Applied Artificial Intelligence
An International Journal

ISSN: 0883-9514 (Print) 1087-6545 (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

A Generalized and Parallelized SSIM-Based Multilevel Thresholding Algorithm

Ikram Boubechal, Rachid Seghir & Redha Benzid

To cite this article: Ikram Boubechal, Rachid Seghir & Redha Benzid (2019) A Generalized and Parallelized SSIM-Based Multilevel Thresholding Algorithm, Applied Artificial Intelligence, 33:14, 1266-1289, DOI: 10.1080/08839514.2019.1683986

To link to this article: https://doi.org/10.1080/08839514.2019.1683986

APPLIED ARTIFICIAL INTELLIGENCE
2019, VOL. 33, NO. 14, 1266–1289
https://doi.org/10.1080/08839514.2019.1683986

Taylor & Francis
Taylor & Francis Group

ARTICLE

A Generalized and Parallelized SSIM-Based Multilevel Thresholding Algorithm

Ikram Boubechal[a], Rachid Seghir[a], and Redha Benzid[b]

[a]LaSTIC Laboratory, Department of Computer Science, University of Batna 2, Algeria; [b]LAAAS Laboratory, Department of Electronics, University of Batna 2, Algeria

ABSTRACT

Multilevel thresholding is a widely used technique to perform image segmentation. It consists of dividing an input image into several distinct regions by finding the optimal thresholds according to a certain objective function. In this work, we generalize the use of the SSIM quality measure as an objective function to solve the multilevel thresholding problem using empirically tuned swarm intelligence algorithms. The experimental study we have conducted shows that our approach, producing near-exact solutions, is more effective compared to the state-of-the-art methods. Moreover, we show that the computation complexity has been significantly reduced by adopting a shared-memory parallel programming paradigm for all the algorithms we have implemented.

Introduction

Image analysis and interpretation are widely used in various fields, such as computer vision, remote sensing, pattern recognition, medical imaging, environment modeling, etc. In this context, segmentation (Pal and Pal 1993; Zhang, Fritts, and Goldman 2008) is considered as one of the essential processing techniques which consists in dividing the input image into homogeneous regions with respect to particular characteristics, such as: color, texture structure and intensity (Fu and Mui 1981; Gonzalez and Woods 2006). The aim of such dividing is to extract the meaningful parts which are easy to analyze and interpret.

There are two fundamental categories of methods used to perform image segmentation: *Similarity-based* methods and *Discontinuity-based* algorithms. The first category partitions an image into similar regions according to a set of predefined criteria. Thresholding, region growing, region splitting and merging are classified under this category. In the second category, the image is partitioned based on abrupt changes in intensity, such as edges (Bhargavi and Jyothi 2014; Gonzalez and Woods 2006; Kaur and Kaur 2014).

CONTACT Ikram Boubechal ikram.boubechal@gmail.com LaSTIC Laboratory, Department of computer science, University of Batna 2, Batna, algeria

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/uaai.

Thresholding plays an important role in image segmentation (Haralick and Shapiro 1985). It is useful to extract objects from the background or to distinguish objects from others having distinct gray levels. Among the techniques proposed in the literature, there are those that determine the same threshold for the whole image, named: global thresholding, and those which determine a threshold for each pixel of the image, called: local or adaptive thresholding (Ballard and Brown 1982; Chehdi and Coquin 1991; Davis, Rosenfeld, and Weszka 1975; De Albuquerque, Esquef, and Mello 2004).

In global thresholding, we can distinguish between *bi-level thresholding*, the process that divides an image into two groups (binarization) using only one threshold value, and *multilevel thresholding* where the image is segmented into several classes, i.e. the process requires more than one threshold (Dey, Bhattacharyya, and Maulik 2014; Gonzalez and Woods 2006; Sun, Zhang, and Wang 2016). Furthermore, bi-level and multilevel thresholding can also be classified into *parametric* and *non-parametric* approaches (Cheriet, Said, and Suen 1998; Hou, Hu, and Nowinski 2006; Reddi, Rudin, and Keshavan 1984; Sezgin and Sankur 2004). Parametric methods assume that the histogram can be approximated using, for instance, linear combination of probability density functions (PDFs) (Snyder et al. 1990), whereas non-parametric methods are based on optimizing objective functions to search the values of optimal thresholds (Kittler and Illingworth 1986), such as Otsu between class variance (Otsu 1979) and Kapur's entropy (Kapur, Sahoo, and Wong 1985). Both Otsu and Kapur were initially developed to perform bi-level thresholding, then they were extended to solve the multilevel thresholding problem. It is worthy to note that multilevel thresholding is a challenging task because the histogram's multimodality of some images makes the selection difficult and, most importantly, it affects the effectiveness of the objective function.

More recently, the most used image-quality measures, Structural SIMilarity (SSIM) (Wang et al. 2004) and Peak Signal to Noise Ratio (PSNR) (Wang and Bovik 2009) have been used as objective functions that are optimized through different meta-heuristic algorithms to enhance the thresholds selection. To the best of our knowledge, the first use of SSIM to solve the thresholding problem was proposed in (Balabanian, Sant'Ana da Silva, and Pedrini 2017). In their work, the authors prove the efficiency of the SSIM against the Otsu approach. However, the proposed solution targets the local bi-level thresholding only. In (Kotte, Kumar, and Injeti 2018) the PSNR is used as an objective function for an improved differential search algorithm to solve the multilevel thresholding problem. Their experimental study shows that the performances of the PSNR approach are better compared to the classical Otsu and Kapur methods. However, the authors do not seem to benefit from the exhaustive search to tune their algorithms in order to further improve their obtained PSNRs. In addition, no comparison against the use of SSIM measure is reported.

Consequently, in our work, we enhance the use of SSIM and PSNR measures to achieve the best performances compared to the state-of-the-art methods. To do so, we have first generalized the use of SSIM measure to the multilevel case, in comparison to (Balabanian, Sant'Ana da Silva, and Pedrini 2017). Then, we have implemented the exhaustive search algorithm using the SSIM, PSNR, and Otsu as objective functions in order to obtain, when it is possible, the exact best solutions. Afterward, we have selected three well-known and widely used swarm intelligence algorithms PSO, Firefly, and Bat and we have empirically tuned their parameters in order to produce near-exact solutions. In this context, the implementation of the exhaustive search is interesting since it allows to check the accuracy of the meta-heuristic solutions (the closest solutions to the global optimum). But, of course, the execution time, especially when using the exhaustive search, increases exponentially according to the number of regions. Therefore, we have also implemented all the algorithms in parallel using the shared-memory parallel programming standard (OpenMP) (Dagum and Menon 1998), which has led to a significant enhancement of the algorithms' performances. The accuracy of the used approaches (employing PSNR and SSIM) are compared against Otsu-based multilevel thresholding and against each other using a set of 110 images.

The remainder of this paper is organized as follows: Section 2 briefly presents some related works. In Section 3, we formulate the problem of optimum thresholding methods and present the Otsu between-class variance method. Section 4 introduces the swarm intelligence algorithms we have used in this work, namely, PSO, Firefly, and Bat. In Section 5, we present and explain the proposed approach. Section 6 shows the experimental study we have conducted. Finally, concluding remarks are given in section 7.

