Available online at www.sciencedirect.com
ScienceDirect
Applied Mathematics and Computation 184 (2007) 503–513
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Multilevel minimum cross entropy threshold selection based on particle swarm optimization

Peng-Yeng Yin

Department of Information Management, National Chi Nan University, 303 University Road, Puli, Nantou 545, Taiwan

Abstract

Thresholding is one of the popular and fundamental techniques for conducting image segmentation. Many thresholding techniques have been proposed in the literature. Among them, the minimum cross entropy thresholding (MCET) have been widely adopted. Although the MCET method is effective in the bilevel thresholding case, it could be very time-consuming in the multilevel thresholding scenario for more complex image analysis. This paper first presents a recursive programming technique which reduces an order of magnitude for computing the MCET objective function. Then, a particle swarm optimization (PSO) algorithm is proposed for searching the near-optimal MCET thresholds. The experimental results manifest that the proposed PSO-based algorithm can derive multiple MCET thresholds which are very close to the optimal ones examined by the exhaustive search method. The convergence of the proposed method is analyzed mathematically and the results validate that the proposed method is efficient and is suited for real-time applications.

Keywords: Minimum cross entropy thresholding; Multilevel thresholding; Recursive programming; Particle swarm optimization

1. Introduction

Image segmentation is fundamental to many image analysis tasks such as object tracking, character recognition, document analysis, just to name a few. Thresholding is one of the most important techniques for performing image segmentation. The goal of thresholding is to select a set of thresholds which can discriminate object and background pixels. Bilevel thresholding selects only one threshold which separates the pixels into two classes, while multilevel thresholding determines multiple thresholds which divide the pixels into several groups. Although the bilevel thresholding is easily adopted, it is not uncommon that the multilevel thresholding is employed for more complex analysis tasks such as color image segmentation [21,14] and mixed-type documents analysis [29].

Over the years, many thresholding techniques have been proposed. Comprehensive surveys can be found in [15,25,27]. These thresholding methods can be roughly divided into two categories. The first category contains the approaches which determine the optimal thresholds by analyzing the profile characteristics of the image histogram. Rosenfeld and De la Torre [24] analyzed the concavities of the histogram by calculating its convex

E-mail address: pyyin@ncnu.edu.tw

0096-3003/$ - see front matter
doi:10.1016/j.amc.2006.06.057

hull. Lim and Lee [18] presented a valley-seeking approach which smoothes the histogram and detects the valleys as thresholds by calculating the derivatives of the smoothed histogram. Yin and Chen [33] proposed another valley-seeking algorithm based on symmetry and duality and a threshold hierarchy is provided for satisfying applications with different granularity. The second category belong the thresholding techniques which determine the optimal thresholds by optimizing a certain objective function. Otsu [19] proposed a thresholding technique which maximizes the between-class variance of gray levels of the object and the background portions. Kittler and Illingworth [12] developed a thresholding method which approximates the histogram by a mixture of normal distributions and minimizes the classification error probability. Pun [23] found the optimal threshold by maximizing a posteriori entropy of the object and background portions.

Among the tremendous amount of image thresholding techniques, entropy-based approaches have drawn the attentions of many researchers. Kapur et al. [9] found some flaws in Pun's derivations and further presented a corrected version. Yen et al. [32] define the entropic correlation and obtain the threshold that maximizes it. Abutaleb [1] extended the entropy thresholding method on a 2D histogram such that the spatial correlation between the pixels can be taken into account. Cheng et al. [3] developed the fuzzy version of the entropy thresholding. Sahoo et al. [26] generalized some of the existing entropy thresholding techniques by Renyi's entropy using different parameter ranges of Renyi power. Li and Lee [16] proposed a thresholding method which selects the threshold by minimizing the cross entropy between the original and segmented images. Brink and Pendock [2] deliberately showed the relationship between the minimum cross entropy thresholding technique and other methods. Pal [20] modeled the histogram by a mixture of Poisson distributions and segmented the image by minimizing the total cross entropy of the object and background portions. Li and Tam [17] presented a fast iterative implementation for the minimum cross entropy thresholding method. However, the iterative scheme is hard to extend to multiple thresholds selection.

The deployment of meta-heuristic computing has been flourishing during the last decade. Many meta-heuristic paradigms such as genetic algorithm [8], simulated annealing [11], tabu search [7], ant colony optimization [5], and particle swarm optimization [10] have been applied to tackle many well-known NP-hard problems. Encouraged by their successful applications, we further investigate the feasibility of using meta-heuristic algorithms for solving image thresholding.

The endeavor of this paper is focused on multilevel thresholding using the minimum cross entropy criterion. We first propose a recursive programming technique which stores the results of preceding tries as the basis for the computation of succeeding ones. Then, based on the recursive programming technique, a particle swarm optimization (PSO) algorithm is presented for searching the optimal thresholds. The performance of the proposed method is evaluated by testing on several real images. The experimental results manifest that our method is efficient and effective.

The remainder of this paper is organized as follows. Section 2 reviews the minimum cross entropy thresholding technique. Section 3 describes the proposed method. In Section 4, the experimental results and comparative performances are presented. Finally, conclusions are made in Section 5.

