Multilevel Image Thresholding Selection Based on the Cuckoo Search Algorithm

Ivona BRAJEVIC, Milan TUBA, Nebojsa BACANIN
Faculty of Computer Science
University Megatrend Belgrade
Bulevar umetnosti 29, N. Belgrade
SERBIA
ivona.brajevic@googlemail.com, tuba@ieee.org, nbacanin@megatrend.edu.rs

Abstract: - The drawback of the conventional multilevel thresholding methods is high computational cost since they do exhaustive search among exponentialy growing number of possible thresholds to optimize the objective functions. In this paper a new multilevel thresholding method based on cuckoo search (CS) algorithm is proposed in order to overcome this obstacle. The optimal thresholds are found by maximizing Kapur's thresholding function. Particle swarm optimization (PSO) algorithm is also implemented and compared to our proposed method. Both algorithms have been tested on four sample images and the experimental results obtained by our proposed CS based method have been found to be better than those obtained by PSO algorithm.

Key-Words: - Maximum entropy thresholding, Image thresholding, Cuckoo search algorithm, Particle swarm optimization, Nature inspired metaheuristics, Swarm intelligence

1 Introduction

Image segmentation refers to the process of partitioning an image into multiple non-overlapping regions corresponding to meaningful background and objects. Thresholding is one of the simplest techniques for performing image segmentation that has many applications in image processing, including segmentation, classification, clustering and object discrimination [1], [2]. The global thresholding methods [3], belonging to parametric and nonparametric approaches, select thresholds by optimizing (maximizing or minimizing) some criterion functions defined from images.

Among the huge amount of image thresholding techniques, entropy-based approaches have interested many researchers [4], [5], [6], [7]. Yin [4] proposed a new method that adopts the particle swarm optimization to select the thresholds based on the minimum cross-entropy. Horng applied the honey bee mating optimization (HBMO), the artificial bee colony (ABC) algorithm [6] and the firefly algorithm [7] to search for the thresholds using the maximum entropy criterion.

Recently, a novel metaheuristic technique, called Cuckoo search (CS), based on cuckoo bird's behaviour has been developed by Yang and Deb [8]. CS algorithm was developed to solve unconstrained optimization problems, where its performance was compared with the performance of the genetic algorithm (GA) and particle swarm optimization (PSO). Simulations and comparison have shown that CS is superior to these algorithms for multimodal objective functions. Also, an object-oriented software implementation of cuckoo search was provided [9], [10] and a modified cuckoo search algorithm was implemented for unconstrained optimization problems [11], [12]. Different approaches based on CS algorithm were successfully applied to solve various optimization problems, such as engineering optimization problems [12], nurse scheduling problem [14] and Knapsack problems [15]. This paper applies the CS algorithm to search for the multilevel thresholds using the maximum entropy criterion. The PSO algorithm is implemented for purposes of comparison. Also, the exhaustive search method is conducted for deriving the optimal solutions for comparison with the results generated from PSO and CS algorithms.

The rest of the paper is organized as follows. Section 2 introduces the CS algorithm. Section 3 presents the multilevel thresholding using CS. Comparative results of the implemented CS and PSO algorithms are presented in Section 4.

This research is supported by Ministry of Science, Republic of Serbia, Project No. 44006

ISBN: 978-1-61804-119-7

2 Cuckoo search algorithm

CS is based on the brood parasitism of some cuckoo species [8]. In addition, CS algorithm is improved by the so-called Lévy flights, rather than by simple isotropic random walks. This algorithm was inspired by the aggressive reproduction strategy of some cuckoo species such as the Ani and Guira cuckoos. These cuckoos lay their eggs in communal nests, though they may remove others' eggs to increase the hatching probability of their own eggs. Quite a number of species engage the obligate brood parasitism by laying their eggs in the nests of other host birds (often other species).

The standard cuckoo search is based on three idealized rules:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
- The best nests with high-quality eggs will be carried over to the next generations.
- The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $p_\alpha \in [0,1]$. In this case, the host bird can either get rid of the egg, or simply abandon the nest and build a completely new nest.