Related Works

Multilevel thresholding attempts to find the optimal thresholds by optimizing an objective function like Otsu, Kapur, Tsallis, Cross Entropy, etc. (Bakhshali and Shamsi 2014; Manic, Priya, and Rajinikanth 2016; Panda et al. 2017). However, the computational time of classical exhaustive methods increases exponentially with the number of thresholds. Therefore, to overcome this weakness, meta-heuristic algorithms have been widely used to solve the multilevel thresholding problem (Horng and Jiang 2010; Sathya and Kayalvizhi 2011). Among them, we can cite: Genetic Algorithm (GA) (Hammouche, Diaf, and Siarry 2008; Manikandan et al. 2014; Muppidi et al. 2015; Yin 1999), Particle Swarm Optimization (PSO) (Liu et al. 2015; Maitra and Chatterjee 2008; Wei and Kangling 2008), Artificial Bee Colony Optimization (ABC) (Cuevas and Sossa et al. 2013; Horng and Jiang 2010), Cuckoo Search (CS) (Bhandari, Kumar, and Singh 2015; Suresh and Lal 2016), Firefly Algorithm (FF) (Vennila and Thamizhmaran 2017), etc.

Even though the previous solutions reported in the literature improve the execution time, they usually consider only the traditional objective functions (Otsu, Kapur, Tsallis, etc.). But recently, the SSIM and PSNR quality measures have been used as objective functions to solve the thresholding problem in (Balabanian, Sant'Ana da Silva, and Pedrini 2017) and (Kotte, Kumar, and Injeti 2018), respectively. In both works, it has been shown that the new approaches, using SSIM and PSNR, are more efficient than the classical Otsu and Kapur-based methods. However, the previous use of SSIM did not consider the multilevel thresholding of gray-scale images, and the use of PSNR did not show significant improvement in solution quality compared to Otsu and Kapur methods. Hence, both approaches deserve, from our point of view, to be further explored and enhanced. In our work, we benefit from the parallel computing and the exhaustive search to empirically tune existing swarm intelligence algorithms (PSO, Bat, and Firefly) in order to achieve the best multilevel thresholding accuracy compared to the ones reported in the state-of-the-art methods.

Problem Formulation of Optimum Thresholding Methods

The objective of multilevel thresholding is to find the thresholds based on the image histogram in which the segment regions of the segmented image satisfy a required property. The optimal thresholds can be determined by optimizing (minimizing or maximizing) an objective function which uses the selected thresholds as parameters. The process can be outlined as follows (Bhandari et al. 2016; Oliva and Cuevas 2016):

The segmented image is generated using quantization, where the grey level value attributed to all pixels belonging to the same class is calculated by their average.

$$\begin{array}{lll} C_1 \leftarrow p & if & 0 \le p < th_1 \\ C_2 \leftarrow p & if & th_1 \le p < th_2 \\ C_i \leftarrow p & if & th_i \le p < th_{i+1} \\ C_M \leftarrow p & if & th_m \le p < L-1 \end{array}$$

where p represents each pixel in the image I which can be denoted in grey scale level $(L = 0, 1, \ldots, 255)$ classified into $(M = m + 1)$ classes $(C_1, C_2, ..., C_M)$ using m thresholds $(th_1, th_2, ..., th_m)$.

Among the most used and efficient traditional techniques proposed to perform image thresholding, we find the Otsu between-class variance method (Otsu 1979). It aims to maximize the between-class variance Equation (1) as follows:

$$w_0 = \sum_{i=0}^{th_1-1} p_i, w_1 = \sum_{i=th_1}^{th_2-1} p_i, \ldots, w_M = \sum_{i=th_M}^{L-1} p_i$$
$$\mu_0 = \sum_{i=0}^{th_1-1} \frac{ip_i}{w_0}, \mu_1 = \sum_{i=th_1}^{th_2-1} \frac{ip_i}{w_1}, \ldots, \mu_M = \sum_{i=th_M}^{L-1} \frac{ip_i}{w_M}$$
$$\mu_T = \sum_{i=0}^{L-1} ip_i$$
$$\sigma_0 = w_0(\mu_0 - \mu_T)^2, \sigma_1 = w_1(\mu_1 - \mu_T)^2, \ldots, \sigma_M = w_M(\mu_M - \mu_T)^2$$

Where: w_i, μ_i and μ_T represent the probabilities of class occurrence, the class mean level and the total mean level of the original image, respectively.

$$th^* = \{th_1^*, th_2^*, \cdots, th_k^*, \cdots, th_{M-1}^*\} = \underset{0 \le th_k \le (L-1)}{\arg\max} \sum_{i=0}^{M} \sigma_i \tag{1}$$

Otsu has been proven as an efficient method for image thresholding. However, the computational complexity increases exponentially with the number of thresholds. Therefore, multilevel thresholding is often treated as an optimization problem solved by meta-heuristic techniques. In the following, we briefly present the swarm intelligence algorithms we are implemented in sequential and in parallel.

Swarm Intelligence

Swarm intelligence (SI) is an artificial intelligence technique based on the study of collective behavior in a decentralized and self-organized way (Beni and Wang 1993). It is widely used for solving optimization problems; a set of individuals browse the research space in order to exploit all the areas that appear promising, without being trapped by a local optimum. Several approaches have been proposed in the literature, which are originally inspired by nature; particularly the biological systems (Bonabeau, Dorigo, and Theraulaz 1999). In the following, we briefly present the swarm intelligence algorithms we have used in our work. Namely PSO, Bat, and FireFly.

Particle Swarm Optimization (PSO)

The particle swarm optimization (PSO) is a meta-heuristic method based on collective intelligence used for solving optimization problems. This method was initially developed by (Kennedy and Eberhart 1995). It was inspired from the social behavior of flocks of birds and school of fish whenever they were looking for food.

PSO (Shi and Eberhart 1999; Zhou and Shi 2011) is a population-based algorithm which uses a swarm of particles; each particle represents a potential solution to the problem of optimization. The position of every particle is influenced by the best position visited by itself ***(pbest)*** and by the position of the best particle in the whole population ***(gbest)***. It is an iterative algorithm that repeats the following steps until some stopping criteria is satisfied:

- Evaluate the fitness value of each particle.
- Update the global best fitness value and the corresponding position ***(gbest)***, do the same for the personal best position ***(pbest)***.
- Update velocity and position of each particle with equations (2) and (3).

$$v_{id}^{t+1} = w^t v_{id}^t + C_1 r_1^t (pbest_{id}^t - x_{id}^t) + C_2 r_2^t (gbest_d^t - x_{id}^t) \quad (2)$$

$$x_{id}^{t+1} = v_{id}^{t+1} + x_{id}^t \quad (3)$$

where d represents the problem's dimension (number of thresholds); x_{id}^t is the position of the i^{th} particle of the population at iteration t and v_{id}^t its velocity; C_1 and C_2 are the cognitive and social coefficients, respectively; r_1 and r_2 are two random numbers belonging to the range [0, 1]; w^t is the inertia weight defined as:

$$w^t = w_{max} - \frac{(w_{max} - w_{min})}{iterartion\_max}\ t \quad (4)$$

Firefly Algorithm

The Firefly algorithm (FF) was developed by (X.-S. Yang 2008). It is a swarm intelligence algorithm based on the brightness' emission/absorption and the mutual attractive behavior of fireflies. Theoretically, FF is based on the following three main rules:

- All fireflies are unisex, involving that every firefly of the population can attract each other.
- The attractiveness is proportional to the brightness and they both decrease as their distance increases. Thus, for any two flashing fireflies, the less bright one will move towards the brighter one. If there is no brighter one than a particular firefly, it will move randomly.
- The brightness of a firefly is determined by the objective function value.

The movement of the firefly i toward the most attractive firefly j (brighter) is calculated by:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{i,j}} (x_j^t - x_i^t) + \alpha(rand(0,1) - 0.5) \quad (5)$$

where the second term is due to the attraction that varies according to the distance between firefly i and firefly j, and α is a random number.

$$r_{i,j} = \sqrt{\sum_{d=0}^{D-1} (x_{i,d} - x_{j,d})^2} \quad (6)$$

β_0 is the attractiveness at $r = 0$.