2. Minimum cross entropy thresholding

The cross entropy was proposed by Kullback in [13]. Let $F = \{f_1, f_2, \ldots, f_N\}$ and $G = \{g_1, g_2, \ldots, g_N\}$ be two probability distributions on the same set. The cross entropy between F and G is an information theoretic distance between the two distributions and it is defined by

$$D(F,G) = \sum_{i=1}^{N} f_i \log \frac{f_i}{g_i}. \tag{1}$$

The minimum cross entropy thresholding (MCET) algorithm [16] selects the threshold by minimizing the cross entropy between the original image and its thresholded version. Let I be the original image and $h(i)$, $i = 1, 2, \ldots, L$, be the corresponding histogram with L being the number of gray levels. Then the thresholded image, denoted by I_t, using t as the threshold value is constructed by

$$I_t(x,y) = \begin{cases} \mu(1,t), & I(x,y) < t, \\ \mu(t, L+1), & I(x,y) \geqslant t, \end{cases} \tag{2}$$

where

$$\mu(a,b) = \sum_{i=a}^{b-1} ih(i) \Big/ \sum_{i=a}^{b-1} h(i). \tag{3}$$

The cross entropy is then calculated by

$$D(t) = \sum_{i=1}^{t-1} ih(i)\log\left(\frac{i}{\mu(1,t)}\right) + \sum_{i=t}^{L} ih(i)\log\left(\frac{i}{\mu(t,L+1)}\right). \tag{4}$$

The MCET determines the optimal threshold t^* by minimizing the cross entropy, *viz.*,

$$t^* = \arg\min_t\{D(t)\}. \tag{5}$$

The computational complexity for determining t^* is $O(L^2)$. However, it could be time-consuming under the multilevel thresholding scenario. For the n-thresholding problem, it requires $O(L^{n+1})$.

3. Proposed method

As traditional techniques for deriving MCET thresholds could be computationally intensive, we propose a fast MCET algorithm based on PSO. We first present a recursive programming technique for computing the MCET objective function. Then the PSO-based algorithm is devised for searching the optimal thresholds.

3.1. Recursive programming

The MCET objective function (4) can be rewritten as

$$D(t) = \sum_{i=1}^{L} ih(i)\log(i) - \sum_{i=1}^{t-1} ih(i)\log(\mu(1,t)) - \sum_{i=t}^{L} ih(i)\log(\mu(t,L+1)). \tag{6}$$

Since the first term is constant for a given image, the objective function can be redefined as

$$\begin{aligned}\eta(t) &= -\sum_{i=1}^{t-1} ih(i)\log(\mu(1,t)) - \sum_{i=t}^{L} ih(i)\log(\mu(t,L+1)) \\ &= -\left(\sum_{i=1}^{t-1} ih(i)\right)\log\left(\sum_{i=1}^{t-1} ih(i) \Big/ \sum_{i=1}^{t-1} h(i)\right) - \left(\sum_{i=t}^{L} ih(i)\right)\log\left(\sum_{i=t}^{L} ih(i) \Big/ \sum_{i=t}^{L} h(i)\right) \\ &= -m^1(1,t)\log\left(\frac{m^1(1,t)}{m^0(1,t)}\right) - m^1(t,L+1)\log\left(\frac{m^1(t,L+1)}{m^0(t,L+1)}\right),\end{aligned} \tag{7}$$

where $m^0(a,b) = \sum_{i=a}^{b-1} h(i)$ and $m^1(a,b) = \sum_{i=a}^{b-1} ih(i)$ are the zero-moment and first-moment on partial range of the image histogram.

Here we propose a recursive programming for expediting the computing process of the objective function (7) with different trial thresholds. Suppose that the current trial threshold is t and the corresponding objective value $\eta(t)$ has been computed. By tallying the intermediate moments $m^0(1,t)$, $m^0(t,L+1)$, $m^1(1,t)$, and $m^1(t,L+1)$, the other moments can be computed recursively by the following equation:

$$\begin{aligned} m^0(1,t+1) &= m^0(1,t) + h(t), \quad m^0(t+1,L+1) = m^0(t,L+1) - h(t), \\ m^1(1,t+1) &= m^1(1,t) + th(t), \quad m^1(t+1,L+1) = m^1(t,L+1) - th(t). \end{aligned} \tag{8}$$

Hence, the computation for $\eta(t+1)$ only costs a constant time if the intermediate moments are tallied. As such, the computational complexity for solving $\arg\min_t\{\eta(t)\}$ is reduced by an order of magnitude.