In addition, the last assumption can be approximated by the fraction of p_α of the N host nests are replaced by new nests (with new random solutions). When generating new solutions $x^{(t+1)}$ cuckoo i, a Lévy flight is performed using the following equation:

$$x^{(t+1)} = x^{(t)} + a \wedge \text{Lévy}(\lambda) \tag{1}$$

where α (α>0) represents a step size. This step size should be related to the scales of problem the algorithm is trying to solve. In most cases, α can be set to the value 1. The above expression is in essence stochastic equation for a random walk which is a Markov chain, whose next location (status) depends on two parameters: current location (first term in Eq. 1) and probability of transition (second term in the same expression). The product ^ represents entry-wise multiplications. Something similar to entry-wise product is seen in PSO algorithm, but random walk via Lévy flight is much more efficient in exploring the search space as its step length is much longer in the long run. The random step length is drawn from a Lévy distribution which has an infinite variance with an infinite mean:

$$\text{Lévy} \sim u = t^{-\lambda} \tag{2}$$

where $\lambda \in [0.3]$.

Here the consecutive jumps (steps) of a cuckoo essentially form a random walk process which obeys a power-law step length distribution with a heavy tail.

3 Proposed approach

The proposed algorithm has two main parts. The first part implies generating the objective function based on image entropy for later developing the CS algorithm. The second phase introduces the CS algorithm for multilevel image thresholding.

3.1 Entropy criterion based measure

The multilevel thresholding problem can be configured as a k-dimensional optimization problem, for determination of k optimal thresholds $[t_1, t_2, ..., t_k]$ which optimizes an objective function. The maximum entropy criterion for image thresholding, first proposed by Pun, and later corrected and improved by Kapur have been widely used in determining the optimal thresholding [3]. Kapur has developed the algorithm for bi-level thresholding, which can also extend to solve multilevel thresholding problems and can be described as follows.

Let there be L gray levels in a given image I having M pixels and these gray levels are in the range $\{0,1,..L-1\}$. The objective function is determined from the histogram of the image, denoted by $h(i)$, $i = 0,1..L-1$, where $h(i)$ represents the number of pixels having the gray level i. The normalized probability at level i is defined by the ratio $P_i = h(i)/M$. The aim is to maximize the objective function:

$$f([t_1, t_2, ..., t_k]) = H_0 + H_1 + H_2 ... + H_k \tag{3}$$

where

$$H_0 = -\sum_{i=0}^{t_1-1} \frac{P_i}{w_0} \ln \frac{P_i}{w_0}, \quad w_0 = \sum_{i=0}^{t_1-1} P_i,$$

$$H_1 = -\sum_{i=t_1}^{t_2-1} \frac{P_i}{w_1} \ln \frac{P_i}{w_1}, \quad w_1 = \sum_{i=t_1}^{t_2-1} P_i,$$

$$H_2 = -\sum_{i=t_2}^{t_3-1} \frac{P_i}{w_2} \ln \frac{P_i}{w_2}, \quad w_2 = \sum_{i=t_2}^{t_3-1} P_i, ...$$

$$H_k = -\sum_{i=t_k}^{L-1} \frac{P_i}{w_k} \ln \frac{P_i}{w_k}, \quad w_K = \sum_{i=t_k}^{L-1} P_i$$

3.2 Image thresholding based on CS

The proposed CS algorithm based on maximum entropy criterion tries to obtain this optimum *K*-dimensional vector $[t_1, t_2, ..., t_k]$ which can maximize Eq.(3). The objective function is also used as the fitness function for the proposed algorithm. The details of the developed approach are introduced as follows.

Step 1. (Generate the initial population of solutions)
CS algorithm generates a randomly distributed initial population of *N* solutions (nests) x_i *(i = 1, 2, ..., N)* with *K* dimensions denoted by matrix *X*,

$$X = [x_1, x_2, ...x_N] \text{ and } x_i = (x_{i,1}, x_{i,2}, ...x_{i,K}) \qquad (4)$$

where x_{ij} is the j^{th} component value that is restricted into [0,…,L-1] and the $x_{ij} < x_{ij+1}$ for all *j*. The objective function values of all solutions x_i are evaluated and set cycle = 1. Before starting to iterative search process, the CS algorithm detects the most successful solution as x_{best} solution.

Step 2. (Calculate the new population)
Calculate matrix of new solutions V performing an update process for each solution in the search population *X* using the Eq. (1). For each solution v_i *(i = 1, 2, ..., N)* evaluate the objective function values by Eq.(3). If the objective function value of the new one (v_i) is higher than that of the previous one (x_i), memorize the new solution and forget the old one. Otherwise it keeps the old solution.

Step 3. (Record the best solution)
Memorize the best solution so far (x_{best}), i.e. the solution vector with the highest objective function value.

Step 4. (Fraction p_α of worse nests are abandoned and new nests are being built)
Apply the crossover operator on each solution x_i in the search population by:

$$v_i = \begin{cases} x_i + rand \cdot (x_{p_1} - x_{p_2}), & if\ rand_i > p_\alpha \\ x_i, & otherwise \end{cases} \qquad (5)$$

where *rand* is random number in [0,1] range, p_1 and p_2 are different rows permutation functions applied on nests matrix.