Bat Algorithm

The bat algorithm (Bat) is a recent meta-heuristic approach proposed by Yang (X. Yang 2010; X.-S. Yang and Hossein Gandomi 2012). The idea

behind this algorithm is to mimic the echolocation behavior of micro-bats. All bats of the population are related to a location x_i and a velocity v_i, in a D dimensional search space.

The bat algorithm consists essentially of three steps:

- Calculate the current position of each bat with equation (9) if r_i^{t+1} is inferior to a number generated randomly else by (12), and its velocity with equation (8).
- Update the loudness and the pulse emission at every iteration with equations (10) and (11), respectively.
- Update the current best solution among all the bats ***(gbest)***.

$$f_i = f_{min} + (f_{max} - f_{min})\ rand(0,1) \tag{7}$$
$$v_i^{t+1} = v_i^t + (x_i^t - x^*)\, f_i \tag{8}$$
$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{9}$$
$$A_i^{t+1} = \alpha A_i^t \tag{10}$$
$$r_i^{t+1} = r_i^0(1 - e^{-\gamma t}) \tag{11}$$
$$x_i^{t+1} = x_i^t + \epsilon A^t \tag{12}$$

Where x^* is the current global best solution, A^t is the average loudness of all bats, α and γ are constants.

The main objective of the nature-inspired methods described above is to determine the maximum or the minimum of mathematical functions, called objective (fitness) functions. All these algorithms have been found to be very efficient in solving global optimization problems.

In this paper, the above algorithms are applied to evaluate the effectiveness of PSNR and SSIM objective functions in solving the multilevel thresholding problem. Parallel versions of these algorithms are also proposed and implemented in order to improve their execution time.

Proposed Approach

In our work, summarized in Figure 1, we propose to further explore and improve the use of the well-known quality metrics SSIM and PSNR as objective (fitness) functions to achieve the best multilevel thresholding results compared to those reported in (Balabanian, Sant'Ana da Silva, and Pedrini 2017) and (Kotte, Kumar, and Injeti 2018). Note that the first work uses the SSIM to solve the bi-level thresholding problem only, while the second one, based on the PSNR metric, targets the multilevel case but seems to be less efficient compared to our work, as shown in section 6. The optimal

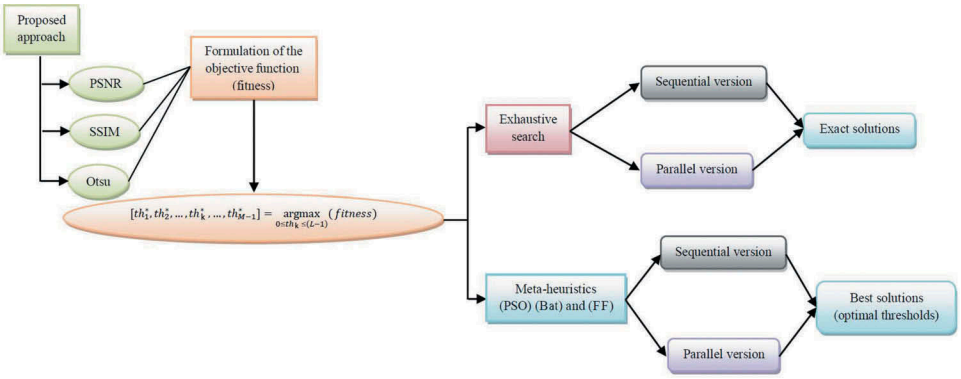


Figure 1. Overview of our work.

thresholds based on the SSIM and PSNR approaches are obtained using equations (13) and (14), respectively.

$$th^* = \{th_0^*, th_1^*, \cdots, th_k^*, \cdots, th_{M-1}^*\} = \underset{0 \le th_k \le (L-1)}{\arg\max} \; SSIM(x, y) \tag{13}$$

$$th^* = \{th_0^*, th_1^*, \cdots, th_k^*, \cdots, th_{M-1}^*\} = \underset{0 \le th_k \le (L-1)}{\arg\max} \; PSNR(x, y) \tag{14}$$

where x and y are original and segmented images, and M is the number of regions in image y.

We have first implemented a parallel exhaustive search algorithm enabling to determine the best thresholds for a given objective function (OF). By the best thresholds, we mean the ones leading to the *e*xact maximum value of the OF. This, in turn, is performed by computing the OF for all the possible combinations of thresholds and selecting the one with the best value of OF. Of course, the complexity of this computation is exponential according to the number of thresholds, but our parallel algorithm is able to produce exact solutions, in a reasonable time, when using Otsu, and until 3 and 4 thresholds when using SSIM and PSNR, respectively. Based on the solutions produced by the parallel exhaustive-search thresholding algorithm, we have tuned three swarm intelligence algorithms: PSO, Firefly, and Bat (4) so that they can produce near-exact solutions more quickly. Indeed, we have implemented these algorithms, in sequential and in parallel, to produce the best thresholds using SSIM, PSNR, and Otsu as objective functions. The PSNR and SSIM values are calculated between the given image and the thresholded one; a higher value of them indicates a better quality of thresholding. A detailed comparative study is provided in section 6. In the following, we present in some details the SSIM and PSNR metrics we have used as objective functions.

Structural SIMilarity

Structural SIMilarity index method (SSIM) (Wang et al. 2004) is a quality metric proposed to determine the structural similarity between two images. Its value ranges between 0 and 1 and indicates the correlation between the original image and the processed one, the closer it is to 1 the more the two images are similar. SSIM is defined mathematically as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 - C_1)(\sigma_{x^2} + \sigma_{y^2} + C_2)} \tag{15}$$

Where μ_x is the average of the original image x; μ_y is the average of the thresholded image y; μ_y is the average of the thresholded image y; σ_{x^2} is the variance of x; σ_{y^2} is the variance of y; σ_{xy} is the covariance of x and y; $C_1 = (k_1 \times L)^2$ and $C_2 = (k_2 \times L)^2$ $(L = 255, k_1 = 0.01$ and $k_2 = 0.03)$.

Peak Signal-to-Noise Ratio

Peak Signal-to-Noise Ratio (PSNR) is also a quality measure widely used to assess image quality. (Wang and Bovik 2009) the PSNR attempts to determine the distortion of a processed image relative to its source. The higher the PNSR is, the more the reconstructed image is similar to the original one. Due to the use of logarithm, the PSNR is expressed in decibels (dB). The PSNR value is defined as follows:

$$PSNR = 20\log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \tag{16}$$

where MSE (mean square error) is expressed as:

$$MSE = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x(i,j) - y(i,j))^2}{M \times N} \tag{17}$$

Here, x and y are original and segmented images of size $M \times N$, respectively.

Parallel Implementation of Meta-heuristic Algorithms

It is well known that the quality of the solutions produced by the meta-heuristic algorithms are improved iteratively until they reach satisfactory values. That is to say, the more the algorithm iterates, the best the obtained solution is. Unfortunately, increasing the number of iterations could lead to huge computation time. To cope with this drawback, it is usually recommended to perform the computations in parallel. In our work, we use the shared-memory parallel programming standard OpenMP (Dagum and Menon 1998) in order to reduce the execution time of the different meta-heuristics we are interested in. We have

also implemented a parallel exhaustive multilevel thresholding algorithm to obtain, when it is possible, the exact solution. This later is compared to the solutions obtained by the meta-heuristic algorithms so that it is possible to decide which of them is more accurate.

Results and Discussion

The methods proposed in our work have been implemented in C language using the OpenMP parallel programming library on Visual Studio. The test machine consists of an Intel® Core™ i56-4590S, 3.00G Hz processor with 8 G of RAM and Windows 7 × 64 Pro operating system.

