Multimed Tools Appl
DOI 10.1007/s11042-016-3891-3

A hybrid bio-inspired learning algorithm for image segmentation using multilevel thresholding

Mohammad Mahdi Dehshibi[1] **· Mohamad Sourizaei**[2] **·**
Mahmood Fazlali[3] **· Omid Talaee**[4] **·**
Hossein Samadyar[1] **· Jamshid Shanbehzadeh**[5]

Received: 26 September 2015 / Revised: 10 August 2016 / Accepted: 22 August 2016

Abstract In the field of image analysis, segmentation is one of the most important preprocessing steps. One way to achieve segmentation is the use of threshold selection, where each pixel that belongs to a determined class, based on the mutual visual characteristics, is labeled according to the selected threshold. In this work, a combination of two pioneer methods, namely Otsu and Kapur, are investigated to solve the threshold selection problem. Optimum parameters of these objective functions are calculated using Bacterial Foraging (BF) optimization algorithm, for its accuracy, and Harmony Search (HS), for its speed. However, the biggest problem of soft computing family algorithms is catching into a local optimum. To resolve this critical issue, we investigate the power of Learning Automata (LA) which works as a controller to make switching between these two optimization methods. LA is a heuristic method which can solve complex optimization problems with interesting results in parameter estimation. Despite other techniques commonly seek through the parameter map, LA explores in the probability space, providing appropriate convergence properties and robustness. The proposed method is

✉ Mohammad Mahdi Dehshibi
dehshibi@iranprc.org; mohammad.dehshibi@yahoo.com

✉ Mohamad Sourizaei
sourizaei.m@gmail.com

1 Department of Computer Engineering, Faculty of Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

2 Young Researchers and Elite Club, Zahedan Branch, Islamic Azad University, Zahedan, Iran

3 Department of Computer Science, Faculty of Mathematics, Shahid Beheshti University, G. C, Tehran, Iran

4 Department of Medical Engineering, Faculty Engineering, University of Shiraz, Shiraz, Iran

5 Department of Computer Science, Faculty of Engineering, Kharazmi University, Tehran, Iran

Published online: 21 September 2016

tested on benchmark images and shows fast convergence avoiding the typical sensitivity to initial conditions such as the Expectation-Maximization (EM) algorithm or the complex, and time-consuming computations which are commonly found in gradient methods. Experimental results demonstrate the algorithm's ability to perform automatic multi-threshold selection and show interesting advantages as it is compared to other algorithms solving the same task.

Keywords Multilevel thresholding · Image segmentation · Hybrid optimization · Kapur function · Otsu function

1 Introduction

Image segmentation facilitates the separation of spatial-spectral attributes contained in images into their individual constituents; a task that is accomplished quite comfortably by our visual system and cortical mechanisms. However, mimicking this capability of human observers in an artificial environment has been found to be an extremely challenging problem. Formally, image segmentation is defined as the process of partitioning or segregating an image into regions (also called as clusters or groups), manifesting homogeneous or nearly homogeneous attributes such as color, texture, gradient and spatial attributes about location. Fundamentally, a segmentation algorithm for an image is said to be "complete" when it provides a unique region or label assignment for every pixel, such that all pixels in a segmented region satisfy certain criteria while the same principles are not universally satisfied for pixels from disjoint regions.

In the context of imagery, segmentation is often viewed as an ill-defined problem with no perfect solution but multiple acceptable solutions due to its subjective nature [23]. The subjectivity of segmentation has been extensively substantiated in experiments conducted at the University of California at Berkeley [14] to develop an evaluation benchmark, where a database of manually generated segmentations of images with natural content was developed using multiple human observers. In Fig. 1a, two images from the database as mentioned earlier are displayed. Additionally, several manually segmented ground truths with region boundaries superimposed (in green) on the original image are shown in Fig. 1b to f. Analysis of the obtained ground truth results by Martin et al. [14] divulged two imperative aspects: (I) an arbitrary image may have a unique suitable segmentation outcome while others possess multiple acceptable solutions, and (II) the variability of inadequate solutions is primarily due to the differences in the level of attention (or granularity) and the degree of detail from one human observer to another, as seen in Fig. 1. Consequently, most present day algorithms for segmentation aim to provide acceptable outcomes rather than a "gold standard" solution.

The most segmentation modus operandi can be viewed as being either spatially blind or spatially guided. Spatially blind approaches perform segmentation in certain attribute/feature spaces, predominantly related to intensity. Popular segmentation techniques that fall within the notion of being spatially blind involve clustering [4, 25] and histogram thresholding [20]. In contrast to spatially blind methods, spatially guided approaches, as the name suggests, are guided by spatial relationships of pixels for segmentation. Their primary objective is to form pixel groupings that are compact or

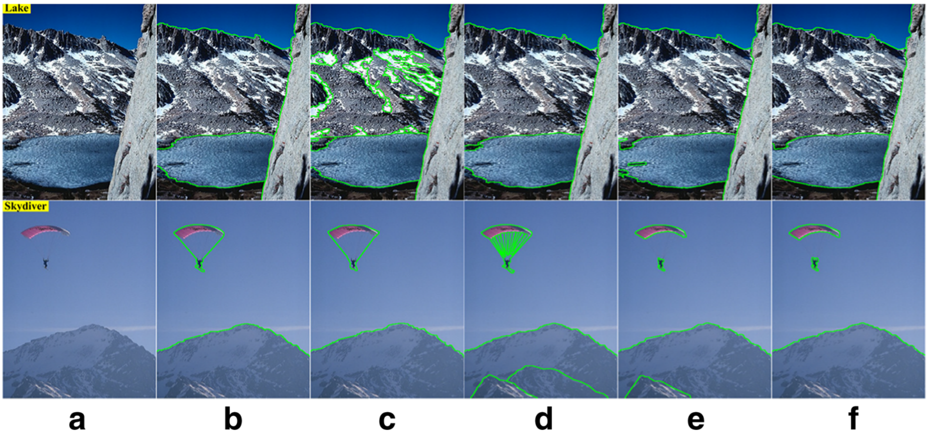


Fig. 1 Berkeley segmentation benchmark [14] (**a**) original images, and (**b**) to (**f**) region boundaries of multiple manually generated segmentations overlaid on the images

homogeneous from a spatial standpoint, irrespective of their relationships in specific feature spaces. However, despite the development of many spatially guided techniques, the use of region and edge information explicitly or in an integrated framework remains widely-accepted alternatives for the formation of spatially compact regions. Segmentation approaches such as region-based [6], energy-based [18] and region and contour based [8, 15] fall within the notion of being spatially guided.

Histogram thresholding [20] is a spatially blind technique primarily based on the principle that segments of an image can be identified by delineating peaks, valleys, and/or shapes in its corresponding intensity histogram. Similar to clustering, histogram thresholding protocols require minimal effort to realize in comparison with most other segmentation algorithms and function without the need for any a priori information about the image being partitioned. Owed to its simplicity, intensity histogram thresholding initially gained popularity for segmenting gray-scale images which also has the capability of converting a gray-scale image into binary one.

In general, if the gray level histogram of the image is bi-modal, the image objects are clearly distinguishable from the background. In this case, it is easy to choose a threshold value by taking the value that is in the valley between two peaks of the histogram. However, in the real world, the gray level histograms of the images are always multi-modal and; hence, it is not simple to determine the exact locations of distinct valleys in multi-modal histograms. Thanks to growing of evolutionary algorithm, the problem of finding the optimal threshold makes easier considering a specific target function [27]. Vantaram and Saber [23] have presented a thorough survey of a variety of thresholding techniques, among which global histogram based algorithms are widely employed to determine the threshold. This global thresholding technique can be classified into parametric and nonparametric approaches. In parametric approaches [5], the gray level distribution of each class has a probability density function that follows a Gaussian distribution. This method is computationally expensive and time-consuming. Non-parametric approaches determine the threshold values in an optimal fashion based on a given criterion. The nonparametric approaches such as Otsu [17] and Kapur [11] are robust and more accurate than the parametric methods.

The Otsu and Kapur methods can be easily extended to multilevel thresholding problem but inefficient in determining the optimal thresholds due to the exponential growth in computation time. To improve the efficiency, many methods have been proposed for solving the multilevel thresholding problem [35]. Liao et al. [13] showed that the recursive algorithm significantly reduces the computational complexity of determining the multi-level thresholds by accessing a look-up table when compared with conventional Otsu and Kapur methods. However, it still suffers from the problem of a significant processing time when the number of thresholds increases.

To eliminate such problems, numerous works on the topic has been presented based on swarm algorithms, including Genetic Algorithm (GA) [7], Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC) [1], Ant Colony Optimization (ACO) [21], Bacterial Foraging (BF) [19], and Honey Bee Mating Optimization (HBMO) [9]. Yin et al. [29] have proposed a PSO based multilevel minimum cross entropy threshold selection method to achieve near-optimal thresholds. Zhang and Wu [33] used ABC algorithm for optimizing Tsallis entropy. After that, Akay [17] employed PSO and ABC to find the optimal multilevel thresholds. Kapur's entropy, one of the maximum entropy techniques, and between-class variance has been investigated as fitness functions. The results of performing this algorithm on a set of test image, using various numbers of thresholds, were assessed using statistical tools and suggest that Otsu's technique, PSO and ABC show equal performance when the number of thresholds is two, while the ABC algorithm performs better than PSO and Otsu's technique when the number of thresholds is greater than two.

Yang and Deb [26] formulated a new meta-heuristic algorithm, called cuckoo search algorithm which is based on the interesting breeding behavior such as brood parasitism of particular species of Cuckoos, and the preliminary studies show that it is very promising and could outperform the existing algorithms such as GA, PSO, ABC, ACO, BF and HBMO [5].

All of these methods have their limitations concerning convergence speed or accuracy, and all researchers aim at finding a tradeoff between these two important aspects. Besides the mentioned limitations, there is a shortcoming with which escaping from is rather impossible; it is local optima. The main idea behind this research is to cope with this deficiency while increasing the accuracy of optimization. It is evident that each optimization may face different local solutions; hence, merging two optimization algorithms has a high capability of escaping from individual local solutions. To reach a robust multilevel thresholding, BF, with its accuracy and Harmony Search (HS), with its convergence speed, [3] are joined to be able of approaching the problem of mentioned local optimum. To do switching between BF and HS, we have investigated the power of a sort of reinforcement learning algorithm, namely Learning Automata (LA) [16], to give a reward in case of right behavior as well as providing a penalty when the algorithm faces local optimum. The presented scheme is used for maximizing the Kapur's entropy and Otsu. Experimental image thresholding results are obtained for qualitative analysis. Quantitative results are defined by peak signal-to-noise ratio (PSNR), structural similarity index (SSIM) and feature similarity index (FSIM). The performance improvement of the proposed algorithm, so-called BFHS, based segmentation approach is measured regarding PSNR, SSIM, and FSIM, in comparison with the state of the art techniques.

The rest of this paper is organized as follows. Section 2 dedicates to formulate the foundations of multilevel thresholding. Brief explanations of the algorithms that are used in this study are given in Section 3. Implementation of the hybrid method is described in Section 4. The experimental results are provided in Section 5, and, finally, the conclusion remarks are given in Section 6.

2 Problem formulation for multilevel thresholding

The optimal thresholding methods search the thresholds such that the segmented classes on the histogram satisfy the desired property. This is performed by maximizing an objective function which uses the selected thresholds as the parameters. In this paper, two broadly used optimal thresholding methods namely entropy criterion (Kapur) method and between-class variance (Otsu) methods are used.

2.1 Kapur method

The entropy criterion method has been employed in determining whether the optimal thresholding can provide histogram-based image segmentation with satisfactory desired characteristics [11, 17]. Kapur has developed the algorithm for bi-level thresholding, and this bi-level thresholding can be described as follows:

Let there be L gray levels in a given image and let them be in a range $\{0, 1, 2, \ldots, (L - 1)\}$. Then one can define $p_i = h(i) \times N$, $(0 \le i \le (L - 1))$ where $h(i)$ denotes the number of pixels for the corresponding gray-level L, and N denotes total number of pixels in the image which is equal to $\sum_{i=0}^{L-1} h(i)$. Then the objective is to maximize the fitness function.

$$f(t) = H_0 + H_1 \tag{1}$$

where $H_0 = -\sum_{i=0}^{t-1}\left(\frac{p_i}{\omega_0}\ln\frac{p_i}{\omega_0}\right)$, $\omega_0 = \sum_{i=0}^{t-1} P_i$, $H_1 = -\sum_{i=t}^{L-1}\left(\frac{p_i}{\omega_1}\ln\frac{p_i}{\omega_1}\right)$, $\omega_1 = \sum_{i=t}^{L-1} p_i$.