The recursive programming technique is easily extended to multilevel thresholding case. Assume that it is required to select n thresholds denoted by $t_1, t_2, \ldots, t_n$. For the convenience of illustration, we add two dummy thresholds $t_0 \equiv 1$, $t_{n+1} \equiv L + 1$, and $t_0 < t_1 < \cdots < t_n < t_{n+1}$. The objective function then becomes

$$\eta(t_1, t_2, \ldots, t_n) = -\sum_{i=1}^{n+1} m^1(t_{i-1}, t_i) \log\left(\frac{m^1(t_{i-1}, t_i)}{m^0(t_{i-1}, t_i)}\right). \tag{9}$$

Before solving $\eta(t_1, t_2, \ldots, t_n)$, we first conduct the recursive programming for obtaining $m^0(a,b)$ and $m^1(a,b)$ for $1 < a < b < L$, which costs $O(L^2)$ for computation. Hence, $\arg\min_{t_1,t_2,\ldots,t_n}\{\eta(t_1, t_2, \ldots, t_n)\}$ can be solved in $O(nL^n + L^2) = O(nL^n)$, which is less than the original computational complexity $O(L^{n+1})$ for deriving n optimal thresholds since $n \ll L$ in practice. Although the computational complexity is reduced, $O(nL^n)$ is still computationally expensive if $n \geqslant 3$. In this paper, we further present a particle swarm optimization algorithm based on the recursive programming technique for solving $\arg\min_{t_1,t_2,\ldots,t_n}\{\eta(t_1, t_2, \ldots, t_n)\}$ efficiently and effectively.

3.2. Particle swarm optimization

The particle swarm optimization (PSO) algorithm was first proposed by Kennedy and Eberhart [10]. It is biologically inspired by observations from social dynamics of bird flocking. Ethologists find that a large number of birds flock synchronously, change direction suddenly, scatter and regroup iteratively, and finally perch on a target. This form of social activity not only increases the success rate for food foraging but also expedites the process. The PSO algorithm simulating bird foraging activity can serve as an optimizer for nonlinear functions of continuous and discrete variables. PSO has exhibited great success in many applications including evolving structure for artificial neural networks [6], manufacture end milling [30], reactive power and voltage control [35], state estimation for electric power distribution systems [28], and curve segmentation [34]. The convergence and parameterization aspects of the PSO have also been discussed [22,4,31]. The general principles of the PSO algorithm are outlined as follows.

- *Particle formulation*. The PSO simulates the bird flock by a swarm of particles. Each particle is a candidate solution to the underlying problem and iteratively moves in the solution space. The particle is a real-valued vector consists of parameter values that characterize the optimization problem. We denote the ith particle by $P_i = (p_{i1}, p_{i2}, \ldots, p_{in})^T \in R^n$, where n is the number of parameters. The particles are collision free, that is, multiple particles are allowed to move to the same position.
- *Swarm*. The PSO explores the solution space by flying a number of particles, called swarm. The initial swarm is usually generated at random or according to a problem-specific heuristic. The swarm size is usually kept constant through iterations. At each iteration, the swarm of particles fly to new positions for targeting the optimal solution by referring to previous flying experiences.
- *Personal best experience and swarm's best experience*. The PSO enriches the swarm intelligence by collecting the awareness from individual particle. In particular, the PSO tallies the best positions visited so far by every particle. We denote by $pbest_i$ the best position ever visited by particle i. There are two versions for recording the swarm's best position, namely the *lbest* and *gbest*. In the *lbest* version, particle i keeps track of the best position, denoted by $lbest_i$, attained by its local neighborhood of particles. For the *gbest* version, the swarm's best position, denoted by *gbest*, is determined by any particles in the entire swarm. Hence, the *gbest* model is a special case of the *lbest* model. Literature has shown that the *lbest* version is often better than the *gbest* version.
- *Particle movement*. The PSO is an evolutionary algorithm according to which a swarm of particles evolve to new positions until the stopping criterion is reached. At each iteration, particle i adjusts its velocity v_{ij} and position p_{ij} through each dimension j by referring to, with random multipliers, the personal best position ($pbest_{ij}$) and the swarm's best position ($lbest_{ij}$) using Eqs. (10) and (11) as follows:

$$v_{ij} = wv_{ij} + c_1 r_1 (pbest_{ij} - p_{ij}) + c_2 r_2 (lbest_{ij} - p_{ij}) \tag{10}$$

and

$$p_{ij} = p_{ij} + v_{ij}, \tag{11}$$

where w is the inertia weight, c_1 and c_2 are the cognitive coefficients, and r_1 and r_2 are random real numbers drawn from $U(0, 1)$. Hence the particle flies toward *pbest* and *lbest* in a navigated way while still can escape from the barrier of local optimality by the stochastic mechanism. Clerc and Kennedy [4] has pointed out that the use of a constriction factor is needed to ensure the convergence of the algorithm by replacing Eq. (10) with the following:

$$v_{ij} = K[v_{ij} + c_1r_1(pbest_{ij} - p_{ij}) + c_2r_2(lbest_{ij} - p_{ij})] \tag{12}$$

and

$$K = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \tag{13}$$

where $\varphi = c_1 + c_2$ and $\varphi > 4$. Typically, φ is set to 4.1 and K is thus 0.729.

- *Stopping criterion*. The PSO algorithm is terminated with a maximal number of iterations or the best particle position of the entire swarm cannot be improved further after a sufficiently large number of iterations.

3.3. The proposed algorithm

Inspired by the great success of PSO for solving many complex problems, we present a PSO-based multilevel MCET algorithm incorporating the recursive programming technique to conquer the curse of high dimensionality.