Image Dataset

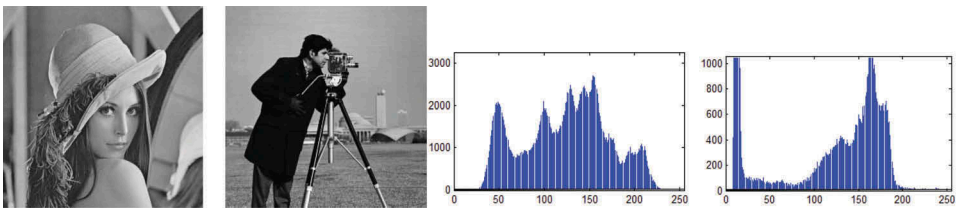
The experimental study has been carried on 110 grey level images, six of them are the standard test images 512 × 512 pixels (Lena, Barbara, Baboon, Cameraman, Pepper, and GoldHill). The rest are taken from the Berkeley Segmentation Dataset (BSD 500)[1]. Due to space limitation, detailed results are given for only two images (Lena and Cameraman) shown in Figure 2. Other results are given in terms of the mean values obtained for the 110 test images.

In all the meta-heuristic algorithms (PSO, Firefly, and Bat), we have used the same size of population (40) and applied the same number of iterations (100 iterations). The overall parameters for the three algorithms are shown in Tables 1, 2 and 3, respectively. Note that these are not the default parameter values of the algorithms, which did not give satisfactory solutions. The values we have reported are those we have found better for the multilevel thresholding problem. They are obtained by manually tuning the algorithms based on the exact solutions produced by the parallel exhaustive search algorithm.

Solution Quality Analysis

Performance Evaluation of the PSO, FF and Bat Algorithms

Tables 4, 5 and 6 present the fitness values obtained, for Lena and Cameraman images, by the different algorithms and their corresponding optimal thresholds.



(a) lena. (b) Cameraman. (c) Lena histogram. (d) Cameraman histogram.

Figure 2. Original images and their corresponding histograms.

Table 1. Parameters used for PSO.

Parameters	Value
Population size	40
max iteration	100
Lower bound	0
Upper bound	255
w_min	0.4
w_max	1.2
C_1	2
C_2	2.1
v_min	−50
v_max	50

Table 2. Parameters used for Bat.

Parameters	Value
Population size	40
max iteration	100
Lower bound	0
Upper bound	255
f_min	0.001
f_max	0.009
α	0.9
γ	0.005
A_max	50

Table 3. Parameters used for Firefly.

Parameters	Value
Population size	40
max iteration	100
Lower bound	0
Upper bound	255
α	0.9
γ	0.00001
β_0	1

The results show that the PSO algorithm generates the closest solutions to the ones obtained by the exhaustive search compared to FF and Bat algorithms. The visual results of this experiment are shown in Figures 3, 4, 5 and 6 when considering 2, 3, 4 and 5 thresholds, respectively.

The results obtained by (Kotte, Kumar, and Injeti 2018) using the PSNR as objective function for their improved differential search algorithm (IDSA) are shown in Table 7 (for Lena and Cameraman images). We can clearly see that the PSNR values obtained by their method are significantly lower than ours reported in Table 5. Indeed, the maximum values of PSNR they obtain are 22.7919 and 23.6858 for Lena and Cameraman images, respectively. While we obtain 29.52 and 29.41 for the same images. This is well justified

Table 4. Comparison of best programmed Otsu objective function values and their corresponding threshold values.

		Otsu value				Thresholds			
Test images	TH	Exhaustive	PSO	FF	Bat	Exhaustive	PSO	FF	Bat
Lena	2	1961.58	1961.58	1957.57	1961.24	93,151	93, 151	88, 152	91, 150
	3	2128.26	2128.26	2031.29	2127.31	81,127,171	81, 127, 171	96, 137, 153	79,126,168
	4	2191.60	2191.51	2168.45	2175.15	75,114,145,180	76,114,145, 180	85,129,149,183	78,113,152,195
	5	2217.59	2217.28	2184.38	2170.61	73,109,136,160,188	73,110,137,160,188	76,87,103,140,172	21,76,122,142,168
Cameraman	2	3650.33	3650.33	3628.63	3649.42	70,144	70, 144	85, 139	74, 144
	3	3725.71	3725.71	3718.49	3715.75	59,119,156	59, 119, 156	52, 125, 159	44, 116, 156
	4	3780.68	3780.66	3766.69	3765.95	42,95,140,170	43,95,140,170	23,77,140,174	37,86,133,154
	5	3812.00	3812.00	3776.11	3796.80	36,82,122,149,173	35,81,122,149,173	58,83,130,145,162	50,101,116,147,174

Table 5. Comparison of best PSNR objective function values and their corresponding threshold values.

		PSNR value				Thresholds			
Test images	TH	Exhaustive	PSO	FF	Bat	Exhaustive	PSO	FF	Bat
Lena	2	22.96	22.96	22.81	22.95	92,152	93,152	84,153	94,149
	3	26.03	26.04	25.23	25.65	79,125,171	79,126,171	75,139,169	89,138,181
	4	28.18	28.19	26.03	26.16	74,112,144,180	75,114,145,181	89,116,123,172	92,115,127,171
	5	–	29.52	26.28	28.09	–	73,110,137,160,187	66,119,171,178,184	83,106,145,174,204
Cameraman	2	24.39	24.40	24.31	24.29	69,144	70,144	78,142	80,146
	3	26.06	26.06	26.04	26.01	58,118,156	59,118,156	59,118,153	63,117,154
	4	27.87	27.88	26.66	26.85	42,95,139,170	40,93,139,170	29,83,146,158	49,107,127,151
	5	–	29.41	27.14	28.39	–	37,83,122,149,173	59,113,117,145,159	53,96,127,156,213

(_) Solution could not be reached in a reasonable time.

Table 6. Comparison of best SSIM objective function values and their corresponding threshold values.

		SSIM value				Thresholds			
Test images	TH	Exhaustive	PSO	FF	Bat	Exhaustive	PSO	FF	Bat
Lena	2	0.944850	0.945702	0.941312	0.938636	89,155	104,161	92,150	104,165
	3	0.985007	0.984276	0.963022	0.961980	85,124,175	75,129,172	86,122,169	56,111,177
	4	–	0.993289	0.981416	0.985963	–	80,109,150,184	80,121,153,176	72,113,151,169
	5	–	0.999165	0.988423	0.993838	–	80,110,142,172,196	73,120,141,186,209	68,112,144,161,177
Cameraman	2	0.972196	0.972196	0.969106	0.970382	70,143	70,143	85,128	72,146
	3	0.982389	0.982222	0.976299	0.980056	51,110,153	51,107,153	76,139,153	58,111,152
	4	–	0.989393	0.982687	0.988221	–	44,96,139,173	58,67,133,156	31,96,142,168
	5	–	0.993984	0.986314	0.989208	–	35,81,122,151,176	56,75,135,156,175	31,62,103,139,153

(–) Solution could not be reached in a reasonable time.

