tif	67 4	59 8	50 0	44 6	39 8
kids tif	450 6	379 0	369 9	329 6	312 5
mri tif	13 2	11 7	10 1	9 5	8 1

The second experiment is taken to show the influence of different e_{max} values on running speed of the algorithm as well as visual effects of the thresholding results. The complexity of both PWT and our proposed method is O (n^2/L), where n is the total number of pixels in the image and L is the number of gray levels. In our proposed method, n is decreased greatly by controlling e_{max} so that the performance of our algorithm can be enhanced. In order to make a comparison with PWT in Wang *et al.* (2008), Table 2 shows the running time of both WPWT and PWT with different e_{max}.

An appropriate e_{max} is vital to the running speed of Parzen-window estimation. According to our experiments, for many images, e_{max} = 0.3 is a good choice for the speed up of the proposed algorithm and the final visual effects of the segmented images. Figure 3 shows the corresponding segmentation results of both WPWT and PWT.

Obviously, e_{max} has an important influence on the optimal threshold t_{opt} However, when $e_{max} \in [0.2, 0.4]$, all the obtained segmentation results are very acceptable from the viewpoint of the visual effects for the segmented images.

The final experiment shows the performance of our algorithm on noisy images. In this experiment, white noise is added into the original image and the segmentation results obtained by the WPWT method are shown in Fig. 4. From the obtained results, we can see that the proposed algorithm can obtain almost the same results with that the original images, which verifies the robustness of the proposed algorithm on noisy images.

CONCLUSION

We propose a novel image thresholding method based on weighted Parzen-window estimation in this study. In our method, we decrease the number of pixels by using the Parzen-window estimation with a hierarchical clustering procedure which calculates a set of the reference pixels and weights by merging the corresponding pixel pairs. During this process, the error produced by merging pixel pairs is controlled by *emax*. Then the thresholding algorithm based on weighted Parzen-window estimation (i.e., WPWT) is presented. Unlike PWT in Wang *et al.* (2008), the proposed WPWT here approximates the probability of each pixel with the reference pixels and weights. Compared with PWT, the experimental results here show that our method can accelerate the image thresholding and reduce the space requirements greatly without degrading final segmentation results a lot.

ACKNOWLEDGMENTS

This study is supported by National 973 Key Project (Grant No. 2006CB705700), National Science Foundation of China, National 863 Project (Grant No.2007AA1Z158), National Defense Basic Research Grant, New_century Outstanding Young Scholar Grant of Ministry of Education of China (Grant No. NCET-04-0496), National KeySoft Lab. at Nanjing University, the Key Lab. of Computer Science at Institute of Software, CAS, China, the Key Lab. of Computer Information Technologies at JiangSu Province, the 2004 key project of Ministry of Education of China and National Key Lab. of Pattern Recognition at Institute of Automation, CAS, China.

REFERENCES

de Albuquerque, M.P., I.A. Esquef and A.R. Gesualdi Mello, 2004. Image thresholding using Tsallis entropy. Pattern Recog. Lett., 25 (10): 1059-1065.

Sarkar, S. and P. Soundararajan, 2000. Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata. IEEE Trans. Pattern Anal. Mach. Intel., 22 (5): 504-525.

Sezgin, M. and B. Sankur, 2004. Survey over image thresholding techniques and quantitative performance evaluation. J. Elect. Imaging, 13 (1): 146-168.

Torkkola, K., 2003. Feature extraction by non-parametric mutual information maximization. J. Mach. Learning Res., 3: 1415-1438.

Wang, S.T. *et al*., 2008. A novel image thresholding method based on Parzen window estimate. Pattern Recog., 41 (1): 117-129.

Yang, L. and X. Yang, 2004. Multi-object segmentation based on curve evolving and region division. Chin. J. Comput., 27 (3): 420-425 (in Chinse with English abstract).

Zhang, Y.J., 2001. Image Segmentation. Press of Science, Beijing (In Chinese).