We formulate each particle as a candidate solution to the multilevel MCET problem. For an n-threshold MCET problem, the particle is formulated as

$$P = (t_1, t_2, \ldots, t_n)^{\mathrm{T}}, \quad \text{subject to } 1 < t_1 < t_2 < \cdots < t_n < L. \tag{14}$$

Here, the n parameters in the particle representation correspond to the n multiple thresholds for the MCET problem and serve as a candidate solution. The initial swarm consists of S particles which are generated randomly according to (14). The values of the moments $m^0(a,b)$ and $m^1(a,b)$ for $1 < a < b < L$ are pre-computed using the recursive programming technique and can be used for expediting the computation of the objective function with n thresholds as described in Section 3.1.

In PSO, particles are competing for *pbest* and *lbest* by evaluating the solution quality, or *fitness*. For the MCET problem, the particle deriving the minimal objective value is considered to be the best, so the objective function (9) can be used for measuring the particle fitness. The velocity updating and particle movement follow the guidelines of PSO, and the swarm evolves until the maximal number of iterations are experienced. When the algorithm terminates, the best position visited so far by the entire swarm is output as the optimal solution found by the algorithm. The PSO-based algorithm for the MCET problem is summarized in Fig. 1.

1. Initialize.
   1.1 Compute $m^0(a,b)$ and $m^1(a,b)$ for $1 < a < b < L$ using the recursive programming technique.
   1.2 Generate S particles according to the particle formulation (14).
   1.3 Generate velocities randomly from [-1.0, 1.0].
2. Repeat for a given maximal number of iterations.
   2.1 Evaluate the fitness of each particle using (9) based on the pre-computed moments.
   2.2 Determine the personal best position $pbest_i$ visited so far by each particle.
   2.3 Determine the local best position $lbest_i$ visited so far by the local neighboring particles of each particle.
   2.4 Update velocities v_{ij} using (12) and (13).
   2.5 Update particles' positions using (11).

Fig. 1. PSO-based algorithm for the MCET problem.

4. Experimental results and comparative performances

We implement the proposed PSO-based algorithm in C++ language with a Pentium IV 1.8 GHz PC. Four images named "WORD", "HEAD", "LENA", and "PEPPER", with image size of $96 * 105$, $468 * 414$, $512 * 512$, and $512 * 512$, respectively, are used for conducting our experiments. The original test images are shown in Fig. 2.

First, in the bilevel thresholding scenario, the proposed recursive programming technique can find the exact optimal solution quickly. Fig. 3 shows the optimal bilevel MCET threshold derived by the recursive programming technique and the corresponding segmented images. It is seen that, for images WORD and HEAD, the minimum cross entropy criterion is feasible and the objects are well segmented. As for images LENA and PEPPER, the image histograms are more complex, a sophisticated segmentation based on multilevel thresholding is needed.

For applying the multilevel thresholding on images LENA and PEPPER, we execute the PSO-based algorithm with three particles for 200 iterations. The exhaustive search method is also conducted for deriving the optimal solutions for comparison. Table 1 shows the multilevel MCET thresholds derived by the PSO-based algorithm and the optimal thresholds reported by the exhaustive search method for image LENA. We observe that the derived MCET thresholds by the PSO-based algorithm are equivalent (for the 2-threshold and 3-threshold problems) or very close (for the 4-threshold problem) to the optimal thresholds derived by the exhaustive search method, and the computation times needed for the PSO-based algorithm are negligible. However, the computation time needed for the exhaustive search method grows exponentially with the number of required thresholds. Hence, the proposed PSO-based method is useful in finding multiple MCET thresholds for complex image analysis.