The optimal threshold is the gray level that maximizes Eq. 1. This Kapur's entropy criterion method tries to achieve a centralized distribution for each histogram-based segmented region of the image. This method is extended to multilevel thresholding as follows:

The optimal multilevel thresholding problem can be configured as an m-dimensional optimization problem, for determination of m optimal thresholds for a given image $[t_1, t_2, \ldots, t_m]$, where the aim is to maximize the objective function:

$$f([t_1, t_2, \ldots, t_m]) = H_0 + H_1 + \ldots + H_m \tag{2}$$

where $H_0 = -\sum_{i=0}^{t_1-1}\left(\frac{p_i}{\omega_0}\ln\frac{p_i}{\omega_0}\right)$, $\omega_0 = \sum_{i=0}^{t_1-1} p_i$, …, $H_m = -\sum_{i=t_m}^{L-1}\left(\frac{p_i}{\omega_m}\ln\frac{p_i}{\omega_m}\right)$, $\omega_m = \sum_{i=t_m}^{L-1} p_i$.

2.2 Otsu method

Thresholding using Otsu's method is a nonparametric segmentation technique, which is used to segment the entire image into many regions; as a result, the variance of the various classes can be maximized. Between-class variance was proposed by [17] as sum of sigma functions of each class and is defined by Eq. (3):

$$f(t) = \sigma_0 + \sigma_1 \tag{3}$$

$$\sigma_0 = \omega_0(\mu_0 - \mu_T)^2.\ \sigma_1 = \omega_1(\mu_1 - \mu_T)^2 \tag{4}$$

where μ_T represents the mean intensity of input image. In the case of bi-level thresholding, the average level of each class (μ_i), can be obtained using Eq. (5):

$$\mu_0 = \sum_{i=0}^{t-1} \frac{ip_i}{\omega_0}. \quad \mu_1 = \sum_{i=t}^{L-1} \frac{ip_i}{\omega_1}. \tag{5}$$

By maximizing the between-class variance function, the optimal threshold value can be achieved using Eq. (6):

$$t^* = \operatorname{argmax}(f(t)) \tag{6}$$

Furthermore, between-class-variance [1] is extended to multilevel thresholding problem as followed by Eq. (7):

$$f(t) = \sum_{i=0}^{m} \sigma_i \tag{7}$$

The sigma term and the mean levels can be obtained from Eq. (8):

$$\begin{gathered}\sigma_0 = \omega_0(\mu_0 - \mu_T)^2. \ \sigma_1 = \omega_1(\mu_1 - \mu_T)^2. \sigma_j = \omega_j\left(\mu_j - \mu_T\right)^2. \sigma_m = \omega_m(\mu_m - \mu_T)^2 \\ \mu_0 = \sum_{i=0}^{t_1-1} \frac{ip_i}{\omega_0}. \ \mu_1 = \sum_{i=t_1}^{t_2-1} \frac{ip_i}{\omega_1}. \mu_j = \sum_{i=t_j}^{t_{j+1}-1} \frac{ip_i}{\omega_j}. \ \mu_m = \sum_{i=t_m}^{L-1} \frac{ip_i}{\omega_m}.\end{gathered} \tag{8}$$

The optimal multilevel thresholding is configured by maximizing the objective function using Eq. (9):

$$\left(\vec{t}\right)^* = \operatorname{argmax}\left(\sum_{i=0}^{m} \sigma_i\right) \tag{9}$$

The Kapur and Otsu methods have been proven as an efficient method for bi-level thresholding in image segmentation. However, when these methods are extended to multilevel thresholding, the computation time grows exponentially with the number of thresholds. It would limit the multilevel thresholding applications. To overcome the above problem, this paper proposes a hybrid Bio-Inspired learning algorithm for solving multi-level thresholding problem. The aim of the proposed method is to maximize the Kapur and Otsu objective functions.

3 Brief explanations of the algorithms in the study

Optimization is the process of making something better than the previous form. Over the last decade, the aggregate intelligent behavior of insect or animal groups in the natural world for example flocks of birds, colonies of ants, schools of fish, swarms of bees, and termites have fascinated the interest of researchers. The collective action of insects, birds or animals is identified as swarm behavior. Many researchers have used swarm behavior as a framework for solving complicated real-world problems.

The aim of this research is to apply the accuracy of BF and the speed of HS to satisfy the target function. LA acts as the intelligent part of the algorithm and, on current conditions of the problem, selects one of the two algorithms so as to reach a satisfaction in optimizing target function.

3.1 Bacterial foraging optimization algorithm

The Bacterial Foraging Optimization Algorithm (BF) [3] is inspired by the group foraging behavior of bacteria such as E.coli and M.xanthus. Specifically, the BF is inspired by the chemotaxis behavior of bacteria that will perceive chemical gradients in the environment (such as nutrients) and move toward or away from specific signals.

The information processing strategy of the algorithm is to allow cells to stochastically and collectively swarm toward the ideal situations. This is achieved through a series of three processes on a population of simulated cells: 1) 'Chemotaxis' where the cost of cells is derated by the proximity to other cells and cells move along the manipulated cost surface one at a time (the majority of the work of the algorithm), 2) 'Reproduction' where only those cells that performed well over their lifetime may contribute to the next generation, and 3) 'Elimination-dispersal' where cells are discarded, and new random samples are inserted with a low probability.

A bacteria cost is derated by its interaction with other cells and is calculated as follows:

$$\begin{aligned} g(cell_k) = &\sum_{i=1}^{S}\left[-d_{attr}\times\exp\left(-w_{attr}\times\sum\nolimits_{m=1}^{P}\left(cell_m^k - other_m^i\right)^2\right)\right] \\ &+\sum_{i=1}^{S}\left[-h_{repel}\times\exp\left(-w_{repel}\times\sum\nolimits_{m=1}^{P}\left(cell_m^k - other_m^i\right)^2\right)\right]. \end{aligned} \tag{10}$$

where $cell_k$ is a given cell, d_{attr} and w_{attr} are attraction coefficients, h_{repel} and w_{repel} are repulsion coefficients, S is the number of cells in the population, P is the number of dimensions on a given cells position vector.

The remaining parameters of the algorithm are as follows: $Cells_{num}$ is the number of cells maintained in the population, N_{ed} is the number of elimination-dispersal steps, N_{re} is the number of reproduction steps, N_c is the number of chemotaxis steps, N_s is the number of swim steps for a given cell, $Step_{size}$ is a random direction vector with the same number of dimensions as the problem space, and each value $\in [-1,1]$, and P_{ed} is the probability of a cell being subjected to elimination and dispersal.

3.2 Harmony search algorithm

Harmony Search [3] was inspired by the improvisation of Jazz musicians. Specifically, the process in which the musicians (who may have never played together before) rapidly refine their individual improvisation through variation resulting in an aesthetic harmony.

The information processing objective is achieved by stochastically creating candidate solutions in a step-wise fashion, where each element is either pulled out randomly from a memory of high-tone results, adjusted from the retention of high-quality solutions or assigned randomly within the bounds of the problem. The memory of candidate solutions is initially random, and a greedy acceptance criterion is employed to admit new candidate solutions only if they sustain an improved objective value, substituting an existing member.

3.3 Learning automata

LA operates by selecting actions via a stochastic process. Such actions operate within an environment while being assessed according to a measure of the system performance. Figure 1a shows the typical learning system architecture. The automaton selects an action (**X**) probabilistically. Such actions are applied to the environment, and the performance evaluation function provides a reinforcement signal β. This is used to update the automaton's internal probability distribution whereby actions that achieve desirable performance are reinforced via an increased probability. Likewise, those underperforming actions are penalized or left unchanged depending on the particular learning rule which has been employed. Over time, the average performance of the system will improve until a given limit is reached. Regarding optimization problems, the action with the highest probability would correspond to the global minimum as demonstrated by rigorous proofs of convergence available in [10, 16].

A wide variety of learning rules has been reported in the literature. One of the most widely used algorithms is the linear reward/penalty (L_{RP}) scheme, which has been shown to guarantee convergence properties (see [16]). With a large number of discrete actions, the probability of selecting any particular action becomes low and the convergence time can become excessive. To avoid this, LA can be connected in a parallel setup like the one shown in Fig. 2b. Each automaton operates a smaller number of actions, and the 'team' works together in a cooperative manner. This scheme can also be used where multiple actions are required.

If action x (parameter) is defined over the range (x_{min}, x_{max}), the probability density function $f(x, n)$ at iteration n is updated according to the following rule:

$$f(x.n+1) = \begin{cases} \alpha[f(x.n) + \beta(n)H(x.r)] & if\ x \in (x_{min}.x_{max}) \\ 0 & otherwise \end{cases} \tag{11}$$

where α is chosen to re-normalize the distribution according to the following condition:

$$\int_{x_{min}}^{max} f(x.n+1)dx = 1 \tag{12}$$

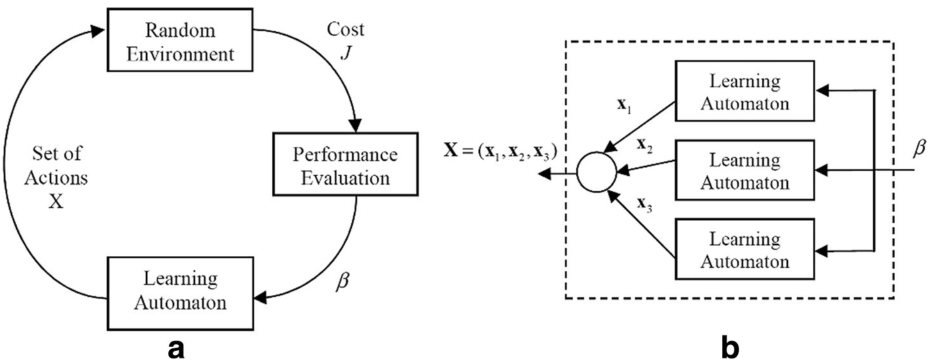


Fig. 2 (**a**) Reinforcement learning system and (**b**) Interconnected automata

where $\beta(n)$ is the reinforcement signal from the performance evaluation and $H(x, r)$ is a symmetric Gaussian neighborhood function centered on $r = x$(n). It yields

$$H(x.r) = \lambda \times \exp\left(-\frac{(x-r)^2}{2\sigma^2}\right) \tag{13}$$

where λ and σ are parameters that determine the height and width of the neighborhood function. They are defined in terms of the range of actions as follows:

$$\sigma = g_w \times (x_{max} - x_{min}), \quad \lambda = \frac{g_h}{(x_{max} - x_{min})} \tag{14}$$

Free parameters thus control the speed and resolution of learning g_w and g_h. Let action $x(n)$ be applied to the environment at iteration n, returning a cost or performance index $J(n)$. Current and previous costs are stored as a reference set $R(n)$. The median and minimum values J_{med} and J_{min} may thus be calculated by means of $\beta(n)$, which is defined as follows:

$$\beta(n) = \max\left\{0.\ \frac{J_{med} - J(n)}{J_{med} - J_{min}}\right\} \tag{15}$$

To avoid problems with infinite storage requirements and to allow the system to adapt to changing environments, only the last m values of the cost functions are stored in $R(n)$. Eq. (15) limits $\beta(n)$ to values between 0 and 1 and only returns nonzero values for those costs that are below the median value. It is easy to understand how $\beta(n)$ affects the learning process as follows: during the learning, the performance and the number of selecting actions can be wildly variable, generating extremely high computing costs. However, $\beta(n)$ is insensitive to such extremes and high values of $J(n)$ resulting from a poor choice of actions. As the learning continues, the automaton converges towards more worthy regions of the parameter space as such actions are chosen to be evaluated more often. When more of such responses are being received, J_{med} gets reduced. Decreasing J_{med} in $\beta(n)$ effectively enables the automaton to refine its reference around better responses (previously received), and hence resulting in a better discrimination between selected actions.

In order to define an action value $x(n)$ which has been associated with a given probability density function, a uniformly distributed pseudo-random number $z(n)$ is generated within the range of [0, 1]. Simple interpolation is thus employed to equate this value to the cumulative distribution function:

$$\int_{x_{min}}^{x(n)} f(x.\ n)dx = z(n) \tag{16}$$

4 Implementation of the Hybrid Method

In the proposed method a pixel is randomly selected to assign it to each member of the hybrid method, from now on BFHS, the population as a position of the pixel. In this study, four different pixel classes are used to segment the images and the idea is to show the effectiveness of the algorithm and its performance against other algorithms solving the same task. However, the implementation can easily be transferred to cases with a greater number of pixel classes.

To approach the histogram of an image by Kapur and Otsu functions, it is necessary to calculate the optimum values of the parameters for each function. This problem can be solved by optimizing Eq. (2) and Eq. (9), the initial values of the relevant parameters are summarized in Table 1.

In the LA optimization, each parameter is considered like an Automaton, which can choose actions. Such actions correspond to values assigned to the parameters by a probability distribution within the interval. For this 2-dimensional problem, one automaton will be created to represent the parametric approach of the corresponding histogram. One of the main advantages of the LA algorithm regarding multi-dimensional problems is that the automatons are coupled only through the environment, which is considered of type P in this study.