Step 5. (Record the best solution)
Memorize the best solution so far (x_{best}), and add the cycle by one.

Step 6. (Check the termination criterion)
If the cycle is equal to the maximum number of iterations then finish the algorithm, else go to Step 2.

4 Experimental results and discussion

The CS and PSO algorithms have been implemented in Java programming language. Four well-known images, namely House , Barbara, Boats and Living room with 256 gray levels are taken as the test images. All the images are of size (512 x 512). These original images with their histograms are shown in Fig 1. Tests were done on a PC with Intel® Core™ i3-2310M processor @2.10 GHz with 2GB of RAM and Windows 7 x64 Professional operating system. Control parameters of the CS algorithm are: the number of nests (*N*), the maximum number of iterations and discovering probability (p_α). Control parameters of the PSO algorithm are: the number of nests (N), the maximum number of iterations, inertia weight (*w*), minimum velocity ($v_{\min}$), maximum velocity ($v_{\max}$), $\phi_{\min}$ and $\phi_{\min}$.

In all experiments for both algorithms the same size of population (N) of 50 is used. In the proposed CS algorithm the maximum number of iterations is 2000, p_α is 0.7. Parameters of PSO algorithm are: the maximum number of iterations is 8000, inertia weight (*w*) is 0.5, minimum velocity ($v_{\min}$) is -5, maximum velocity ($v_{\max}$) is 5 , $\phi_{\min}$ is 0 and $\phi_{\max}$ is 2. Each experiment was repeated 30 times. The size *N* and the maximum number of iterations have a great impact on the convergence and on the computing time. As these two parameters are related, for both algorithms the same size *N* (number of nests or population size) is used. The maximum number of iterations was taken as a variable, in order to further facilitate the comparison between them for the time convergence. In order to compare the quality of the results achieved by CS and PSO algorithms for the multilevel thresholding, the value of the best fitness $F(T^*)$ corresponding to the best threshold solution T^* is used as comparative criterion. The run of each algorithm was stopped when the fitness value of the best solution $F(T^*)$ reached the optimal value of the objective function (F_{opt}), i.e. $\left|F(T^*) - F_{opt}\right| \le \varepsilon = 10^{-9}$, where ε is a threshold value which fixes the accuracy of the measurement. We have computed and recorded the iteration number and the time taken by each algorithm to achieve the desired accuracy. In that way the stopping condition for both algorithms is based on the value of the fitness and not of the number of iterations.

Table 1 shows the optimal thresholds, the optimal objective function values and the processing time provided by the exhaustive search method.