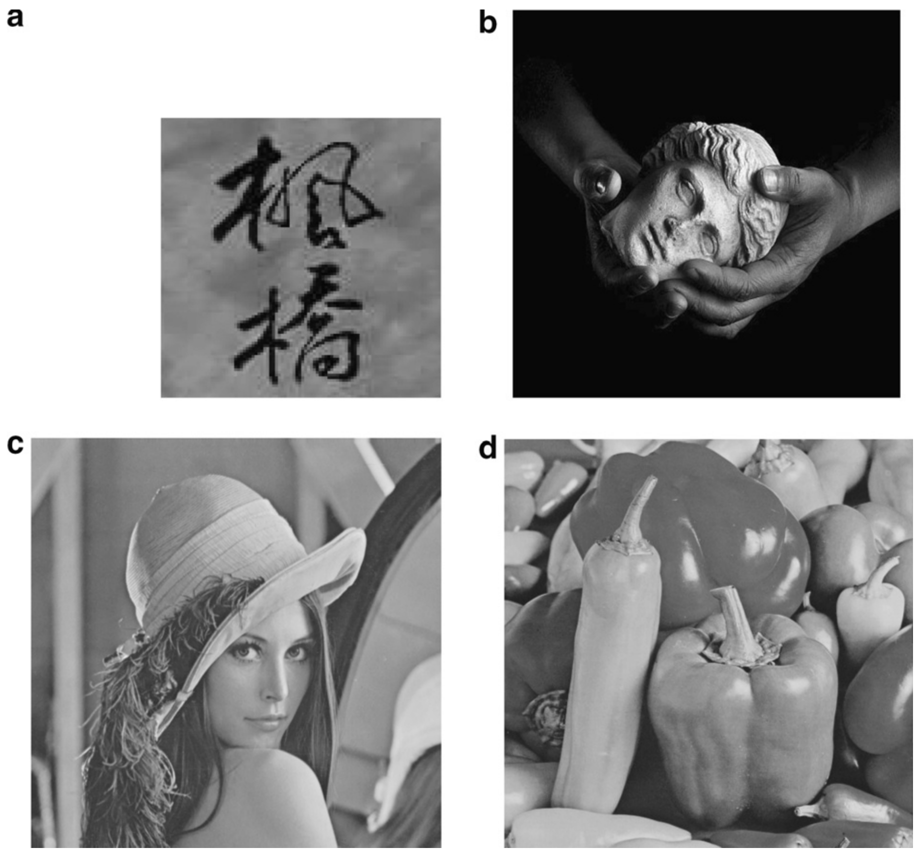

Fig. 2. The test images: (a) WORD, (b) HEAD, (c) LENA, and (d) PEPPER.

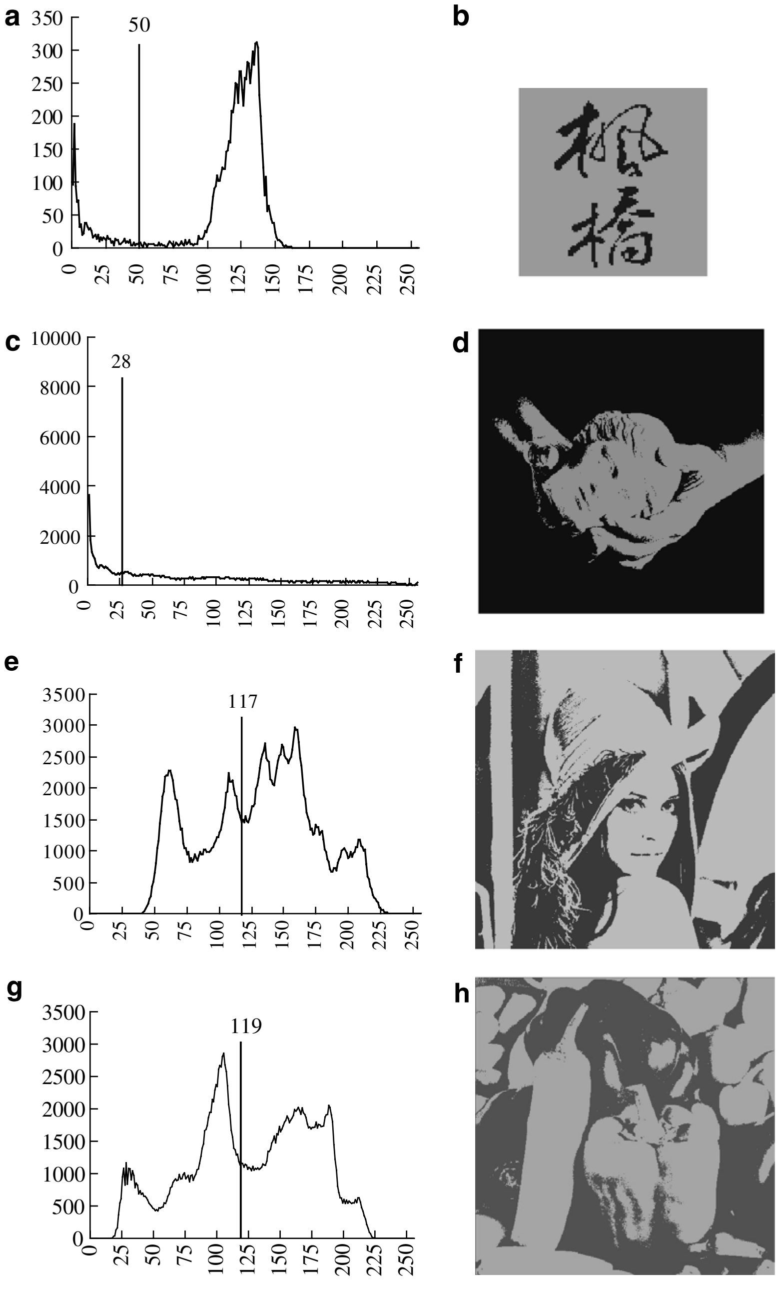

Fig. 3. The optimal bilevel MCET threshold and the segmented images.

Table 1
The MCET thresholds for LENA derived by the PSO-based method and the exhaustive search method, the computation times are reported in seconds

n thresholds	PSO-based		Exhaustive	
	Thresholds	CPU time	Thresholds	CPU time
2	93,147	0.01	93,147	0.26
3	85,127,169	0.01	85,127,169	16.20
4	81,116,145,178	0.01	81,116,146,179	987.12

Fig. 4 shows the derived multiple MCET thresholds superimposed on the histograms for the PSO-based method, the corresponding multilevel thresholded images are also illustrated. It is shown that as the number of thresholds increases, the image quality improves.