As a matter of fact, if each of automata's actions fails, it will be punished, and if each of them is successful, it will get a reward. An action fails when it cannot improve $A(n) = \{s, N_c, N_s, N_{re}, N_{ed}, d_{attract}, w_{attract}, h_{repellent}, w_{repellent}, P_{ed}, \text{HM}, \text{HMCR}, \text{PAR}, \text{BW}, \text{NI}\}$ in an iteration, therefore, it will be punished. If an action can improve $A(n)$, it will get a reward. At the beginning of performing the proposed algorithm, LA probability vector is equal for both actions. BF action is chosen with 50 % probability [0 … 50] and also HS action is selected with 50 % probability (50 … 100]. In fact, given probabilities are both set to 50 %. LA generates a random number with uniform distribution. Regarding the random number and probability vector of LA, one of the actions is selected. If the chosen action is successful, it gets the reward, and its probability value increases according to learning algorithm. However, if it fails, its probability will be decreased accordingly. Therefore, LA learns in different conditions to find out which action is better to be executed more.

The quality of the approach is converted into a reinforcement signal $\beta(n)$ (through Eq. 15). After the reinforcement value $\beta(n)$ is defined as a product of the elected approach $A(n)$, the distribution of probability is updated for $n+1$ of each automaton (according to the Eq. 11). To simplify parameters in Eq. (14), they will take the same

Table 1 Initial parameters of BFHS

Method	Parameter	Value
Bacterial Foraging (BF)	Number of bacterium (s)	20
	Number of chemotactic steps (N_c)	10
	Swimming length (N_s)	10
	Number of reproduction steps (N_{re})	4
	Number of elimination of dispersal events (N_{ed})	2
	Depth of attract ($d_{attract}$)	0.1
	Width of attract ($w_{attract}$)	0.2
	Height of repellent ($h_{repellent}$)	0.1
	Width of repellent ($w_{repellent}$)	10
	Probability of elimination and dispersal (P_{ed})	0.02
Harmony Search (HS)	Harmony memory (HM)	100
	Harmony memory consideration rate (HMCR)	0.75
	Pitch adjusting rate (PAR)	0.5
	Distance bandwidth (BW)	0.3
	Number of improvisations (NI)	250

value for the automaton, such that $g_w = 0.02$ and $g_h = 0.3$. In this work, the optimization process considers a limit up to 2000 iterations.

The final step is to determine the optimal threshold values T_i, just as it is illustrated in Fig. 3.

The optimization algorithm can thus be described as follows (see Table 2):

5 Results and discussions

In this research, two sets of experiments were conducted. In the first set, experiments are done on ten benchmark gray-scale images, Lenna, Pepper, Baboon, Hunter, Map, Cameraman, Living room, House, Airplane, Butterfly (refer to Fig. 4), with size of 512 × 512 and the *uniformity* metric (Eq. 17) [12] is used in order to compare image segmentation performance. Whereas in the second set, experiments are done on a set of satellite images [31], The obtained results are then compared with HS, BF and GA as shown visually in Figs. 7, 8, 9, 10, 11 and 12 and quantitatively in Tables 4, 5, 6, 7, 8 and 9.

5.1 First set of experiments

The hybrid algorithm (BFHS) along with BF, KPSO [22], and GA are used to segment these benchmark images. BF and KPSO parameters are adjusted according to [22]. GA parameters are adjusted on [28]. The region *uniformity* is first stated by Levine and Nazif [12], which is primarily used as a criterion for measuring the quality of image segmentation. The *uniformity* of a feature over a region is defined as being inversely proportional to the variance of the values of that feature evaluated, at every pixel belonging to that region, with an appropriate weighting factor.

$$u = 1-2 \times c \times \frac{\sum_{j=0}^{c} \sum_{i \in R_j} \left(f_i - \mu_j\right)^2}{N \times \left(f_{\max} - f_{\min}\right)^2} \tag{17}$$

where c is the number of thresholds. R_j is the segmented region j. N is the total number of pixels in the given image. f_{i} shows the gray level of pixel i. μ_{i} is the mean gray level of pixels

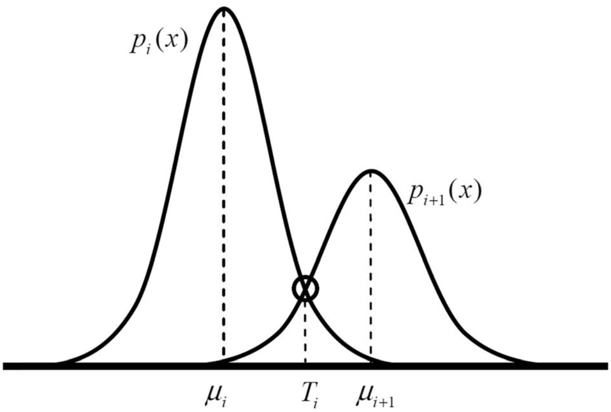


Fig. 3 Thresholding points determination

Table 2 Pseudo-code of the hybrid method

1:	**Initialization** Initialize $A(n) = \{s, N_c, N_s, N_{re}, N_{ed}, d_{attract}, w_{attract}, h_{repellent}, w_{repellent}, P_{ed}$, HM, HMCR, PAR, BW, NI$\}$ to the parameters of **Table 1**, set iteration number $n = 0$, $g_w = 0.02$, $g_h = 0.3$
2:	$A = \arg\min J(n)$
3:	**Repeat**
4:	*LA* select an action *i* based on the probability vector *p*
5:	**If** selected action is *BF* procedure, **then**
6:	**Loop** $l = l + 1$ //Elimination-dispersal loop
7:	**Loop** $k = k + 1$ //Reproduction loop
8:	**Loop** $j = j + 1$ //Chemotaxis loop
9:	**For** i = 1, 2, . . ., *s* takes a chemotactic step for bacterium *i* as follows:
10:	Compute the value fitness function using Eq. (10). Let $P_{best} = g(i, j, k, l)$ to save this value since we may find a better cost via the run.
11:	**Tumble**: Generate a random vector $\Delta(i)$ with each element $\Delta_m(i)$, $m = 1, 2, . . ., P$, a random number on [-1, 1].
12:	**do** MOVE
13:	**do** SWIM
14:	$m = 0$
15:	**While** $m < N_s$
16:	$m = m + 1$.
17:	**If** $g(i, j + 1, k, l) < P_{best}$, **then**
18:	$P_{best} = g(i, j + 1, k, l)$ $other^i(j + 1. k. l) = other^i(j + 1. k. l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
	// where $C(i)$ is the direction of the tumble for bacterium *i*
19:	**Else**, $m = N_s$.
20:	**go** to **10** to process the next bacterium.
21:	**End Loop**
22:	Step 5. Perform reproduction and elimination-dispersal operation.
23:	**End Loop**
24:	**End Loop**
25:	If the maximum number of chemotactic, reproduction and elimination dispersal steps are reached, then go to **26**. Otherwise, go to **6**.
26:	$A(n) = P_{best}$
27:	**elseif** selected action is *HS* **then**
28:	Improvise a new harmony $\mathbf{x}_{new}$ as follows:
29:	**While** $j < n$ **do**
30:	**If** $r_1 <$ HCMR **then**
31:	$x_{new}(j) = x_a(j)$ where $a \in (1, 2, \ldots, \text{HMS})$
32:	**If** $r_2 <$ PAR **then**
33:	$x_{new}(j) = x_a(j) \pm r_3 \times \text{BW}$ // $r_1, r_2, r_3 \in \text{rand}(0, 1)$
34:	**If** $x_{new}(j) < l(j)$ **then**

(Continued)

35:	$x_{new}(j) = l(j)$ // l is the lower bound
36:	**If** $x_{new}(j) > u(j)$
37:	$x_{new}(j) = u(j)$ // u is the upper bound
38:	**else**
39:	$x_{new}(j) = l(j) \pm r \times (u(j) - l(j))$ // $r \in$ rand(0, 1)
40:	**End Loop**
41:	**Update** the HS as $\mathbf{x}_{worst} = \mathbf{x}_{new}$ if $f(\mathbf{x}_{new}) > f(\mathbf{x}_{worst})$
42:	If *NI* is completed or the stop criterion is met, jump to **43**; else go to **28**.
43:	$A(n) = \mathbf{x}_{best}$
44:	Update probability vector p of *LA* using learning rule.
45:	Obtain the minimum, J_{min}, and median, J_{med} of $J(n)$.
46:	Evaluate $\beta(n)$ via Eq. (15).
47:	**Until** $n < 2000$

in the jth region. $f_{\min}$ and $f_{\max}$ are the minimum and maximum gray level of pixels in the given image, respectively. Typically, $u \in [0, 1]$ and a larger amount for u declares that the thresholds are specified with better quality on the histogram.

Figure 5 shows segmented images using the proposed algorithm with 5, 4, and 3 thresholds, respectively. Average *uniformity* obtained from algorithms on these benchmark images with thresholds of 2, 3, 4, and 5 are tabulated in Table 3.

Due to the low ability in local search, GA algorithm leads to the worst results for all cases. Moreover, obtained results from KPSO are not better than BFHS for all cases. The reason is that occasionally KPSO converges toward a local optimum and thus, obtained results are not appropriate. Although BFHS is responsible for exiting from local optimum, it sometimes may not be successful.

Indeed, the inappropriate result of KPSO causes fast convergence of particles to a local optimum. Obtained results from the proposed algorithm are better than other algorithms in all cases. Basically, in the proposed algorithm, LA tries to perform an approach which involves the best result according to the current conditions of the algorithm in the optimization process. Hence, when the problem conditions are such that an algorithm would not be able to improve optimization process, it is used less by LA. As a result, both BF and HS algorithms abilities are utilized in the proposed algorithm. It is observed in the course of experiments that the *uniformity* amount is improved by increasing the number of thresholds for all algorithms.

5.2 Second set of experiments

In this part of experiments, a distinct study on the application of BFHS, BF, HS and GA with two different objective functions (Kapur and Otsu) is made for multilevel thresholding for image segmentation. For the entire test of satellite images that have been considered (see Fig. 6), the BFHS performs as well as or is better than the BF,

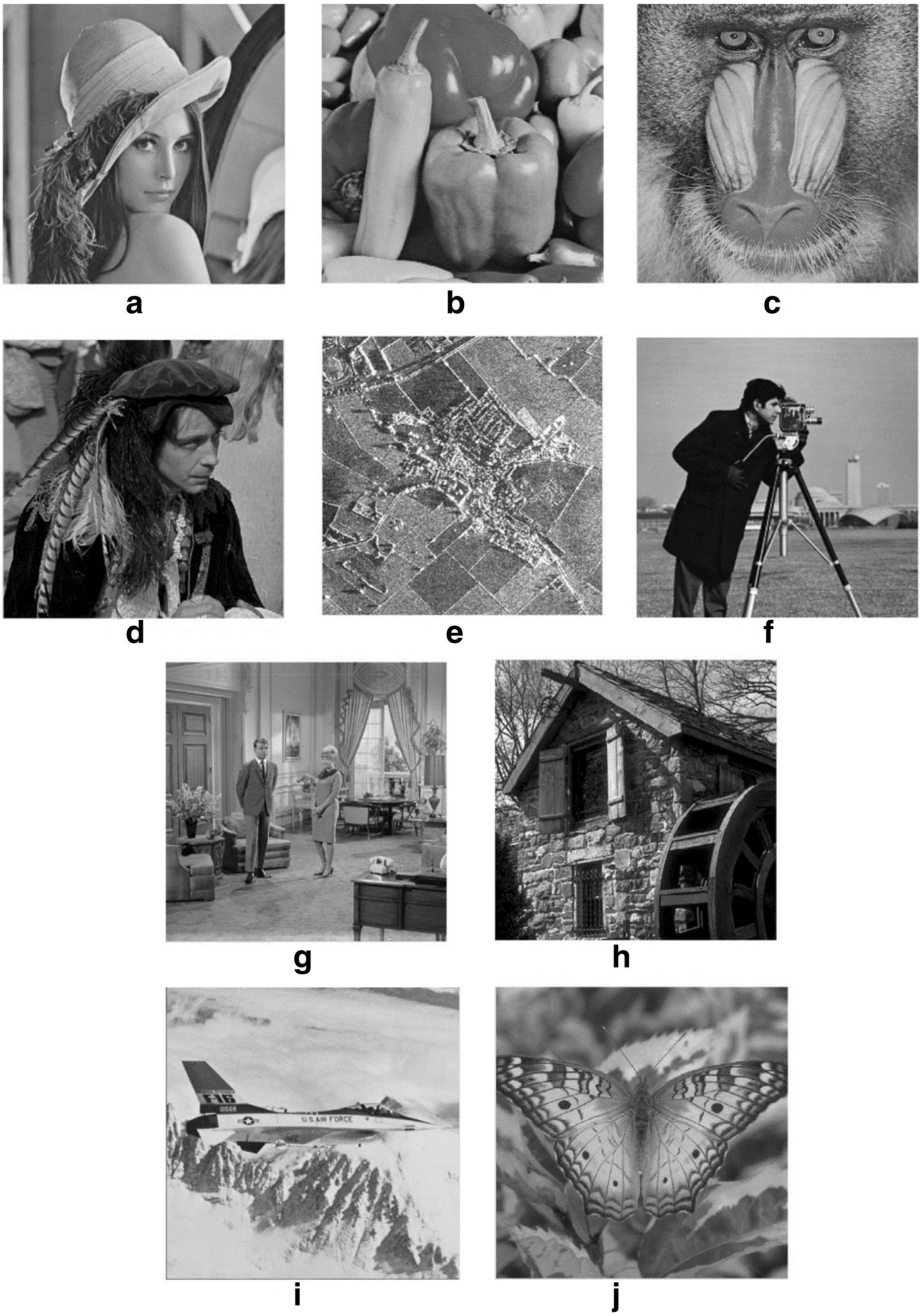


Fig. 4 Standard images which are used for testing the hybrid method. (**a**) Lenna, (**b**) Pepper, (**c**) Baboon, (**d**) Hunter, (**e**) Map, (**f**) Cameraman, (**g**) Living room, (**h**) House, (**i**) Airplane, (**j**) Butterfly

HS, and GA. The experimental results provide evidence for outstanding performance, accuracy and convergence of the proposed algorithm in comparison to other methods.