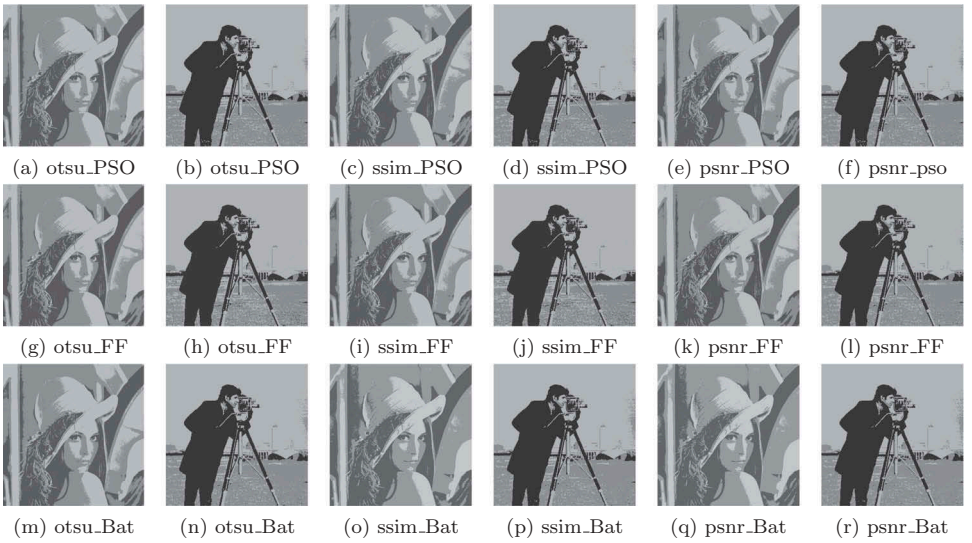


Figure 3. 2_ level thresholding for lena and cameraman images according to the thresholds obtained by the different objective functions using PSO, FF and Bat algorithms.

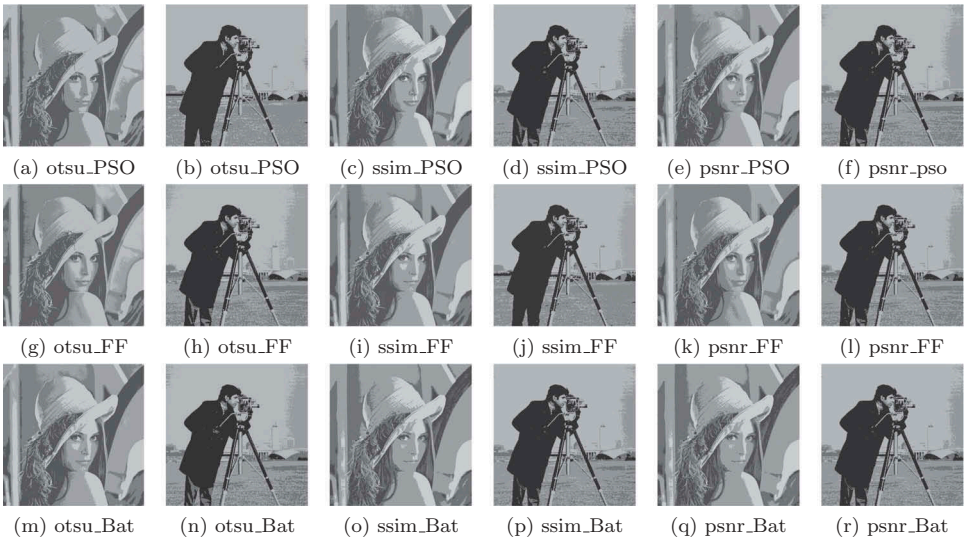


Figure 4. 3_ level thresholding for lena and cameraman images according to the thresholds obtained by the different objective functions using PSO, FF and Bat algorithms.

since the thresholds we obtain are closer to the exact thresholds compared to those produced by (Kotte, Kumar, and Injeti 2018).

Since the meta-heuristic algorithms are stochastic, the results found after each run may not be identical. Therefore, we have conducted another test to analyze the accuracy and the stability of the three algorithms using the mean and the standard deviation of the objective function values. In particular,

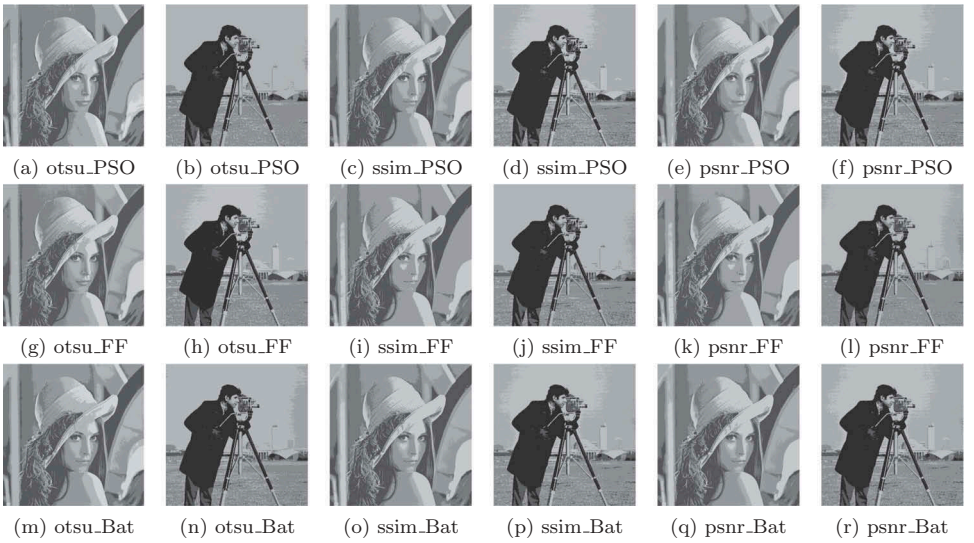


Figure 5. 4_ level thresholding for lena and cameraman images according to the thresholds obtained by the different objective functions using PSO, FF and Bat algorithms.

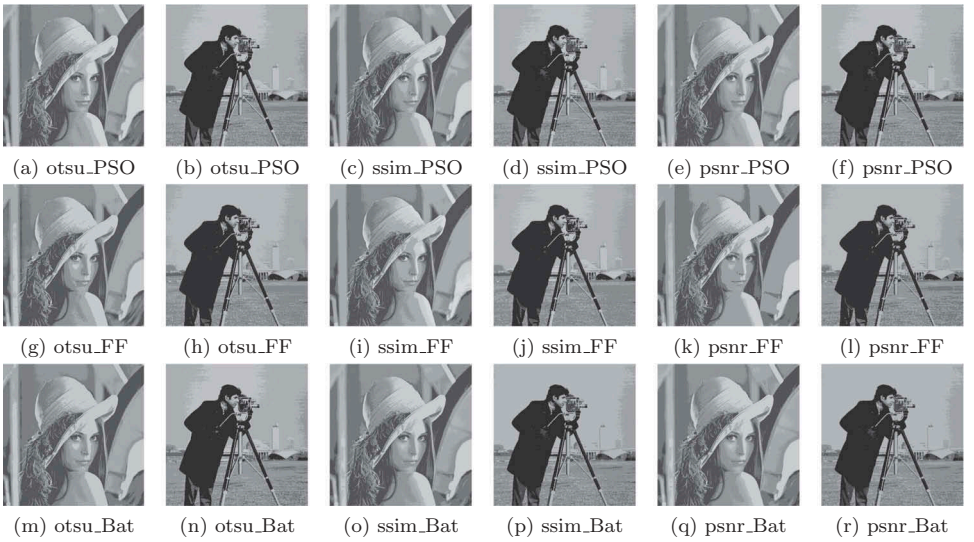


Figure 6. 5_ level thresholding for lena and cameraman images according to the thresholds obtained by the different objective functions using PSO, FF and Bat algorithms.

a lower value of standard deviation indicates higher stability, and a higher mean value of the objective function indicates a higher accuracy.

The mean values of the PSNR and SSIM objective functions are given in Table 8, where we can observe that the PSO-based method finds higher mean value of the objective functions than the Firefly and Bat-based methods. The standard deviation of the algorithms is given in Table 9, which shows that the PSO-based method has a lower standard deviation

Table 7. Results obtained by the improved differential search algorithm (IDSA) (Kotte, Kumar, and Injeti 2018).