Table 2 displays the derived multiple MCET thresholds and the consumed computation times by the PSO-based algorithm and the exhaustive search method for image PEPPER. Analogous to the previous experiment, it is seen that the derived MCET thresholds by the PSO-based algorithm are very close to the optimal thresholds derived by the exhaustive search method, and the computation times needed for the PSO-based algorithm are negligible. On the other hand, because the computation complexity for the exhaustive search is exponential, the needed CPU times for $n \geqslant 3$ are absolutely unacceptable.

Fig. 5 corresponds to the derived multiple MCET thresholds superimposed on the histograms and the corresponding segmented images by the proposed PSO-based method. Similarly, the segmented images are more informative as the number of thresholds increases. Thus, the proposed PSO-based algorithm is suited for more complex image analysis.

To analyze the convergence behavior of the proposed PSO-based algorithm, we examine whether all of the particles of the swarm evolve to the same optimization target. We propose the information *entropy* for measuring the similarity convergence among the particles as follows. Let p_i be the probability with which a particle in the swarm chooses gray level i as one of the MCET thresholds. We can calculate the *particle entropy* as follows.

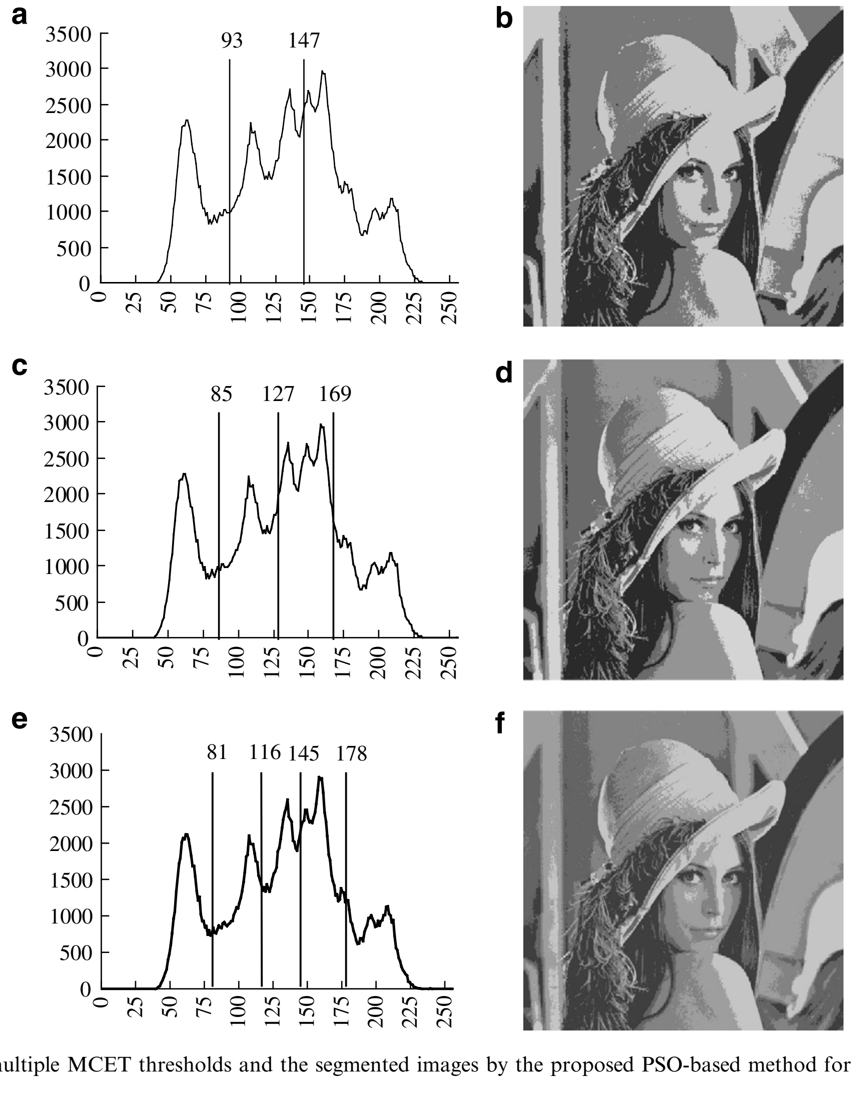

Fig. 4. The multiple MCET thresholds and the segmented images by the proposed PSO-based method for image LENA.