Fig. 5 Thresholded images obtained by the hybrid method on Kapur thresholding (**a**)–(**j**) represents 3-level thresholding, (a’)–(j’) represents 4-level thresholding, (a”)–(j”) represents 5-level thresholding

On the other hand, it is proved that the computational cost of BFHS is lower than other evolutionary approaches used in the comparison Fig. 7.

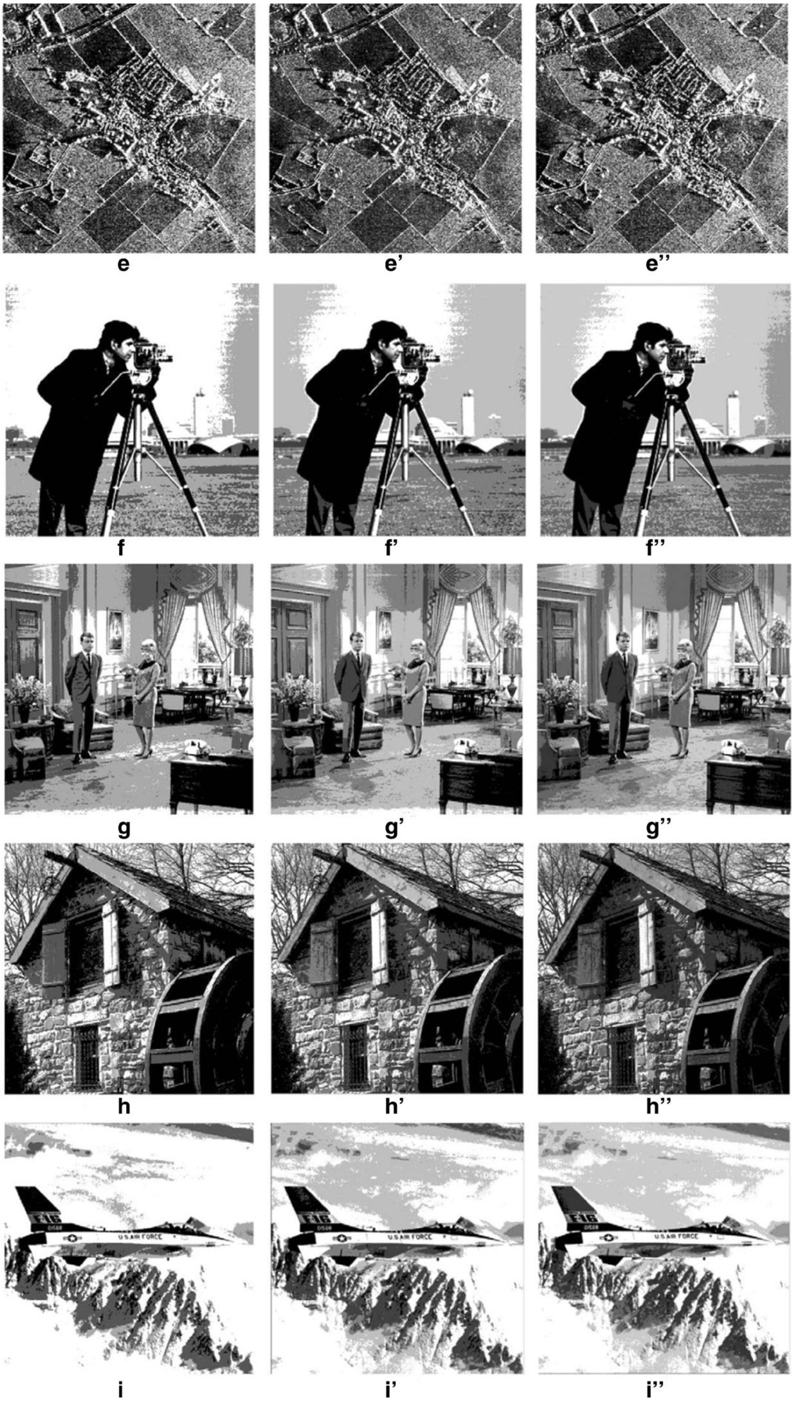


Fig. 5 (continued)

Fig. 5 (continued)

Although, satellite images often need segmentation in the presence of uncertainty, caused due to the factors like highly dependent on environmental conditions, poor resolution, and poor illumination, and have a very low spatial resolution. Owing to the presence of different bands with different wavelength region in the satellite images, the efficiency of the algorithms is affected and containing high resolution is one more cause of inefficiency. As a result, it leads to computational complexity during segmentation.

In satellite images, the rate of information is very high because of that existing features in the image is very dense. Due to that, the rate of change from one region to another region is very rapid. Therefore, in the case of segmentation of remote sensing images or satellite images, accurate segmentation is a very challenging task.

To achieve an accurate and fast segmentation of satellite images, BFHS based robust technique with two most popular objective functions of multilevel thresholding are utilized in this paper, which shows the effectiveness of their segmented results.

While estimating the segmented images, PSNR gives the similarity of an image against a reference image based on the mean square error (MSE) of each pixel which was also reported in [2]. The SSIM is used to compare the structure of original and thresholded image [24]. The SSIM index is calculated from [2]. FSIM [34] is used to calculate the similarity between two images, which can be calculated from [2].

5.2.1 Based on Kapur's entropy

In this section, the results acquired for various satellite images using Kapur's objective function are discussed. Table 4 depicts the number of thresholds, objective values and corresponding optimal threshold values obtained by BFHS, BF, HS and GA methods. It is examined that the objective value evaluated by the proposed method yields highest among all the techniques being compared to different satellite images. PSNR (dB) and MSE values obtained using the proposed BFHS based method are listed in Table 5 and compared with the result acquired using BF, HS, and GA methods respectively. Despite the quality estimation factor using PSNR and MSE, algorithm efficiency (CPU Timing (in seconds)) and feature measurement assessment parameters are also checked using FSIM and SSIM, which is shown in Table 6. It can be clearly observed from Tables 4, 5 and 6 that the proposed BFHS-Kapur's

Table 3 Comparison of uniformity of the proposed algorithm and four comparative approaches

Image	T	GA	BF	KPSO	BFHS
Lenna	2	0.8844	0.9730	0.9728	0.9781
	3	0.9164	0.9781	0.9783	0.9802
	4	0.9198	0.9816	0.9811	0.9833
	5	0.9269	0.9835	0.9834	0.9841
Pepper	2	0.8659	0.9719	0.9720	0.9729
	3	0.8970	0.9773	0.9771	0.9785
	4	0.9054	0.9784	0.9769	0.9793
	5	0.9080	0.9814	0.9825	0.9825
Baboon	2	0.8534	0.9720	0.9731	0.9792
	3	0.8705	0.9759	0.9752	0.9803
	4	0.8812	0.9801	0.9772	0.9829
	5	0.8944	0.9831	0.9838	0.9862
Hunter	2	0.8545	0.9722	0.9732	0.9793
	3	0.8718	0.9765	0.9762	0.9812
	4	0.8812	0.9804	0.9777	0.9832
	5	0.8961	0.9839	0.9838	0.9862
Map	2	0.8546	0.9729	0.9732	0.9795
	3	0.8732	0.9770	0.9764	0.9815
	4	0.8837	0.9806	0.9786	0.9839
	5	0.8996	0.9841	0.9838	0.9868
Cameraman	2	0.8591	0.9736	0.9736	0.9799
	3	0.8751	0.9783	0.9767	0.9816
	4	0.8850	0.9811	0.9794	0.9840
	5	0.8996	0.9842	0.9840	0.9878
Living room	2	0.8592	0.9736	0.9746	0.9800
	3	0.8801	0.9785	0.9767	0.9817
	4	0.8868	0.9816	0.9798	0.9842
	5	89.96	0.9844	0.9851	0.9880
House	2	0.8638	0.9738	0.9747	0.9801
	3	0.8803	0.9796	0.9768	0.9820
	4	0.8896	0.9826	0.9821	0.9842
	5	0.9020	0.9845	0.9855	0.9880
Airplane	2	0.8641	0.9753	0.9750	0.9802
	3	0.8810	0.9800	0.9771	0.9828
	4	0.8924	0.9827	0.9826	0.9846
	5	0.9073	0.9848	0.9866	0.9882
Butterfly	2	0.8641	0.9753	0.9750	0.9802
	3	0.8808	0.9800	0.9772	0.9829
	4	0.9031	0.9828	0.9873	0.9850
	5	0.9124	0.9849	0.9874	0.9885

based technique offers superior objective values in comparison with other evolutionary algorithms such as BF, HS, and GA.

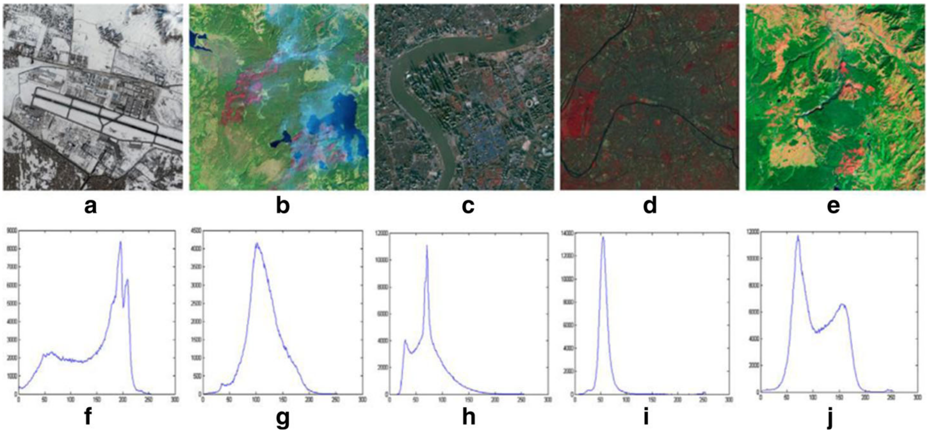


Fig. 6 Five different satellite images are used in the experiments. (**a**–**e**) represent original satellite images, (**f**–**j**) illustrate corresponding histogram image

5.2.2 Based on between-class variance

The performance evaluation of Otsu approach for numerous satellite images is discussed. The number of thresholds, objective values, and corresponding optimal thresholds determined using proposed BFHS technique are listed in Table 7, and results are compared with those obtained using BF, HS, and GA methods respectively. PSNR (dB) and MSE values obtained using the proposed BFHS based method are listed in Table 8 and compared with the results acquired using BF, HS and GA methods, respectively. High PSNR values might be obtained from methods that minimize MSE. Indeed, PSNR gives the similarity of an image against a reference image based on the MSE of each pixel. Apart from quality measurement using PSNR and MSE, algorithm efficiency (CPU time) and feature measurement assessment (FSIM and SSIM) are also checked and presented in Table 9. It can be evidently realized from Tables 7, 8 and 9 that the proposed BFHS-Otsu's based technique provides higher objective values than the other evolutionary algorithms like BF, HS, and GA. Figure 8 shows the segmented images for different threshold levels ($m = 2–5$) obtained for BFHS-Otsu, BF-Otsu, HS-Otsu, and GA-Otsu. Figures 9, 10, 11 and 12 show the segmented images for various threshold levels ($m = 2–5$) obtained for BFHS-Otsu.