Test image	TH	PSNR	Thresholds	Test image	TH	PSNR	Thresholds
Lena	2	16.3330	69, 121	Cameraman	2	18.5141	96, 147
	3	18.4791	59, 105, 152		3	20.6560	84, 124, 158
	4	20.7278	32, 77, 116, 159		4	22.3218	53, 100, 132, 160
	5	22.7919	31, 69, 101, 130, 172		5	23.6858	43, 92, 121, 148, 168

Table 8. Mean values of the objective functions obtained by PSO, FF, and Bat methods.

		PSNR Objective function			SSIM Objective function		
Test images	TH	PSO	FF	Bat	PSO	FF	Bat
Lena	2	22,967	22,802	22,888	0,944753	0,932601	0,938599
	3	26,042	25,293	25,589	0,981988	0,968078	0,970431
	4	28,197	26,653	27,057	0,994832	0,980758	0,980972
	5	29,487	28,056	28,052	0,998110	0,985730	0,987207
Cameraman	2	24,4	24,218	24,238	0,972141	0,968338	0,969033
	3	26,068	25,704	25,867	0,982038	0,978315	0,979081
	4	27,884	26,862	27,166	0,989312	0,984866	0,985090
	5	29,401	28,082	28,308	0,993077	0,988070	0,988952

Table 9. Standard deviation of the objective functions obtained by PSO, FF, and Bat methods.

		PSNR Objective function			SSIM Objective function		
Test images	TH	PSO	FF	Bat	PSO	FF	Bat
Lena	2	0	0,180179	0,120916	0,000936	0,009503	0,004812
	3	0,003042	0,528541	0,391998	0,001827	0,009892	0,007396
	4	0,004682	0,623804	0,558027	0,002581	0,006826	0,005618
	5	0,052787	0,729292	0,61101	0,001930	0,006903	0,004498
Cameraman	2	0	0,193174	0,209822	0,000126	0,002861	0,004518
	3	0,001672	0,279028	0,570294	0,000296	0,002729	0,001853
	4	0,004352	0,413611	0,356242	0,000398	0,002227	0,001402
	5	0,011042	0,542404	0,384759	0,000613	0,002111	0,001916

than Firefly and Bat-based methods. Hence, the PSO-based method is more stable and gives more accurate thresholds compared to the Firefly and Bat methods.

Performance of the SSIM, PSNR, and Otsu-based Approaches

In this experiment, we use the SSIM, PSNR, and Otsu as objective functions to be optimized by the PSO algorithm, since it is the one that gives the better results compared to FF and Bat algorithms as discussed before. The quality of the thresholded images is evaluated using the PSNR and SSIM metrics.

Tables 10 and 11 respectively show the PSNR and SSIM values obtained by the PSO algorithm, for Lena and Cameraman thresholded images, when using SSIM, PSNR, and Otsu as objective functions. We can notice that our generalized SSIM-based method results in better image quality compared to PSNR and Otsu. For instance (Lena with th = 3, **SSIM = 0.98154**), using **SSIM** objective function compared to Otsu (SSIM = 0.94664) and PSNR

Table 10. Comparison of PSNR values using PSO.

Test images	Th	Otsu	SSIM	PSNR
Lena	2	22.96	22.84	22.96
	3	26.03	25.52	26.04
	4	28.17	27.75	28.19
	5	29.49	28.25	29.52
Cameraman	2	24.39	24.39	24.40
	3	26.06	25.98	26.06
	4	27.88	27.82	27.88
	5	29.37	29.31	29.41

Table 11. Comparison of SSIM values using PSO.

Test images	Th	Otsu	SSIM	PSNR
Lena	2	0.92563	0.94469	0.92627
	3	0.94664	0.98154	0.97639
	4	0.98055	0.99329	0.98797
	5	0.98463	0.99672	0.98786
Cameraman	2	0.97055	0.97219	0.97055
	3	0.98079	0.98222	0.97972
	4	0.98671	0.98939	0.98710
	5	0.99253	0.99398	0.99114

(SSIM = 0.97639). This is confirmed using the whole set of test images (110 images), as shown in Figures 7 and 8 where we can clearly notice that when considering PSNR as a quality metric, the results obtained using SSIM, PSNR, and Otsu as objective functions are comparable. But when we consider SSIM as a quality metric, the results obtained using SSIM as objective function are significantly better compared to PSNR and Otsu objective functions, which confirms the advantage of using SSIM as a quality measure, as first stated in (Wang et al. 2004).

Computation Time Comparisons

Table 12 shows a comparison in terms of execution time, for Lena and Cameraman images, between PSO, FF and Bat algorithms using Otsu, SSIM and PSNR as objective functions. On the one hand, we can see that the Otsu method is significantly faster than SSIM and PSNR functions, and that SSIM is about twice slower than PSNR. On the other hand, one can notice that our implementation of PSO is slightly slower than the ones of Firefly and Bat algorithms. The comparison of computation times for the whole image dataset is given in Table 13 as the mean execution times for the 110 images. Again, we can notice that PSO is slightly slower than Firefly and Bat algorithms, and that SSIM is about twice slower than PSNR. But as we have concluded in the previous experiments, combining the PSO algorithm with SSIM as its objective function leads to the best results. Hence, it is worth considering our generalized SSIM-based method using the PSO algorithm with our tuned parameters for the problem of multilevel thresholding problem.

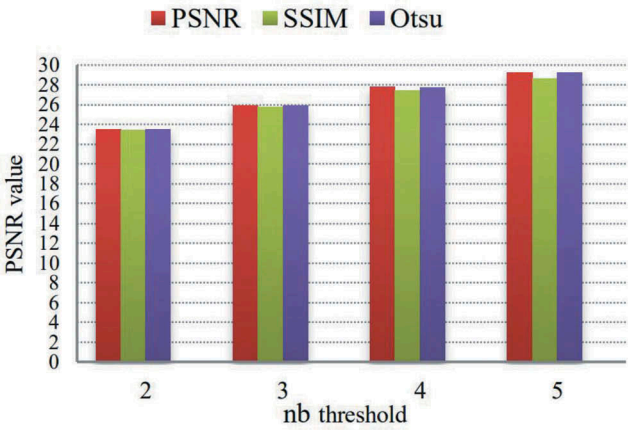


Figure 7. Average PSNR of PSO algorithm over all images (110).

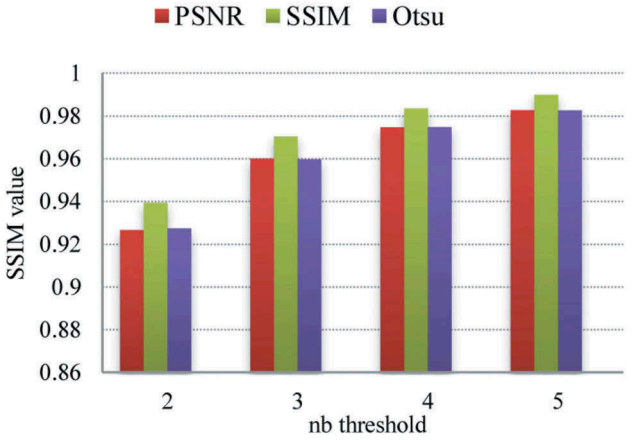


Figure 8. Average SSIM of PSO algorithm over all images (110).

Table 12. Execution time comparison between different methods using PSO, FF and Bat methods (in seconds).