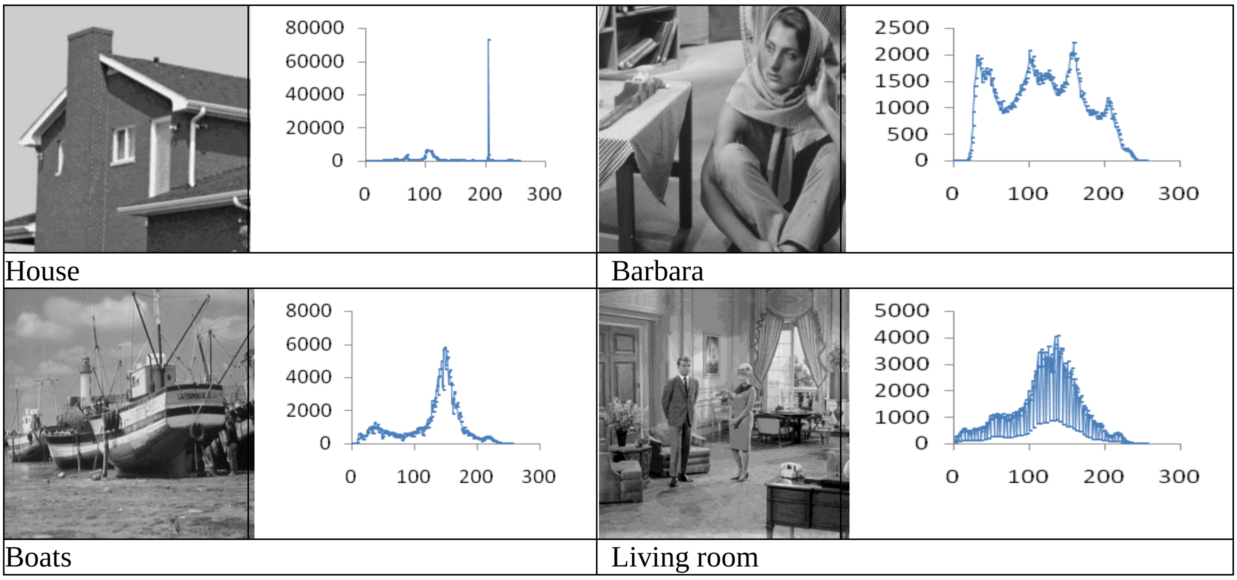

Fig 1: Test images and their histograms

Image	k	Kapur		
		Threshold values	Objective function	Time (ms)
House	1	96	7.4075657620	15
	2	95, 208	10.7304334093	827
	3	47, 97, 208	13.6166909944	30576
	4	20, 61, 98, 208	16.2329933268	1873643
Barbara	1	127	9.2012298049	15
	2	96, 168	12.6683365400	773
	3	76, 127, 178	15.7470877982	28461
	4	66, 99, 141, 185	18.5567868611	1881104
Boats	1	115	8.9642189414	22
	2	107, 176	12.5747982440	785
	3	64, 119, 176	15.8209028600	27311
	4	48, 88, 128, 181	18.6557335697	1713449
Living room	1	94	8.9194560614	24
	2	94, 175	12.4059855919	846
	3	47, 103, 175	15.5526222129	30687
	4	47, 98, 149, 197	18.4710555782	1856990

Table 1: Thresholds, objective function values and time processing provided by the exhaustive search

Table 2 presents the mean values and standard deviations over 30 runs provided by both algorithms for each image with a threshold numbers from 1 to 4, while Table 3 reports the mean number of iterations and the average of the CPU time taken by each algorithm to satisfy the stopping condition.

.

ISBN: 978-1-61804-119-7

Image	k	PSO		CS	
		Mean value	St. Dev.	Mean value	St. Dev.
House	1	7.4075657620	2.66E-15	7.4075657620	2.66E-15
	2	10.7304334093	5.33E-15	10.7304334093	5.33E-15
	3	13.6053954738	2.26E-02	**13.6166909944**	**8.88E-15**
	4	15.9463956570	2.62E-01	**16.2329933268**	**1.07E-14**
Barbara	1	9.2012298049	3.55E-15	9.2012298049	3.55E-15
	2	12.6683365400	5.33E-15	12.6683365400	5.33E-15
	3	15.7470877982	5.33E-15	15.7470877982	5.33E-15
	4	18.5547953876	1.07E-02	**18.5567868611**	**3.55E-15**
Boats	1	8.9642189414	1.55E-15	8.9642189414	3.55E-15
	2	12.5747982440	5.33E-15	12.5747982440	5.33E-15
	3	15.8205596472	1.04E-03	**15.8209028600**	**7.11E-15**
	4	18.6337909083	3.29E-02	**18.6557335697**	**1.07E-14**
Living room	1	8.9194560613	3.55E-15	8.9194560614	3.55E-15
	2	12.4057528026	1.52E-04	**12.4059855919**	**7.1E-15**
	3	15.5519647409	2.18E-03	**15.5526222129**	**8.88E-15**
	4	18.4675880405	5.53E-03	**18.4710555782**	**3.55E-15**

Table 2: Mean values and standard deviations over 30 runs

Image	k	PSO		CS	
		Time (ms)	Iteration number	Time (ms)	Iteration number
House	1	5.23	1.43	48.33	12.27
	2	12.00	6.67	426.67	172.6
	3	890.9	1611.87	719.93	276.13
	4	2939.7	5341.93	1788.27	621.87
Barbara	1	4.07	1.2	51.47	17.2
	2	3.63	7.03	309.93	132.77
	3	12.5	12.03	952.67	391.53
	4	282.33	550.9	1390.73	516.17
Boats	1	2.80	1.13	48.4	13.47
	2	11.6	8.13	275.37	108.27
	3	450.5	813	633.23	248.03
	4	3062.47	5606.2	1206.7	441.83
Living room	1	4.80	1.27	27.57	4.33
	2	3051.2	5602.4	247.8	98.47
	3	454.03	812.7	670.1	258.77
	4	1638.53	2945.9	1670.53	596.3

Table 3: Computational time of the PSO and CS

From Table 2 it can be seen that CS algorithm performs equally or better than PSO algorithm, both in terms of accuracy (mean fitness) and robustness (small standard deviation). The mean values obtained by CS algorithm are equal to the optimal objective function values derived from the exhaustive search method for each image and each threshold number. The PSO algorithm couldn't achieve the optimal solution in each run for threshold number 4 (each image), 3 (majority of images) and 2 (Boats image). In these cases, the smaller standard deviation values of CS algorithm illustrates the robustness of the proposed algorithm.