5.3 Convergence and computational cost

The convergence curves of BF-Otsu, HS-Otsu, and BFHS-Otsu algorithms have been plotted for the Lenna image for various threshold levels ($m = 2–5$), as shown in Fig. 13. Convergence curves are fairly identical in their plots of HS, BF, and BFHS algorithms. These show that for different thresholds values, BF algorithm converges at the slowest speed and locates local optima only. The HS has fastest convergence speed but fails to locate the global optima. The BFHS not only has fast convergence but also managed to locate the true global optima.

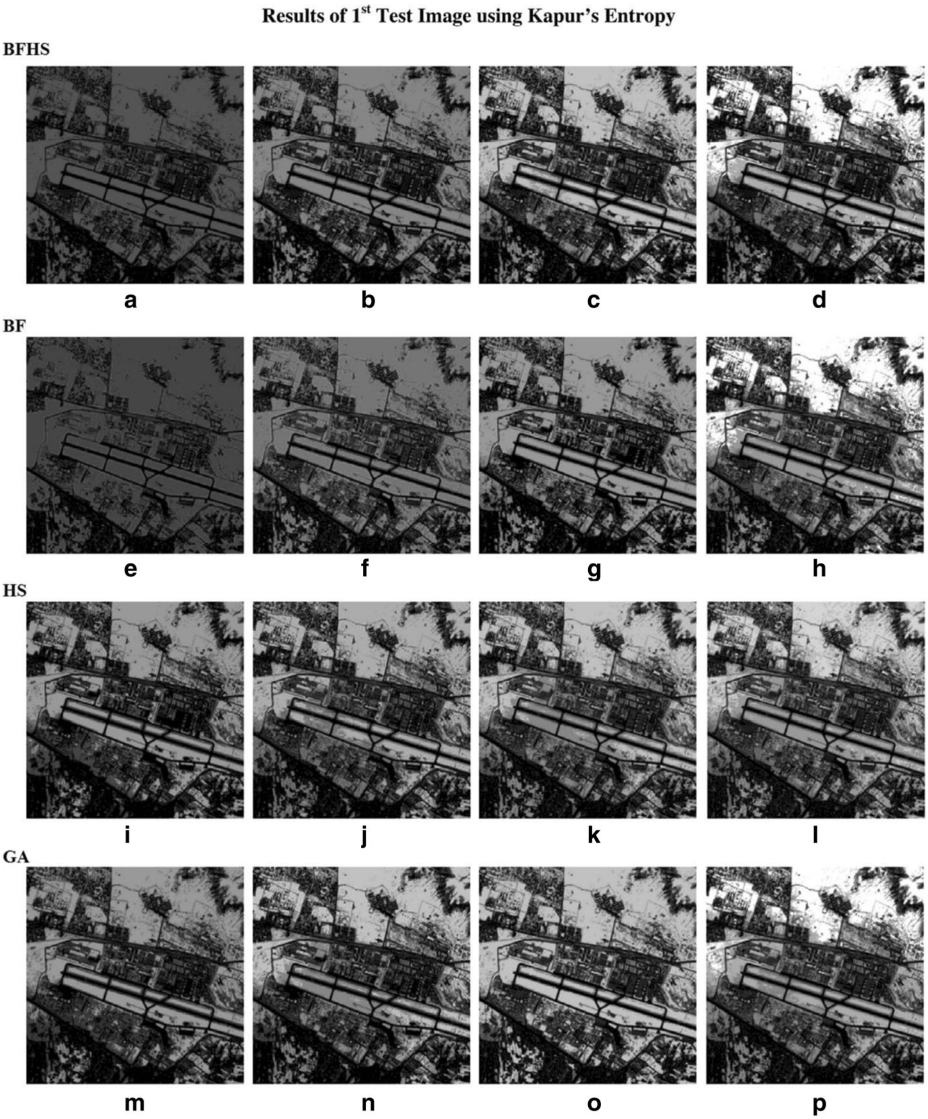


Fig. 7 Results of 1*st* test satellite image using BFHS, BF, HS and genetic algorithm (GA) with Kapur's entropy. (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Kapur's entropy criterion, (**e**–**h**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BF algorithm using Kapur's entropy criterion, (**i**–**l**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from HS algorithm using Kapur's entropy criterion and (**m**–**p**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from GA using Kapur's entropy criterion

Due to the fact that several learning algorithms are sensitive and dependent on the initial value of parameters, it is worth taking this dependency into account. In this experiment, initial values for all methods are initialized in different values while the same histogram is considered for the approximation task. Two sets of randomized values using a uniform distribution were generated to set the initial values of the BF, HS, and BFHS.

Table 4 Comparison of best objective function values and their corresponding threshold values between BFHS, BF, HS, and GA-based technique using Kapur's entropy

Test images	*m*	Best objective function values				Optimum threshold values			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	0.898988	0.898986	0.898972	0.898912	62, 180	65, 183	115, 136	31, 177
	3	1.296296	1.296286	1.296282	1.296280	92, 105, 238	95, 108, 204	138, 193, 250	99, 104, 212
	4	1.654392	1.654390	1.654382	1.654320	35, 121, 170, 199	18, 118, 168, 189	14, 116, 134, 163	55, 107, 168, 231
	5	1.999993	1.999991	1.999984	1.999900	83, 103, 169, 236, 256	68, 156, 157, 225, 255	33, 76, 110, 111, 189	69, 74, 107, 132, 197
2 (512 × 512)	2	0.899171	0.898077	0.897812	0.892253	68, 204	66, 199	77, 212	66, 99
	3	1.299174	1.297668	1.295387	1.287382	52, 138, 212	67, 130, 198	146, 175, 240	12, 113, 179,
	4	1.659995	1.659987	1.655493	1.656630	46, 104, 161, 203	30, 105, 162, 189	75, 115, 164, 199	51, 78, 150, 179
	5	1.999997	1.999955	1.999982	1.999985	56, 89, 153, 195, 226	36, 90, 139, 186, 219	12, 24, 96, 138, 140	69, 74, 107, 132, 197
3 (616 × 616)	2	0.888930	0.888239	0.885786	0.882399	61, 159	70, 130	14, 137	80, 108
	3	1.296299	1.296290	1.296192	1.294283	69, 180, 201	78, 177, 215	8, 167, 255	89, 185, 216
	4	1.659990	1.659991	1.654419	1.654222	56, 66, 109, 197	50, 63, 111, 204	18, 135, 189, 204	52, 65, 94, 178
	5	1.999991	1.999990	1.999979	1.999919	65, 66, 127, 145, 184	70, 129, 192, 241, 255	6, 36, 99, 146, 186	08, 35, 154, 227, 240
4 (512 × 512)	2	0.888892	0.888862	0.888465	0.888206	66, 178	62, 178	49, 174	74, 125
	3	1.299488	1.299077	1.297182	1.297481	48, 138, 202	52, 142, 199	62, 122, 197	34, 202, 226
	4	1.659922	1.650984	1.655421	1.655544	75, 144, 180, 211	72, 140, 162, 199	136, 146, 161, 222	161, 164, 183, 186
	5	1.999995	1.999994	1.999994	1.998369	13, 64, 139, 160, 182	108, 171, 229, 254, 255	13, 22, 86, 170, 208	26, 117, 188, 223, 231
5 (900 × 900)	2	0.888889	0.888886	0.888889	0.888806	69, 178	69, 178	129, 226	69, 176
	3	1.299346	1.999922	1.296335	1.292477	52, 82, 199	52, 80, 190	7, 55, 166	51, 100, 225
	4	1.659233	1.659055	1.656620	1.659716	64, 99, 104, 189	62, 92, 105, 195	63, 92, 98, 175	114, 196, 221, 246
	5	1.999999	1.999996	1.999996	1.999897	10, 33, 121, 196, 287	96, 151, 221, 222, 255	60, 60, 98, 157, 199	59, 80, 87, 146, 231

Table 5 Comparison of PSNR (dB) and MSE values between BFHS, BF, HS and GA based technique using Kapur's entropy

Test images	*m*	PSNR (dB)				MSE			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	24.640759	24.659402	24.649714	24.691119	2.233595	2.224027	2.228993	2.207844
	3	24.838077	24.867909	24.840048	24.904497	2.134384	2.119773	2.133415	2.101989
	4	25.065555	25.039729	25.063891	25.978840	2.025465	2.037546	2.026241	1.641331
	5	26.219373	26.133128	25.241718	25.859903	1.426084	1.584044	1.944950	1.686902
2 (512 × 512)	2	24.545122	24.542932	24.551459	24.582743	2.283327	2.284478	2.279997	2.263632
	3	24.765263	24.792172	24.747597	24.756188	2.170471	2.157064	2.179318	2.175011
	4	24.976139	25.012000	24.969124	25.897146	1.006857	2.050597	2.070941	1.672498
	5	25.826587	25.760391	25.188178	25.676672	1.240895	1.726001	1.969076	1.759596
3 (616 × 616)	2	24.626587	24.606034	24.575999	24.591409	2.240895	2.251525	2.26715	2.259120
	3	24.801272	24.801903	24.840962	24.811471	2.152549	2.152236	2.132967	2.147500
	4	25.024089	25.008767	25.008374	25.979138	2.044896	2.033309	2.052309	1.641218
	5	25.833679	25.542806	25.238145	26.106857	1.697119	1.814678	1.946551	1.593656
4 (512 × 512)	2	24.537128	24.569481	24.701880	24.697130	2.287533	2.270555	2.202380	2.204790
	3	24.808983	24.784945	24.790649	24.903379	2.14873	2.160657	2.15782	2.102530
	4	25.101653	25.052971	25.008708	25.051311	2.008699	2.031342	2.052152	1.690243
	5	25.916055	25.627852	25.234411	25.853588	1.665232	1.779488	1.948225	1.650902
5 (900 × 900)	2	24.320651	24.513275	24.467392	24.484757	2.404447	2.300132	2.324561	2.315285
	3	24.659718	24.630789	24.703762	24.709665	2.223865	2.238728	2.201425	2.198436
	4	24.860823	24.847207	24.954384	25.893057	2.123234	2.129901	2.077982	1.674073
	5	26.066495	25.957263	25.206009	26.150378	1.028536	1.076605	1.961008	1.577765

Table 6 Comparison of CPU Timing, SSIM and FSIM between BFHS, BF, HS and GA based technique using Kapur's entropy

Test images	m	CPU timing				SSIM				FSIM			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	2.81911	8.235545	25.22988	436.11369	0.918842	0.902112	0.943819	0.947394	0.772895	0.722617	0.895365	0.897415
	3	3.286868	8.075897	24.80887	433.87366	0.961164	0.959592	0.968163	0.963107	0.884754	0.882816	0.937511	0.928888
	4	2.612452	6.917417	24.86667	438.91758	0.978343	0.961763	0.979184	0.976280	0.947416	0.900145	0.955266	0.909506
	5	2.821316	8.247721	25.73357	296.15088	0.989280	0.977779	0.985351	0.981165	0.964843	0.939080	0.964817	0.955579
2 (512 × 512)	2	2.843121	6.890611	21.00694	185.94096	0.900747	0.865230	0.936475	0.941127	0.669172	0.579772	0.894690	0.891273
	3	2.48574	6.641995	22.24080	181.80301	0.966901	0.953207	0.965857	0.962966	0.914892	0.806251	0.937512	0.936411
	4	2.528806	6.336246	21.84823	184.04964	0.97696	0.970332	0.977355	0.969124	0.935541	0.906891	0.955900	0.897320
	5	3.181302	6.661542	23.07188	121.46275	0.989512	0.981282	0.984123	0.979080	0.967901	0.942435	0.966675	0.944366
3 (616 × 616)	2	2.947659	6.510813	24.00778	268.01849	0.886746	0.877771	0.945456	0.945627	0.711995	0.690732	0.910538	0.909899
	3	2.487444	6.295622	23.27194	266.77671	0.958891	0.949243	0.967704	0.965602	0.884364	0.861097	0.946963	0.950250
	4	2.621718	7.009322	23.85787	268.79580	0.976986	0.974446	0.979841	0.975580	0.947437	0.927941	0.967487	0.910598
	5	2.733291	6.570533	24.95336	183.32426	0.985457	0.978247	0.985389	0.982706	0.978693	0.957271	0.977198	0.959924
4 (512 × 512)	2	2.881887	6.883393	21.67048	184.39762	0.919911	0.916411	0.942495	0.941113	0.808156	0.758135	0.924892	0.924158
	3	2.901766	6.518426	22.29530	180.80266	0.965901	0.939357	0.967192	0.958473	0.920468	0.809099	0.962968	0.969056
	4	2.583831	6.224254	23.29745	185.24697	0.978174	0.977754	0.978779	0.973418	0.958493	0.954723	0.975200	0.946636
	5	2.670841	6.753141	22.89839	123.92201	0.985296	0.979699	0.984767	0.981415	0.984812	0.965946	0.982297	0.963563
5 (900 × 900)	2	2.706739	3.260656	25.04838	609.70270	0.888275	0.869001	0.912923	0.931914	0.763680	0.61448	0.935430	0.910566
	3	3.18756	3.141117	26.31972	611.34520	0.947652	0.924667	0.952287	0.961039	0.854385	0.841511	0.962090	0.936919
	4	3.782609	3.073785	26.74656	594.61794	0.96061	0.948236	0.972788	0.972305	0.941277	0.805258	0.975354	0.922978
	5	3.317179	3.118346	26.48113	400.37624	0.982745	0.969831	0.981889	0.977216	0.989692	0.939845	0.982301	0.949224