		Otsu			SSIM			PSNR		
Test images	TH	PSO	FF	Bat	PSO	FF	Bat	PSO	FF	Bat
Lena	2	0.039	0.028	0.033	8.518	8.517	8.283	3.713	3.526	3.323
	3	0.040	0.031	0.036	9.375	9.063	8.955	4.508	4.368	3.775
	4	0.041	0.040	0.040	10.015	9.422	9.422	5.304	4.446	4.711
	5	0.062	0.058	0.056	10.702	10.015	9.921	5.897	4.665	5.413
Cameraman	2	0.012	0.009	0.005	2.122	2.121	2.075	0.905	0.889	0.858
	3	0.020	0.012	0.009	2.340	2.153	2.200	1.138	1.061	0.967
	4	0.016	0.014	0.012	2.621	2.309	2.371	1.341	1.108	1.108
	5	0.020	0.018	0.015	2.761	2.465	2.542	1.560	1.248	1.217

The extra computation time induced by this method can easily be coped with by adopting the parallel version of our algorithm. Indeed, we have implemented all the algorithms discussed above in parallel using the parallel programming library

Table 13. Execution time comparison for 110 images between different algorithms using PSNR and SSIM objective functions (in seconds).

	PSNR objective function			SSIM objective function		
TH	PSO	FF	Bat	PSO	FF	Bat
2	250,778	245,307	230,375	579,455	572,888	560,002
3	303,864	282,912	269,264	631,776	610,622	599,703
4	360,789	311,506	307,997	685,317	642,96	636,799
5	405,339	345,363	342,647	731,507	671,726	681,76

Table 14. Comparison of parallel execution time for 110 images between different algorithms using PSNR and SSIM objective functions (in seconds).

	PSNR objective function			SSIM objective function		
TH	PSO	FF	Bat	PSO	FF	Bat
2	90,547	87,631	86,264	194,713	191,673	187,955
3	106,221	95,796	91,385	214,324	200,116	193,249
4	128,225	104,721	100,221	240,727	217,45	221,727
5	142,949	113,028	118,544	245,388	235,097	233,637

Table 15. Obtained speedups using parallel implementation.

	PSNR Objective function			SSIM Objective function		
TH	PSO	FF	Bat	PSO	FF	Bat
2	2,7695	2,7993	2,6705	2,9759	2,9888	2,9794
3	2,8606	2,6634	2,9464	2,9477	3,0513	3,1032
4	2,8137	2,4293	3,0731	2,8468	2,9568	2,8719
5	2,8355	2,3969	2,8904	2,9810	2,8572	2,9180

OpenMP. The obtained execution times for the different algorithms and the whole test images are summarized in Table 14 where we can see that, with only four processor cores, we have gained an acceleration between 2.42 and 3.10 for all the implemented algorithms, as shown in Table 15. For instance, the acceleration of our generalized SSIM-based method using the PSO algorithm with tuned parameters is 2.98 (with th = 5). Finally, it is worth noting that these accelerations might be more important by using a more powerful machine with a higher number of processors.

Conclusion

Multilevel thresholding is among the widely used techniques in the context of image segmentation. The success or failure of image analysis and interpretation depend on the accuracy of the used segmentation technique.

In our work, we have generalized the use of the SSIM quality measure to solve the multilevel thresholding problem in order to achieve the best performances compared to the state-of-the-art methods using PSNR and Otsu as objective functions. We have implemented the exhaustive search algorithm using the SSIM, PSNR, and Otsu as objective functions in order to obtain, when it is possible, the exact best solutions. Afterward, we have

selected three well-known and widely used swarm intelligence algorithms PSO, Firefly, and Bat and we have empirically tuned their parameters in order to produce near-exact solutions. All these algorithms have also been implemented in parallel using OpenMP.

The experimental study performed on a hundred and one images has shown that the SSIM approach gives better thresholds compared to the other thresholding techniques (PSNR and Otsu). Moreover, the PSO algorithm is found to be more accurate compared to Firefly and Bat. The produced thresholds are almost the same as those obtained by the exhaustive search or very close to them. On the other hand, we have shown that the use of parallelism improves considerably the computation time of the different algorithms, which allows to obtain more precise results in quite reasonable times.

Note

1. https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/.

References

Bakhshali, M. A., and M. Shamsi. 2014. Segmentation of color lip images by optimal thresholding using bacterial foraging optimization (bfo). *Journal of Computational Science* 5 (2):251–57. doi:10.1016/j.jocs.2013.07.001.

Balabanian, F., E. Sant'Ana da Silva, and H. Pedrini. 2017. Image thresholding improved by global optimization methods. *Applied Artificial Intelligence* 31 (3):197–208.

Ballard, D. H., and C. M. Brown. 1982. *Computer vision*. 1st ed. Englewood Cliffs, NJ: Prentice Hall Professional Technical Reference.

Beni, G., and J. Wang. 1993. Swarm intelligence in cellular robotic systems. In *Robots and biological systems: Towards a new bionics?* ed. P. Dario, G. Sandini, and P. Aebischer, 703–12. Berlin, Heidelberg: Springer Berlin Heidelberg.

Bhandari, A., A. Kumar, and G. Singh. 2015, December. Tsallis entropy based multilevel thresholding for colored satellite image segmentation using evolutionary algorithms. *Expert Systems with Applications* 42 (22):8707–30. doi:10.1016/j.eswa.2015.07.025.

Bhandari, A., A. Kumar, S. Chaudhary, and G. Singh. 2016, November. A novel color image multilevel thresholding based segmentation using nature inspired optimization algorithms. *Expert Systems with Applications* 63 (C):112–33. doi:10.1016/j.eswa.2016.06.044.

Bhargavi, K., and S. Jyothi. 2014. A survey on threshold based segmentation technique in image processing. *International Journal of Innovative Research and Development* 3 (12):234–39.

Bonabeau, E., M. Dorigo, and G. Theraulaz. 1999. *Swarm intelligence: From natural to artificial systems*. New York, NY, USA: Oxford University Press, Inc.

Chehdi, K., and D. Coquin. 1991, May. Binarisation of various images by detecting local thresholds with a validation test. [1991] ieee pacific rim conference on communications, computers and signal processing conference proceedings, vol.2, Victoria, BC, Canada, 611–14.

Cheriet, M., J. N. Said, and C. Y. Suen. 1998, June. A recursive thresholding technique for image segmentation. *IEEE Transactions on Image Processing* 7 (6):918–21. doi:10.1109/83.679444.

Dagum, L., and R. Menon. 1998, January. Openmp: An industry standard api for shared-memory programming. *IEEE Computational Science and Engineering* 5 (1):46–55. doi:10.1109/99.660313.

Davis, L. S., A. Rosenfeld, and J. S. Weszka. 1975, May. Region extraction by averaging and thresholding. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-5 (3):383–88. doi:10.1109/TSMC.1975.5408419.

De Albuquerque, M. P., I. A. Esquef, and A. G. Mello. 2004. Image thresholding using tsallis entropy. *Pattern Recognition Letters* 25 (9):1059–65. doi:10.1016/j.patrec.2004.03.003.

Dey, S., S. Bhattacharyya, and U. Maulik. 2014, November. Quantum behaved multi-objective pso and aco optimization for multi-level thresholding. 2014 international conference on computational intelligence and communication networks, Bhopal, India, 242–46.

Fu, K.-S., and J. Mui. 1981. A survey on image segmentation. *Pattern Recognition* 13 (1):3–16. doi:10.1016/0031-3203(81)90028-5.

Gonzalez, R. C., and R. E. Woods. 2006. *Digital image processing. (3rd* ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Hammouche, K., M. Diaf, and P. Siarry. 2008, February. A multilevel automatic thresholding method based on a genetic algorithm for a fast image segmentation. *Computer Vision and Image Understanding* 109 (2):163–75. doi:10.1016/j.cviu.2007.09.001.