From the Table 1 it can be seen that the computation time of exhaustive search method is exponential and for $k = 4$ it is unacceptable. The reported results from the Table 3 show that as for the exhaustive search, for both algorithms, the number of iterations and the run time increase with the number of threshold, but not in the same manner. The convergence times of the CS and PSO are faster than those of the exhaustive search. For the threshold numbers from 1 to 4, for the majority of test images, we can see that PSO is more efficient in terms of computation time than CS. However, for images House (for k=3 and k=4), Boats (for k=4) and Living room (for k=2) the computational time of the PSO is longer, which means that in these cases PSO algorithm has difficulty to achieve the desired accuracy.

5 Conclusion

We propose the cuckoo search (CS) algorithm based on cuckoo bird's behaviour for multilevel thresholds selection using the maximum entropy criterion. The experimental results demonstrated that the proposed CS algorithm can search for multiple thresholds which are very close to the optimal ones determined by the exhaustive search method. Compared to the PSO, the segmentation results show that the CS algorithm outperformed PSO algorithm with respect to the solution quality and robustness. The contribution of this paper is to demonstrate the feasibility of CS method for multilevel thresholding. Also, it offers a new option to the conventional methods due to its simplicity and efficiency.

References:

[1] H.F. Ng, Automatic thresholding for defect detection, Pattern Recognition Letters, Volume 27, Issue 14, 2006, pp. 1644-1649

[2] Chen Y.L., Nighttime Vehicle Light Detection on a Moving Vehicle using Image Segmentation and Analysis Techniques, WSEAS transactions on computers, Volume 8, Issue 3, 2009., pp. 506-515

[3] M. Sezgin, B. Sankur, Survey over image thresholding techniques and quantitative performance evaluation, Journal of Electronic Imaging, Vol. 13, No. 1, 2004, pp. 146–165.

[4] P. Y. Yin, Multilevel minimum cross entropy threshold selection based on particle swarm optimization, Applied Mathematics and Computation, Volume 184, Issue 2, 2007, pp. 503-513.

[5] M.H. Horng, Multilevel Minimum Cross Entropy Threshold Selection based on Honey Bee Mating Optimization, Proc. of the 3rd WSEAS Int. Conf. on Circuits, Systems, Signal and Telecommunications (CISST'09),2009, pp. 25-30.

[6] M.H. Horng, Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation, Expert Systems with Applications, Volume 38, Issue 11, 2011, pp. 13785–13791.

[7] M.H. Horng, T.W. Jiang, Multilevel Image Thresholding Selection Based on the Firefly Algorithm, 2010 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, 2010, pp.58-63.

[8] Yang, X. S. and Deb, S., Cuckoo search via Lévy flights, in: Proc. of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009), 2009, pp. 210-214.

[9] N. Bacanin, An object-oriented software implementation of a novel cuckoo search algorithm, Proc. of the 5th European Conference on European Computing Conference (ECC'11), 2011, pp. 245-250

[10] N. Bacanin, Implementation and performance of an object-oriented software system for cuckoo search algorithm, International Journal of Mathematics and Computers in Simulation, Volume 6, Issue 1, 2012, pp. 185 - 193.

[11] M. Tuba, M. Subotic, and N. Stanarevic, Modified cuckoo search algorithm for unconstrained optimization problems, Proc. of the 5th European Conference on European Computing Conference (ECC'11), 2011, pp. 263-268.

[12] Milan Tuba, Milos Subotic, Nadezda Stanarevic: Performance of a modified cuckoo search algorithm for unconstrained optimization problems, WSEAS Transactions on Systems, Volume 11, Issue 2, February 2012, pp. 62-74

[13] Yang, X.S., and Deb, S. Engineering Optimization by Cuckoo Search,Int. J. of Mathematical Modeling and Numerical Optimization, Vol. 1, No. 4, 2010, pp.330–343.

[14] Tein L. H. and Ramli R., Recent advancements of nurse scheduling models and a potential path, in: Proc. 6th IMT-GT Conference on Mathematics, Statistics and its Applications (ICMSA 2010), 2010, pp. 395-409.

[15] A. Layeb, A novel quantum-inspired cuckoo search for Knapsack problems, Int. J. Bio-inspired Computation, Volume 3, Issue 5, 2011, pp. 297 - 305.