Table 7 Comparison of best objective function values and their corresponding threshold values between BFHS, BF, HS and GA based technique using Otsu

Test images	m	Best objective function values				Optimum threshold values			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	0.9324672	0.9217461	0.9018988	0.8962822	92, 165	90, 165	90, 164	89, 159
	3	0.9557924	0.9429610	0.9416061	0.9160700	50, 130, 203	50, 118, 201	50, 116, 181	50, 136, 201
	4	0.9685061	0.9510638	0.9636437	0.9558676	60, 90, 164, 195	62, 90, 160, 185	29, 87, 134, 194	69, 108, 184, 184
	5	0.9953252	0.9897911	0.9835491	0.9738400	32, 69, 123, 174, 224	61, 118, 197, 239, 254	59, 91, 133, 179, 217	41, 128, 215, 221, 247
2 (512 × 512)	2	0.9379702	0.9375483	0.9235074	0.9205700	92, 176	90, 170	92, 173	88, 170
	3	0.9510369	0.9503637	0.9480333	0.9419880	65, 148, 201	69, 130, 201	62, 138,204	70, 120, 185
	4	0.9780577	0.9668738	0.9661590	0.9552768	52, 99, 132, 190	58, 92, 132, 188	56, 95, 130, 179	109, 162, 217, 242
	5	0.998046	0.9966540	0.9947617	0.9717720	35, 81, 127, 175, 217	58, 106, 141, 173, 227	49, 85, 132, 171, 213	41, 88, 125, 209, 249
3 (616 × 616)	2	0.9205902	0.9053900	0.9231815	0.9155424	90, 177	92, 164	91, 168	77, 164
	3	0.9550135	0.9514701	0.9496108	0.944740	65, 142, 195	62, 138, 188	65, 133, 193	65, 141, 184
	4	0.9737103	0.9687232	0.9670056	0.9587931	60, 90, 169, 214	52, 96, 160, 203	45, 94, 159, 200	88, 161, 208, 245
	5	0.989655	0.9818421	0.9805966	0.9834300	24, 70, 117, 169, 222	40, 86, 146, 200, 238	66, 103, 148, 171, 218	23, 50, 162, 195, 248
4 (512 × 512)	2	0.9218636	0.9172908	0.9087261	0.9038803	77, 190	72, 188	73, 171	104, 174
	3	0.9489847	0.9307470	0.9318175	0.9253098	60, 118, 189	68, 152, 177	62, 108, 183	88, 158, 199
	4	0.9748262	0.9480545	0.9557225	0.9517529	52, 98, 149, 204	36, 98, 147, 210	51, 97, 147, 210	34, 98, 171, 171
	5	0.9974924	0.9881785	0.9768757	0.9660907	47, 95, 145, 187, 227	68, 104, 155, 211, 254	61, 94, 128, 180, 199	27, 62, 151, 204, 245
5 (900 × 900)	2	0.9128477	0.9108475	0.9092101	0.8993944	80, 170	82, 172	87, 176	97, 172
	3	0.955479	0.9376586	0.9256240	0.9106151	88, 138, 204	86, 135, 204	82, 134, 204	72, 122, 201
	4	0.9618102	0.9570892	0.9439054	0.9542124	64, 123, 168, 214	62, 113, 160,214	64, 119, 160, 212	128, 207, 221
	5	0.9979752	0.9849869	0.9803246	0.9843358	39, 78, 120, 166, 210	30, 59, 113, 164, 215	50, 79, 134, 155, 185	48, 123, 162, 207, 245

Table 8 Comparison of PSNR (dB) and MSE values between BFHS, BF, HS, and GA-based technique using Otsu

Test images	m	PSNR (dB)				MSE			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	24.653841	24.648178	24.644845	24.645697	2.226877	2.229782	2.2314943	2.231056
	3	24.856153	24.870026	24.859707	24.870600	2.125519	2.118740	2.1237805	211.8451
	4	25.067745	25.054861	25.066102	26.279928	2.024444	2.030459	2.0252100	1.531396
	5	25.769200	25.082472	25.293798	25.124000	1.722504	2.017590	1.9217658	199.8379
2 (512 × 512)	2	24.56352	24.579735	24.546290	24.552551	2.273674	2.265201	2.2827130	2.279424
	3	24.780481	24.759637	24.732067	24.764200	2.162878	2.173284	2.1871251	217.1000
	4	24.981269	25.006653	25.012021	24.874078	2.065158	2.053123	2.0505869	2.116764
	5	25.627989	25.4729200	25.197580	25.004700	1.779431	1.844116	1.9648177	205.4032
3 (616 × 616)	2	24.621500	24.583300	24.585789	24.570418	2.24352	2.263334	2.2620458	2.270065
	3	24.800300	24.794900	24.836735	24.786600	2.153032	2.155715	2.1350436	215.9814
	4	24.968400	25.004600	24.973612	24.974895	2.071303	2.054118	2.0688030	2.068191
	5	25.771400	25.300500	25.234673	25.158200	1.721626	1.918794	1.9481078	198.2723
4 (512 × 512)	2	24.652600	24.596500	24.683958	24.612642	2.227535	2.256463	2.2114878	2.248102
	3	24.890500	24.800200	24.826838	24.828800	2.108757	2.153061	2.1399148	213.8947
	4	25.100800	25.014100	24.998579	26.075901	2.009097	2.049629	2.0569436	1.605056
	5	25.397300	25.136500	25.323115	25.007600	1.876502	1.992666	1.9088367	205.2699
5 (900 × 900)	2	24.528200	24.518400	24.477724	24.461037	2.292235	2.297417	2.3190380	2.327965
	3	24.70400	24.705900	24.712801	24.725000	2.201311	2.200361	2.1968492	219.0674
	4	24.935600	24.953200	24.948091	24.958259	2.087012	2.078549	2.0809957	2.076129
	5	25.792700	25.680600	25.170055	25.013500	1.713199	1.757999	1.9773103	204.9903

Table 9 Comparison of CPU Timing, SSIM, and FSIM between BFHS, BF, HS and GA based technique using Otsu

Test images	m	CPU timing				SSIM				FSIM			
		BFHS	BF	HS	GA	BFHS	BF	HS	GA	BFHS	BF	HS	GA
1 (750 × 750)	2	4.667568	5.007778	7.29691	272.89039	0.920762	0.904131	0.9423958	0.942725	0.772803	0.728028	0.8925695	0.889432
	3	5.363554	5.276051	6.587852	275.03171	0.963812	0.968801	0.9687778	0.962500	0.917134	0.929694	0.9360198	0.924800
	4	5.076405	5.519536	7.154403	272.78504	0.971162	0.972597	0.9778467	0.955294	0.936363	0.938145	0.9487690	0.870566
	5	4.784902	5.427858	6.753919	275.44405	0.982051	0.967690	0.9819275	0.95570000	0.961703	0.922072	0.9597429	0.894800
2 (512 × 512)	2	2.251826	2.511814	13.23507	121.92735	0.907199	0.898708	0.9341685	0.936426	0.690586	0.673149	0.8938984	0.892676
	3	2.383504	2.495365	4.30062	118.78825	0.958494	0.950182	0.9613643	0.964400	0.834441	0.803622	0.9362953	0.928500
	4	2.243267	2.596011	4.000213	119.73264	0.97805	0.976883	0.9771952	0.924582	0.929875	0.912289	0.9267295	0.892821
	5	2.228959	2.434094	4.336584	119.81306	0.984679	0.978082	0.9837941	0.972000	0.968999	0.942692	0.9662359	0.929000
3 (616 × 616)	2	3.244759	3.504219	5.748055	174.33693	0.885600	0.869500	0.9450091	0.945003	0.704500	0.668900	0.9009851	0.903914
	3	3.437252	3.588553	5.722739	174.18969	0.958300	0.957800	0.9666987	0.962600	0.881700	0.885600	0.9464230	0.929000
	4	3.440058	3.744837	5.670794	174.57630	0.978500	0.978500	0.9784969	0.960120	0.971100	0.967400	0.9639295	0.939038
	5	3.406708	3.732528	5.793264	173.69577	0.981600	0.979800	0.9806148	0.947300	0.968800	0.966400	0.9680752	0.929800
4 (512 × 512)	2	2.122679	2.358085	4.154642	120.80439	0.922700	0.917100	0.9384357	0.920310	0.767700	0.750400	0.9305519	0.910021
	3	2.194987	2.365144	4.415085	117.20405	0.965300	0.96400	0.9668690	0.947000	0.899300	0.914800	0.9562953	0.9399
	4	2.345375	2.634723	4.647286	122.83085	0.978700	0.977700	0.9779908	0.944344	0.958200	0.975200	0.9782707	0.915081
	5	2.366824	2.664677	4.727931	118.01789	0.982700	0.975600	0.9817766	0.961100	0.972300	0.969900	0.9680768	0.951300
5 (900 × 900)	2	8.237369	8.095456	8.227432	401.23638	0.894200	0.905700	0.9361244	0.931305	0.681000	0.710500	0.9231666	0.905986
	3	7.747557	7.904731	8.647928	398.72867	0.960500	0.958400	0.9527245	0.956800	0.872100	0.862200	0.9445500	0.949800
	4	7.998608	8.594646	8.813292	403.10783	0.979300	0.978100	0.9712128	0.890741	0.955700	0.966800	0.9695881	0.838519
	5	8.233814	8.289011	8.960079	404.39053	0.981400	0.977300	0.9801409	0.971100	0.956100	0.952200	0.9506624	0.952100

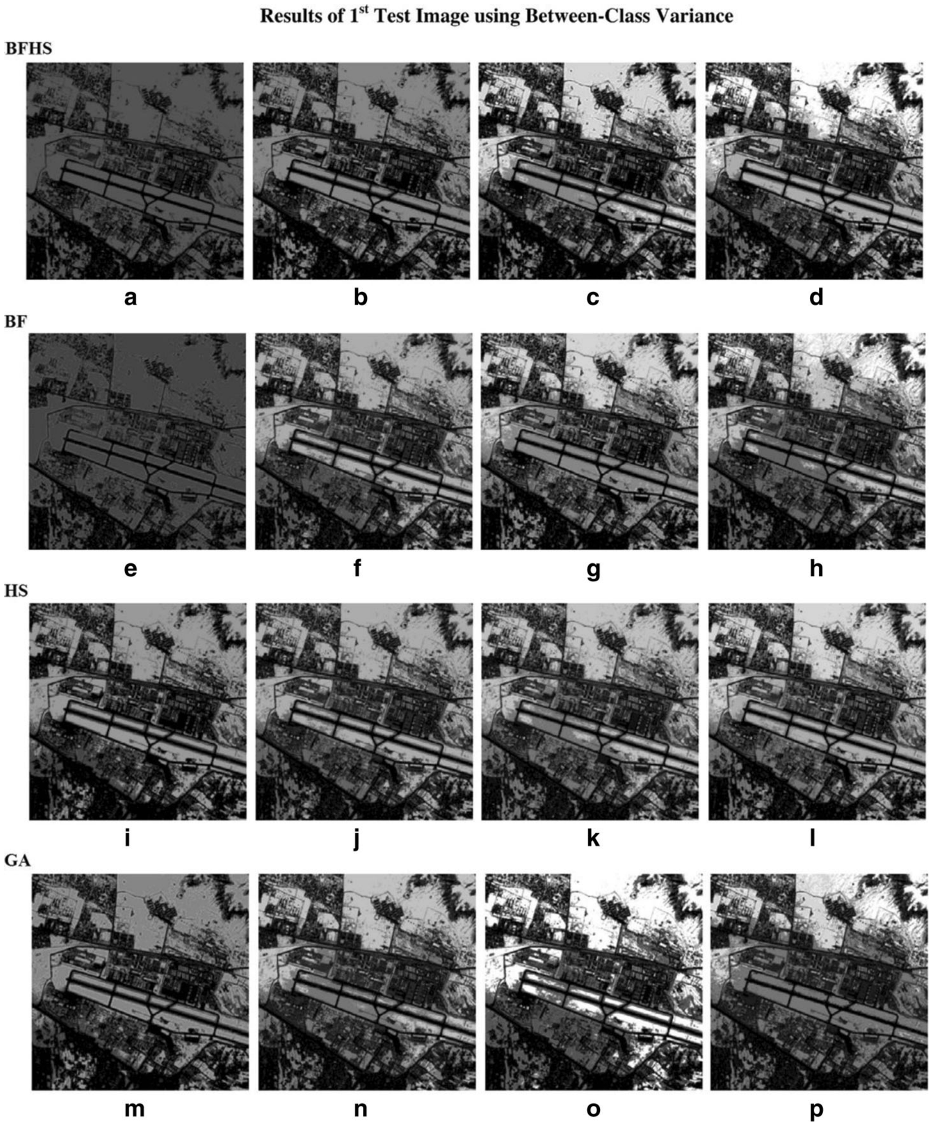


Fig. 8 Results of 1*st* test satellite image using BFHS, BF, HS and GA with Otsu entropy. (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Otsu's entropy criterion, (**e**–**h**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BF algorithm using Otsu's entropy criterion, (**i**–**l**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from HS algorithm using Otsu's entropy criterion and (**m**–**p**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from GA using Otsu's entropy criterion

In the BFHS case, the LA algorithm does not require initialization as it works with random initial values; however, in order to assure a valid comparison, the same initial values are considered for the BF, the HS, and the BFHS methods. Figure 14 shows a clear pixel misclassification in some sections of the image as a consequence of such sensitivity.