Haralick, R. M., and L. G. Shapiro. 1985. Image segmentation techniques. *Applications of Artificial Intelligence Ii* 548:2–10.

Horng, M.-H., and T.-W. Jiang. 2010. Multilevel image thresholding selection using the artificial bee colony algorithm. Proceedings of the 2010 international conference on artificial intelligence and computational intelligence: Part ii, 318–25, Berlin, Heidelberg: Springer-Verlag. http://dl.acm.org/citation.cfm?id=1926560.1926609

Hou, Z., Q. Hu, and W. L. Nowinski. 2006, October. On minimum variance thresholding. *Pattern Recognition Letters* 27 (14):1732–43. doi:10.1016/j.patrec.2006.04.012.

Kapur, J. N., P. K. Sahoo, and A. K. Wong. 1985. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing* 29 (3):273–85. doi:10.1016/0734-189X(85)90125-2.

Kaur, D., and Y. Kaur. 2014. Various image segmentation techniques: A review. *International Journal of Computer Science and Mobile Computing* 3 (5):809–14.

Kennedy, J., and R. Eberhart. 1995, November. Particle swarm optimization. Proceedings of icnn'95 - international conference on neural networks, vol. 4, Perth, WA, Australia, 1942–48.

Kittler, J., and J. Illingworth. 1986, January. Minimum error thresholding. *Pattern Recognition* 19 (1):41–47. doi:10.1016/0031-3203(86)90030-0.

Kotte, S., P. R. Kumar, and S. K. Injeti. 2018. An effcient approach for optimal multilevel thresholding selection for gray scale images based on improved differential search algorithm. *Ain Shams Engineering Journal* 9 (4):1043–67. doi:10.1016/j.asej.2016.06.007

Liu, Y., C. Mu, W. Kou, and J. Liu. 2015, May. Modified particle swarm optimization-based multilevel thresholding for image segmentation. *Soft Computing* 19 (5):1311–27. doi:10.1007/s00500-014-1345-2.

Maitra, M., and A. Chatterjee. 2008, February. A hybrid cooperative- comprehensive learning based pso algorithm for image segmentation using multilevel thresholding. *Expert Systems with Applications* 34 (2):1341–50. doi:10.1016/j.eswa.2007.01.002.

Manic, K. S., R. K. Priya, and V. Rajinikanth. 2016. Image multithresholding based on kapur/tsallis entropy and firefly algorithm. *Indian Journal of Science and Technology* 9 (12). doi:10.17485/ijst/2016/v9i12/89949.

Manikandan, S., K. Ramar, M. W. Iruthayarajan, and K. Srinivasagan. 2014. Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm. *Measurement* 47:558–68. doi:10.1016/j.measurement.2013.09.031.

Muppidi, M., P. Rad, S. S. Agaian, and M. Jamshidi. 2015, November. Image segmentation by multi-level thresholding using genetic algorithm with fuzzy entropy cost functions. 2015 international conference on image processing theory, tools and applications (ipta), Orleans, France, 143–48.

Oliva, D., and E. Cuevas. 2016. *Advances and applications of optimised algorithms in image processing*. 1st ed. Cham, Switzerland: Springer Publishing Company, Incorporated.

Osuna-Enciso, V., E. Cuevas, and H. Sossa. 2013. A comparison of nature inspired algorithms for multi-threshold image segmentation. *Expert Syst. Appl.* 40 (4):1213–19. http://dx.doi.org/10.1016/j.eswa.2012.08.017

Otsu, N. 1979, January. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* 9 (1):62–66. doi:10.1109/TSMC.1979.4310076.

Pal, N. R., and S. K. Pal. 1993. A review on image segmentation techniques. *Pattern Recognition* 26 (9):1277–94. doi:10.1016/0031-3203(93)90135-J.

Panda, R., S. Agrawal, L. Samantaray, and A. Abraham. 2017, January. An evolutionary gray gradient algorithm for multilevel thresholding of brain mr images using soft computing techniques. *Applied Soft Computing* 50 (C):94–108. doi:10.1016/j.asoc.2016.11.011.

Reddi, S. S., S. F. Rudin, and H. R. Keshavan. 1984, July. An optimal multiple threshold scheme for image segmentation. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-14 (4):661–65. doi:10.1109/TSMC.1984.6313341.

Sathya, P. D., and R. Kayalvizhi. 2011, November. Optimal multilevel thresholding using bacterial foraging algorithm. *Expert Systems with Applications* 38 (12):15549–64. doi:10.1016/j.eswa.2011.06.004.

Sezgin, M., and B. Sankur. 2004. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging* 13 (1):146–66. doi:10.1117/1.1631315.

Shi, Y., and R. C. Eberhart, 1999, July. Empirical study of particle swarm optimization. Proceedings of the 1999 congress on evolutionary computation-cec99 (cat. no. 99th 8406), vol. 3, Washington, DC, USA, 1945–50.

Snyder, W., G. Bilbro, A. Logenthiran, and S. Rajala. 1990, December. Optimal thresholding—A new approach. *Pattern Recognition Letters* 11 (12):803–10. doi:10.1016/0167-8655(90)90034-Y.

Sun, G., A. Zhang, and Z. Wang. 2016. Grayscale image segmentation using multilevel thresholding and nature-inspired algorithms. In *Hybrid soft computing for image segmentation*, 23–52. Cham: Springer, International Publishing AG 2016. https://doi.org/10.1007/978-3-319-47223-22

Suresh, S., and S. Lal. 2016, October. An efficient cuckoo search algorithm based multilevel thresholding for segmentation of satellite images using different objective functions. *Expert Systems with Applications* 58 (C):184–209. doi:10.1016/j.eswa.2016.03.032.

Vennila, K., and K. Thamizhmaran. 2017. Implementation of multilevel thresholding on image using firefly algorithm. *International Journal of Advanced Research in Computer Science* 8 (3):373–78.

Wang, Z., and A. C. Bovik. 2009, January. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE Signal Processing Magazine* 26 (1):98–117. doi:10.1109/MSP.2008.930649.

Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. 2004, April. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing* 13 (4):600–12. doi:10.1109/TIP.2003.819861.

Wei, C., and F. Kangling. 2008, July. Multilevel thresholding algorithm based on particle swarm optimization for image segmentation. 2008 27th chinese control conference, Kunming, China, 348–51. doi:10.1039/b711199a.

Yang, X. 2010. A new metaheuristic bat-inspired algorithm nature inspired cooperative strategies for optimization (nicso 2010) 2010 284 berlin. *Germany Springer* 65:74.

Yang, X.-S. 2008. *Nature-inspired metaheuristic algorithms*. University of Cambridge, UK: Luniver Press.

Yang, X.-S., and A. Hossein Gandomi. 2012. Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations* 29 (5):464–83. doi:10.1108/02644401211235834.

Yin, P.-Y. 1999, January. A fast scheme for optimal thresholding using genetic algorithms. *Signal Processing* 72 (2):85–95. doi:10.1016/S0165-1684(98)00167-4.

Zhang, H., J. E. Fritts, and S. A. Goldman. 2008, May. Image segmentation evaluation: A survey of unsupervised methods. *Computer Vision and Image Understanding* 110 (2):260–80. doi:10.1016/j.cviu.2007.08.003.

Zhou, Z., and Y. Shi. 2011. Inertia weight adaption in particle swarm optimization algorithm. Proceedings of the second international conference on advances in swarm intelligence - volume part I, 71–79, Berlin, Heidelberg: Springer-Verlag. http://dl.acm.org/citation.cfm?id=2026282.2026293