Table 2
The MCET thresholds for PEPPER derived by the PSO-based method and the exhaustive search method, the computation times are reported in seconds

n thresholds	PSO-based		Exhaustive	
	Thresholds	CPU time	Thresholds	CPU time
2	68,133	0.01	68,134	0.32
3	64,117,163	0.01	64,117,164	21.51
4	51,87,125,167	0.01	51,87,126,167	1037.64

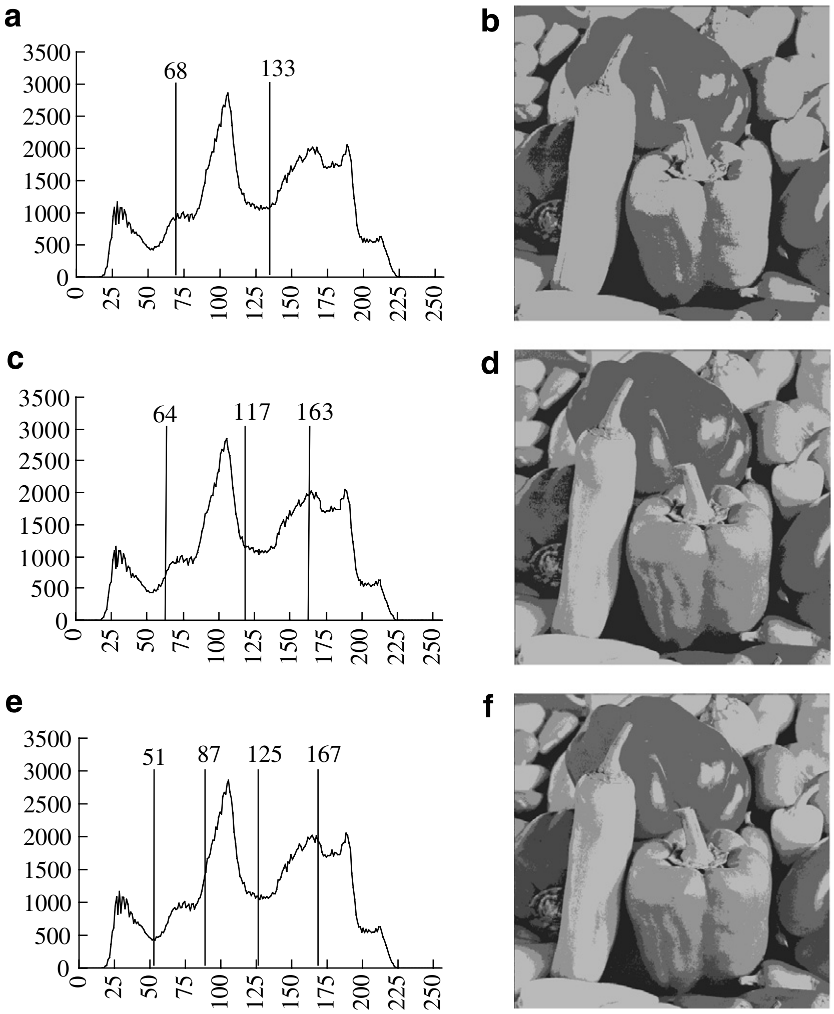

Fig. 5. The multiple MCET thresholds and the segmented images by the proposed PSO-based method for image PEPPER.

$$\text{Entropy} = -\sum_{i=1}^{L} p_i \log_2(p_i). \tag{15}$$

The particle entropy is smaller if the probability distributions are denser. As such, the variations of particle entropy during the swarm evolution measure the convergence about the similarity among all particles. If the particles are highly similar to one another, the values of the non-zero p_i would be high, resulting in denser probability distributions and less entropy value. This also means the swarm particles reach the consensus about which gray levels should be selected as the MCET thresholds for the test image.

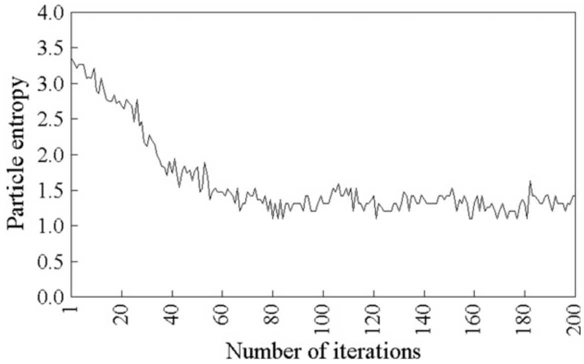

Fig. 6. The variations of the particle entropy as the number of iterations increases.

Fig. 6 shows the variations of particle entropy as the number of iterations increases. It is observed that the trend of the entropy value plunges down as the swarm evolves during the first 60 generations since the particles exchange information by referring to the swarm's best solution. After this period, the entropy value is relatively fixed due to the good quality solutions found and the high similarity among the particles, meaning the particles are resorting to the same MCET thresholds as the swarm converges.

5. Conclusions

In this paper, we have proposed a particle swarm optimization (PSO)-based method for selecting multiple minimum cross entropy thresholds (MCET). The contributions of our paper include: (1) A recursive programming technique is proposed for reducing the computation complexity of the MCET objective function by an order of magnitude. (2) We extend the application of MCET to multilevel thresholding such that the real-time complex image analysis is feasible. (3) The experimental result is promising and it encourages future research for applying PSO to complex image processing and computer vision problems.