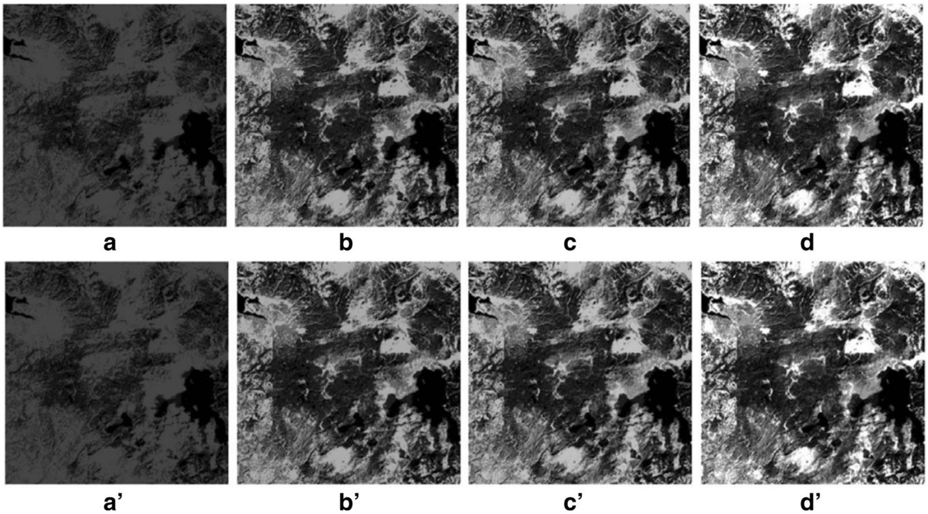


Fig. 9 Results of *2nd* test satellite image using BFHS (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Kapur's entropy criterion, (a'–d') 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Otsu's entropy criterion

The experiment aims to measure the number of required steps and the computing time spent by the Expectation-Maximization (EM) [32], the (Levenberg-Marquardt) LM [30] and the LA algorithm needed to calculate the parameters of the Kapur and Otsu in satellite images. All experiments consider four threshold levels ($m = 2$–5). Table 10 shows the averaged measurements as they are obtained from 20 experiments.

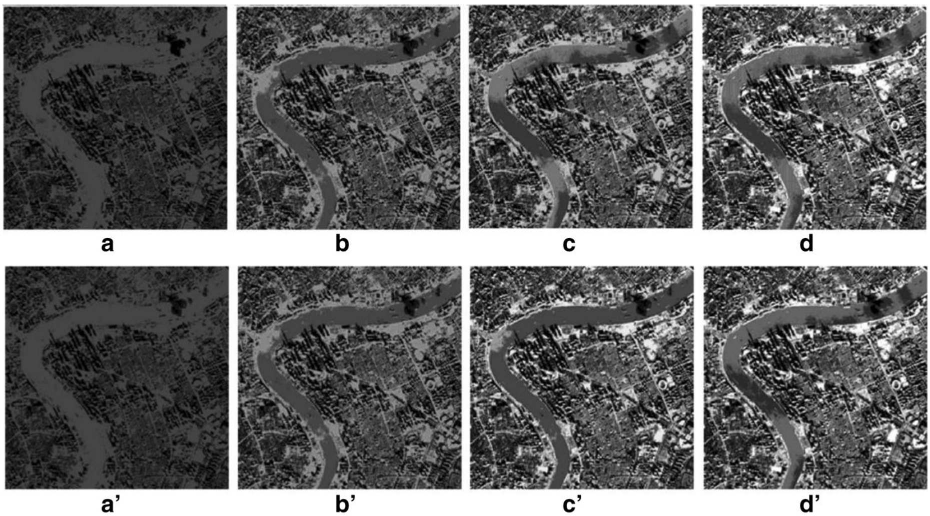


Fig. 10 Results of *3rd* test satellite image using BFHS (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Kapur's entropy criterion, (a'–d') 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Otsu's entropy criterion

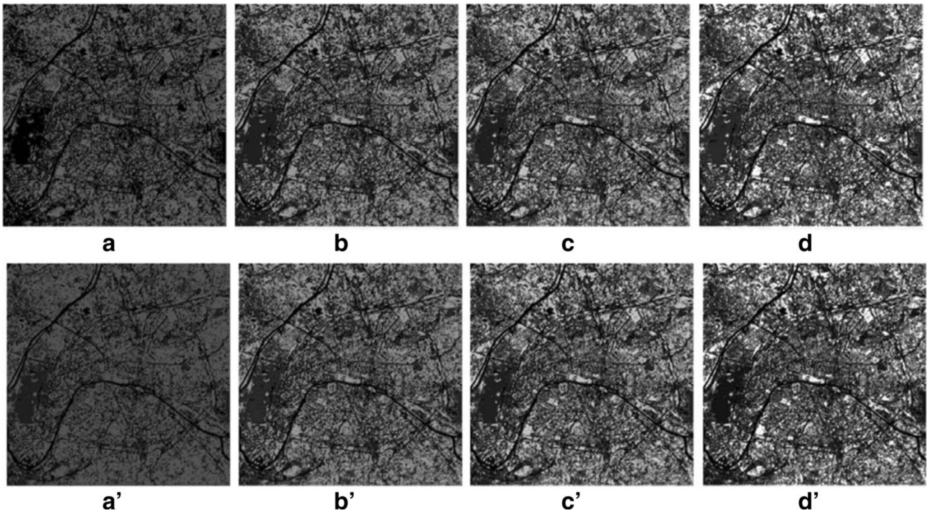


Fig. 11 Results of 4*th* test satellite image using BFHS (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Kapur's entropy criterion, (a'–d') 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Otsu's entropy criterion

It is evident that the EM is the slowest to converge (iterations), and the LM shows the highest computational cost (elapsed time) because it requires complex Hessian approximations. On the other hand, the LA shows an acceptable compromise between its convergence time and its computational cost.

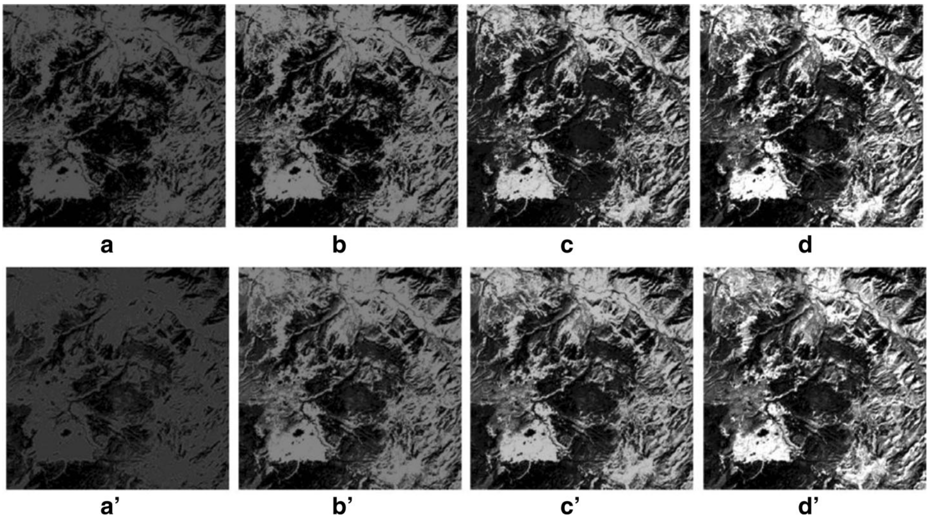
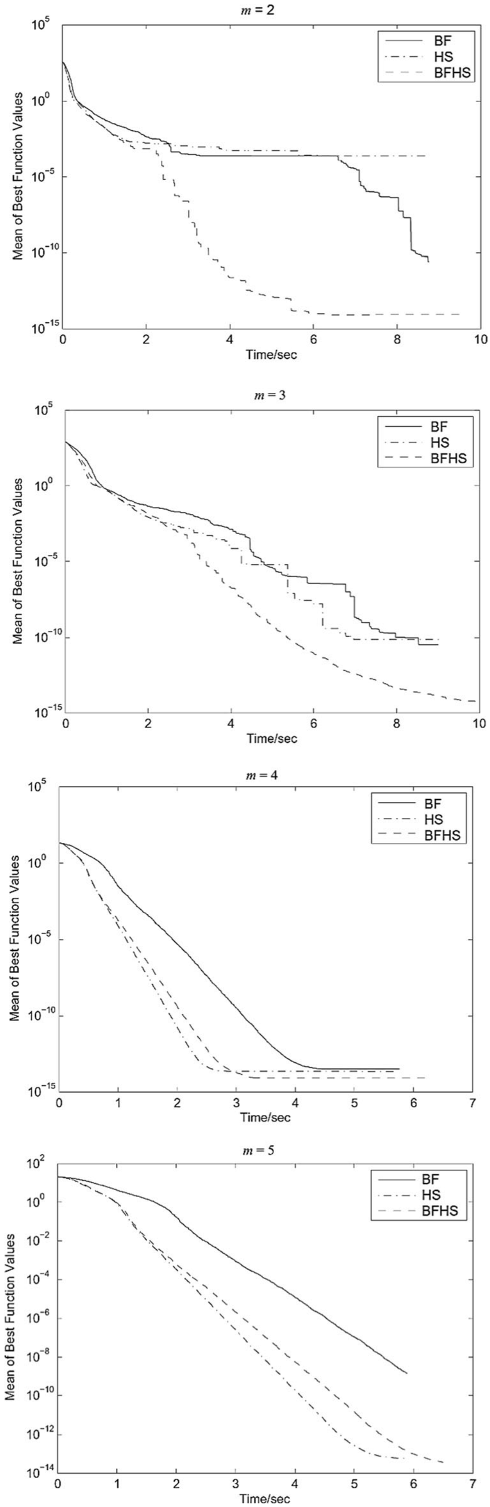


Fig. 12 Results of 5*th* test satellite image using BFHS (**a**–**d**) 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Kapur's entropy criterion, (a'–d') 2-level to 5-level thresholding based segmented image with the best thresholds obtained from BFHS algorithm using Otsu's entropy criterion

Fig. 13 Convergence curves of BF, HS, and BFHS algorithms for different threshold levels



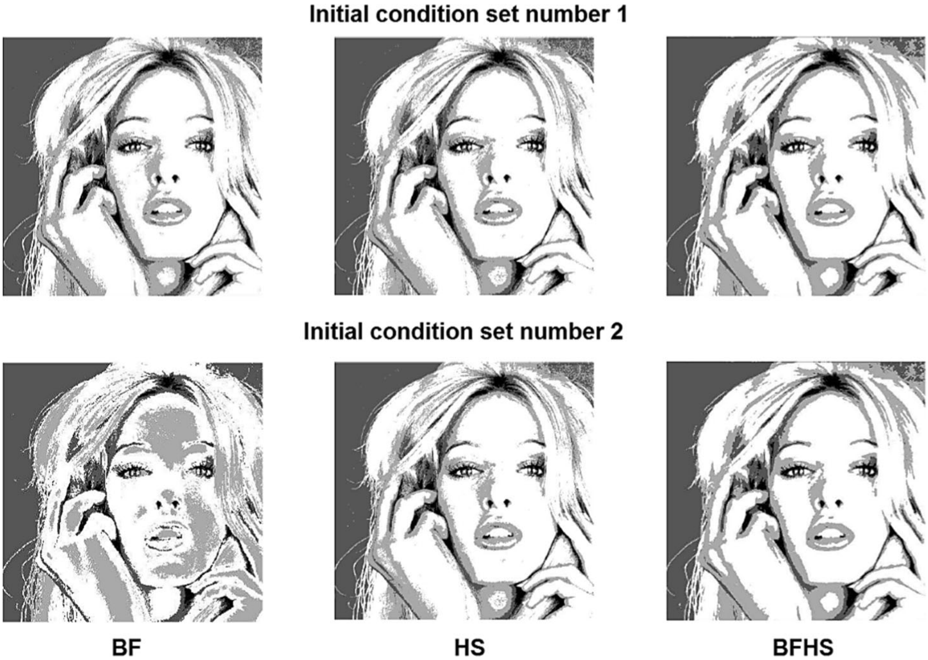


Fig. 14 Segmented images after applying the BF, the HS, and the BFHS algorithms with different initial conditions

5.4 Time complexity

The time complexity of BFHS is based on the complex nature of its inline processes, namely, the LA-based switching, calculating the quality of the segment, applying BF, and applying HS. Assuming T_1 is the number of iterations taken by the BF to converge, and T_2 is the number of iterations taken by the HS to converge. Then the complexity of LA-based switching is $O(s\ T_1\ T_2\ N_c\ N_p\ N_d)$, while the complexity of calculating the quality of a partition will depend on the time complexity of the validity index which is some constant, q, multiplied by N_p for all the indices used in this paper. Therefore, the complexity of this step will be $O(q\ T_1\ T_2\ N_p)$. The parameters T_1, T_2, N_c, s, and ξ can be fixed in advance. Typically, $T_1, T_2, N_c, s, \xi, N_d \ll N_p$. Let α be the multiplication of T_1, T_2, N_c, s, N_d ($\alpha = s \times T_1 \times T_2 \times N_c \times N_d$). If $\alpha \le N_p$ then the time complexity of BFHS will be $O(N_p)$. However, if $\alpha \approx N_p$, then the time complexity of BFHS will be $O(N_p^2)$.

Table 10 Average iterations and time requirements of the EM, the LM, and the LA algorithm

Iterations Elapsed time	Image 1	Image 2	Image 3	Image 4	Image 5
EM	1855	1833	1861	1870	1925
	7.06 s	4.42 s	5.58 s	5.71 s	10.23 s
LM	985	988	945	958	1028
	10.48 s	6.62 s	10.19 s	10.43 s	16.96 s
LA	970	991	951	951	1009
	3.92 s	2.51 s	3.03 s	3.1 s	5.65 s

6 Conclusion

In this paper, an automatic image multi-threshold approach based on the combination of Bacterial Foraging (BF) optimization algorithm, Harmony Search (HS) algorithm and Learning Automata (LA) is proposed. In this approach, the initial population is produced by combining chaotic systems with opposition-based learning routine to improve global convergence rate. In this paper, BFHS, BF, HS and GA algorithms were exploited to maximize Kapur's entropy and between-class variance (Otsu) separately to find optimum multilevel thresholds. The results suggest that the hybrid algorithm (BFHS) with Kapur's and Otsu's entropy criterion can be efficiently used in multilevel thresholding segmentation for two set of images, namely standard and satellite image.

The numerical illustrations and fidelity assessments for almost every sample images, considered in this paper, demonstrate that the BFHS-Kapur's and BFHS-Otsu's outperform other competitive algorithms with each objective functions. The best part of BFHS is its computational efficiency, the accuracy of segmentation. Moreover, experimental evidence shows that LA algorithm has an acceptable compromise between its convergence time and its computational cost when it is compared to the Expectation-Maximization (EM) method and the Levenberg-Marquardt (LM) algorithm. The results have shown that the stochastic search accomplished by the LA method shows a consistent performance with no regard to the initial value and still indicating a greater chance to reach the global minimum.

The study also explores the comparison between the two objective functions with each optimization technique, which reveals that Kapur's entropy is superior for each sample images. The advantage of Kapur's entropy is that it uses a global and objective property of the histogram; because of its general nature, this criterion can be used for segmentation purpose. The validity and accuracy of the proposed BFHS based multilevel thresholding technique are reported both qualitatively and quantitatively. To measure the performance of proposed approach, uniformity, best objective value, MSE, SD, FSIM, SSIM, and PSNR, which assesses the segmentation quality, considering the coincidences between the segmented and the original images, has been used.

It is concluded that the order of CPU runtime from low to high are sorted in the order of BFHS < HS < BF < GA. This is due to the fast convergence rate of modified search initialization that the input parameters such as the number of iteration, the number of population size used are lesser than BF, HS, and GA.

The experimental results are very promising and encourage future research for applying BFHS to complex image processing application such as satellite image enhancement, satellite image denoising, and optimization based image classification and also in various computer vision problems. The performance of few more objective functions such as minimum cross entropy and Renyi's entropy can also be estimated using BFHS techniques for a standard set of images as well as for satellite images to check the robustness of this algorithm for multilevel thresholding problem.

References

1. Akay B (2013) A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding. Appl Soft Comput 13:3066–3091
2. Bhandari AK, Singh VK, Kumar A, Singh GK (2014) Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. Expert SysT Appl 41:3538–3560
3. Brownlee J (2011) Clever algorithms: nature-inspired programming recipes, Jason Brownlee

4. Chen J, Pappas TN, Mojsilović A, Rogowitz BE (2005) Adaptive perceptual color-texture image segmentation. Imag Process, IEEE Trans 14:1524–1536
5. Cuevas E, Sossa H (2013) A comparison of nature inspired algorithms for multi-threshold image segmentation. Expert Syst Appl 40:1213–1219
6. Fan J, Yau DK, Elmagarmid AK, Aref WG (2001) Automatic image segmentation by integrating color-edge extraction and seeded region growing. Imag Process, IEEE Trans 10:1454–1466
7. Hammouche K, Diaf M, Siarry P (2008) A multilevel automatic thresholding method based on a genetic algorithm for a fast image segmentation. Comput Vis Image Underst 109:163–175
8. Hill PR, Canagarajah CN, Bull DR (2003) Image segmentation using a texture gradient based watershed transform. Imag Process, IEEE Trans 12:1618–1633
9. Horng M-H (2010) Multilevel minimum cross entropy threshold selection based on the honey bee mating optimization. Expert Syst Appl 37:4580–4592
10. Kaddour N, Alexander PS (1994) Learning automata: theory and applications. Pergamon Press, Inc
11. Kapur JN, Sahoo PK, Wong AK (1985) A new method for gray-level picture thresholding using the entropy of the histogram. Comput Vision, Graphics, Imag Process 29:273–285
12. Levine MD, Nazif AM (1985) Dynamic measurement of computer generated image segmentations. Pattern Anal Mach Intell, IEEE Trans 155–164
13. Liao P-S, Chen T-S, Chung P-C (2001) A fast algorithm for multilevel thresholding. J Inf Sci Eng 17:713–727
14. Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Computer Vision, 2001. ICCV 2001. Proc Eighth IEEE Int Conf, IEEE. 416–423
15. Morand C, Benois-Pineau J, Domenger J-P, Zepeda J, Kijak E, Guillemot C (2010) Scalable object-based video retrieval in HD video databases. Signal Process Image Commun 25:450–465
16. Narendra KS, Thathachar M (1974) Learning automata-a survey. Syst, Man Cybernet, IEEE Trans 323–334
17. Otsu N (1975) A threshold selection method from gray-level histograms. Automatica 11:23–27
18. Rother C, Kolmogorov V, Blake A (2004) Grabcut: interactive foreground extraction using iterated graph cuts. ACM Trans Graphics (TOG), ACM. 309–314
19. Sathya P, Kayalvizhi R (2011) Optimal multilevel thresholding using bacterial foraging algorithm. Expert Syst Appl 38:15549–15564
20. Sezgin M (2004) Survey over image thresholding techniques and quantitative performance evaluation. J Electron Imag 13:146–168
21. Tao W, Jin H, Liu L (2007) Object segmentation using ant colony optimization algorithm and fuzzy entropy. Pattern Recogn Lett 28:788–796
22. Van der Merwe D, Engelbrecht AP (2003) Data clustering using particle swarm optimization. Evolution Comput, 2003. CEC'03. 2003 Congress, IEEE 215–220
23. Vantaram SR, Saber E (2012) Survey of contemporary trends in color image segmentation. J Electron Imag 21:040901–040928
24. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. Imag Process, IEEE Trans 13:600–612
25. Wang J, Ju L, Wang X (2009) An edge-weighted centroidal Voronoi tessellation model for image segmentation. Imag Process, IEEE Trans 18:1844–1858
26. Yang X-S, Deb S (2010) Engineering optimisation by cuckoo search. Int J Math Model Numer Optim 1:330–343
27. Yazdani D, Arabshahi A, Sepas-Moghaddam A, Dehshibi MM (2012) A multilevel thresholding method for image segmentation using a novel hybrid intelligent approach. Hybrid Intell Syst (HIS), 2012 12th Int Conf IEEE. 137–142
28. Yin P-Y (1999) A fast scheme for optimal thresholding using genetic algorithms. Signal Process 72:85–95
29. Yin P-Y (2007) Multilevel minimum cross entropy threshold selection based on particle swarm optimization. Appl Math Comput 184:503–513
30. Yu X, Bui T, Krzyżak A (1994) Robust estimation for range image segmentation and reconstruction. Pattern Anal Mach Intell, IEEE Trans 16:530–538
31. Zahara E, Fan S-KS, Tsai D-M (2005) Optimal multi-thresholding using a hybrid optimization approach. Pattern Recogn Lett 26:1082–1095
32. Zhang Y, Brady M, Smith S (2001) Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. Med Imag, IEEE Trans 20:45–57
33. Zhang Y, Wu L (2011) Optimal multi-level thresholding based on maximum Tsallis entropy via an artificial bee colony approach. Entropy 13:841–859
34. Zhang L, Zhang L, Mou X, Zhang D (2011) FSIM: a feature similarity index for image quality assessment. Imag Process, IEEE Trans 20:2378–2386
35. Zuva T, Olugbara OO, Ojo SO, Ngwira SM (2011) Image segmentation, available techniques, developments and open issues. Canadian J Imag Process Comput Vision 2:20–29

Mohammad Mahdi Dehshibi received his Ph.D. degree, in Computer Engineering, from Islamic Azad University (IAU), Tehran, Iran, in 2016. From 2008, he has been a Visiting Lecturer at Shahid Beheshti University of Iran (SBU). From 2014, he was with the Pattern Research Center (PRC), where he is the director and responsible for research projects in the area of Pattern Recognition/Formation, and Cellular Automata. Dr. Dehshibi is a member of IEEE, MVIP, and DISWC and has contributed to over 50 papers published in scientific Journals or International Conferences.

Mohammad Sourizaei was born in Zahedan, Iran, in 1987. He received his B.Sc. in Electrical Engineering from Islamic Azad University, Zahedan Branch, Iran, in 2009. He is the co-founder and the presiding officer of Hamoun S&T Company located in the Science and Technology Park, Zahedan, Iran. Hamoun S&T is a Knowledge-based company. His research areas lie in electrical instruments, image processing, and computer science. He is a member of the Young Researchers and Elites Club. He has also invented a new electromagnetic touch key.

Mahmood Fazlali received BSc degree in 2001 from Shahid Beheshti University (SBU) Iran, M.Sc. degree in 2004 from University of Isfahan Iran, and a Ph.D. degree in 2010 from SBU in computer architecture. He performed studies on reconfigurable computing systems in computer engineering lab of the Delft University of Technology, The Netherlands. Currently, he is affiliated as an assistant professor in computer science department at SBU, G. C. His research interest includes multicore and parallel systems, reconfigurable computing, and computer vision.

Omid Talaee was born in Saqqez, Iran on September 2, 1984. He received the B.Sc. degree in Physics in 2008 from Mahabad University and M.Sc. degree from University of Shiraz, Shiraz, Iran in 2012. During 2013–14 he studied Thermal Power Plants and Nuclear Engineering at the Polytechnic University of St. Petersburg, Russia. Currently, he is a Ph.D. student in Medical Engineeringat the University of Shiraz. His research interest includes Design and Construction of Detectors, Radiotherapy, Brachytherapy and Imaging Systems.

Hossein Samadyar is currently a PhD student at Science and Research Branch, Islamic Azad University, Tehran, Iran. His research interest lies in the Industrial Image Processing and Deep Learning.

Jamshid Shanbehzadeh received his B.S. and M.S. degree in electrical engineering from University of Tehran, Tehran, Iran in 1986 and a Ph.D. degree in electrical and computer engineering from Wollongong University, Australia in 1996. He is currently an Associate Professor in the Department of Computer Engineering of Kharazmi University, Tehran, Iran. His research interests include computer vision, image retrieval, image processing/coding, e-learning, and e-content development. He has contributed to over 150 papers published in scientific journals or conference proceedings.