References

[1] A.S. Abutaleb, Automatic thresholding of gray-level pictures using two-dimensional entropy, Computer Vision Graphics and Image Processing 47 (1989) 22–32.
[2] A.D. Brink, N.E. Pendock, Minimum cross-entropy threshold selection, Pattern Recognition 29 (1996) 179–188.
[3] H.D. Cheng, J.R. Chen, J.G. Li, Threshold selection based on fuzzy *c*-partition entropy approach, Pattern Recognition 31 (1998) 857–870.
[4] M. Clerc, J. Kennedy, The particle swarm explosion, stability, and convergence in a multidimensional complex space, IEEE Transaction on Evolutionary Computation 6 (2002) 58–73.
[5] M. Dorigo, Optimization, learning, and natural algorithms, Ph.D. Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy, 1992.
[6] R.C. Eberhart, Y. Shi, Evolving artificial neural networks, in: Proc. Int'l. Conf. on Neural Networks and Brain, PL5–PL13, 1998.
[7] F. Glover, Tabu search – Part I, ORSA Journal of Computing 1 (1989) 190–206.
[8] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, Reading, MA, 1997.
[9] J.N. Kapur, P.K. Sahoo, A.K.C. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Computer Vision Graphics and Image Processing 29 (1985) 273–285.
[10] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proc. IEEE Int'l. Conf. on Neural Networks, vol. IV, 1995, pp. 1942–1948.
[11] S. Kirkpatrick, C. Gelatt Jr., M. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[12] J. Kittler, J. Illingworth, Minimum error thresholding, Pattern Recognition 19 (1986) 41–47.
[13] S. Kullback, Information Theory and Statistics, Dover, New York, 1968.
[14] F. Kurugollu, B. Sankur, A.E. Harmanci, Color image segmentation using histogram multithresholding and fusion, Image and Vision Computing 19 (2001) 915–928.
[15] S.U. Lee, S.Y. Chung, R.H. Park, A comparative performance study of several global thresholding techniques for segmentation, Computer Vision Graphics and Image Processing 52 (1990) 171–190.
[16] C.H. Li, C.K. Lee, Minimum cross entropy thresholding, Pattern Recognition 26 (1993) 617–625.

[17] C.H. Li, P.K.S. Tam, An iterative algorithm for minimum cross entropy thresholding, Pattern Recognition Letters 19 (1998) 771–776.
[18] Y.K. Lim, S.U. Lee, On the color image segmentation algorithm based on the thresholding and the fuzzy *c*-means techniques, Pattern Recognition 23 (1990) 935–952.
[19] N. Otsu, A threshold selection method from gray-level histograms, IEEE Transactions on Systems Man and Cybernetics SMC-9 (1979) 62–66.
[20] N.R. Pal, On minimum cross-entropy thresholding, Pattern Recognition 29 (1996) 575–580.
[21] N. Papamarkos, C. Strouthopoulos, I. Andreadis, Multithresholding of color and gray-level images through a neural network technique, Image and Vision Computing 18 (2000) 213–222.
[22] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization, Natural Computing 1 (2002) 235–306.
[23] T. Pun, Entropy thresholding: A new approach, Computer Vision Graphics and Image Processing 16 (1981) 210–239.
[24] A. Rosenfeld, P. De la Torre, Histogram concavity analysis as an aid in threshold selection, IEEE Transactions on Systems Man and Cybernetics SMC-13 (1983) 231–235.
[25] P.K. Sahoo, S. Soltani, A.K.C. Wong, Y.C. Chen, A survey of thresholding techniques, Computer Vision Graphics and Image Processing 41 (1988) 233–260.
[26] P.K. Sahoo, C. Wilkins, J. Yeager, Threshold selection using Renyi's entropy, Pattern Recognition 30 (1997) 71–84.
[27] B. Sankur, M. Sezgin, Survey over image thresholding techniques and quantitative performance evaluation, Journal of Electronic Imaging 13 (2004) 146–165.
[28] N. Shigenori, G. Takamu, Y. Toshiku, F. Yoshikazu, A hybrid particle swarm optimization for distribution state estimation, IEEE Transaction on Power Systems 18 (2003) 60–68.
[29] C. Strouthopoulos, N. Papamarkos, Multithresholding of mixed-type documents, Engineering Applications of Artificial Intelligence 13 (2000) 323–343.
[30] V. Tandon, Closing the gap between CAD/CAM and optimized CNC end milling, Master thesis, Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis, 2000.
[31] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters 85 (2003) 317–325.
[32] J.C. Yen, F.J. Chang, S. Chang, A new criterion for automatic multilevel thresholding, IEEE Transactions on Image Processing IP-4 (1995) 370–378.
[33] P.Y. Yin, L.H. Chen, New method for multilevel thresholding using the symmetry and duality of the histogram, Journal of Electronic Imaging 2 (1993) 337–344.
[34] P.Y. Yin, A discrete particle swarm algorithm for optimal polygonal approximation of digital curves, Journal of Visual Communication and Image Representation 15 (2004) 241–260.
[35] H. Yoshida, K. Kawata, Y. Fukuyama, Y. Nakanishi, A particle swarm optimization for reactive power and voltage control considering voltage stability, in: Proc. Int'l. Conf. on Intelligent System Application to Power Systems, 1999, pp. 117–121.