The Journal of Supercomputing (2021) 77:6734–6753
https://doi.org/10.1007/s11227-020-03566-7

Multilevel thresholding using an improved cuckoo search algorithm for image segmentation

Longzhen Duan[1] · Shuqing Yang[1,2] · Dongbo Zhang[3]

Accepted: 10 December 2020 / Published online: 2 January 2021

Abstract

Multilevel thresholding image segmentation is an important technique, which has attracted much attention in recent years. The conventional exhaustive search method for image segmentation is efficient for bilevel thresholding. However, they are time expensive when dealing with multilevel thresholding image segmentation. To better tackle this problem, an improved cuckoo search algorithm (ICS) is proposed to search for the optimal multilevel thresholding in this paper, and Otsu is considered as its objective function. In the ICS, two modifications are used to improve the standard cuckoo search algorithm. First, a parameter adaptation strategy is utilized to improve exploration performance. Second, a dynamic weighted random-walk method is adopted to enhance the local search efficiency. A total of six benchmark test images are used to perform the experiments, and seven state-of-the-art metaheuristic algorithms are introduced to compare with the ICS. A series of measure indexes such as objective function value and standard deviation, PSNR, FSIM, and SSIM as well as the Wilcoxon rank sum and convergence performance are performed in the experiments; the experimental results show that the proposed algorithm is superior to other seven well-known heuristic algorithms.

Keywords Improved cuckoo search algorithm · Image segmentation · Multilevel thresholding · Otsu

✉ Shuqing Yang
ysqsinuo@qq.com

1 School of Information Engineering, Nanchang University, Nanchang 330031, China

2 School of Software, Jiangxi University of Science and Technology, Nanchang 330013, China

3 Guangdong Key Laboratory of Modern Control Technology, Guangdong Institute of Intelligent Manufacturing, Guangzhou 510070, China

1 Introduction

Image segmentation is an important image preprocessing technology. The calculation of thresholding image segmentation is simple and effective [1]. Hence, thresholding has become a popular approach for image segmentation. Basically, thresholding is used to separate an image into targets and background on the basis of the gray histogram. Assume that an image can be divided into two regions, such as the background region and object region, which is known as the bilevel thresholding image segmentation method [2]. In the case of an image containing more than one object, bilevel thresholding does not achieve the desired result [3]. On account of multilevel thresholding, we can accurately segment the image into different significant regions [4]. For traditional exhaustive methods, multilevel thresholding can be very time-consuming. It is an NP-hard optimization problem. In such cases, the metaheuristic algorithms are proposed to solve the optimal multilevel thresholding. Many related works are based on metaheuristic algorithms, such as the improved particle swarm optimization (IPSO) [5], the improved electromagnetism optimization algorithm (IEMO) [6], the modified artificial bee colony (MABC) [7] algorithm, the improved harmony search algorithm (IHSA) [8], the crow search algorithm (CSA) [9], the cuckoo search algorithm (CS) [10], the improved emperor penguin optimization (IEPO) [11], the improved flower pollination algorithm (IFPA) [12], the wind-driven optimization (WDO) [13], the hybrid Harris Hawks optimization (HHHO) [14], and the fuzzy adaptive gravitational search algorithm (FAGSA) [15]. Moreover, a number of modified and improved metaheuristic algorithms have been applied to multilevel thresholding [16–18].

The cuckoo search algorithm is a popular metaheuristic algorithm that was inspired by the egg-laying behavior of natural cuckoo birds [19]. More recently, CS has been widely used in feature selection [20], numerical optimization [21], data clustering [22], and many-objective optimization [23]. Studies have shown that CS is an effective algorithm for solving various optimization problems [24]. The successes recorded by the CS are that fewer parameters need to be adjusted for execution, and the robustness of the algorithm is perfect. However, although the CS algorithm has been well-applied for solving different practical application problems when dealing with complex optimization problems, the CS still needs to be improved to enhance the performance. An adaptive parameter strategy was proposed in the CS by Wang and Zhou. The experimental tests indicated that the proposed algorithm was superior to CS with fixed parameters [25]. Guerrero M et al. [26] proposed a fuzzy cuckoo search (FCS), in which a fuzzy system was designed to dynamically adjust the control parameters. The results demonstrated that the FCS outperformed the standard CS. Walton S et al. [27] presented a modified cuckoo search (MCS). In MCS, a dynamic parameter control strategy for step size α was used. The results showed that the MCS had advantage over DE, PSO, and CS. Wang G et al. [28] proposed a chaotic cuckoo search (CCS), in which 12 chaotic maps were used to adjust the step size of the CS. The experimental results showed that a suitable chaotic map was superior to the CS. Huang et al. proposed a fully informed cuckoo search algorithm (FICS). In FICS, a fully informed strategy was firstly proposed to improve particle swarm optimization. The proposed FICS was tested by four benchmark images; the results suggested that the FICS was better than other algorithms [29]. A hybrid adaptive cuckoo search squirrel search algorithm (ACSS) for brain MR image segmentation was proposed [30]. In ACSS, an adaptive step size strategy was used to improve the convergence and a new hybrid search was also adopted. The experimental results indicated that the ACSS was superior to CS, ACS, and SS. In a word, the standard CS can be categorized as adaptive, self-adaptive, and hybridizing according to the kinds of parameter control strategies. In conclusion, all operations are adopted to balance exploration and exploitation [31].

All the above-mentioned CS variants are aimed to improve the convergence speed and accuracy of the algorithm and then obtain the desired results when solving different optimization problems. The improvement of CS focused on the control parameters. In this paper, we propose novel parameter adaptation strategies to enhance the CS and then apply it to solve the optimal multilevel thresholding. The main contribution of this study is that two modifications are proposed to improve the standard CS: (1) the adaptive control parameters method and (2) the dynamic weighted random-walk strategy. We evaluate the performance of the ICS on six test images as well as seven well-known metaheuristic algorithms. Some measure indexes such as objective function value and standard deviation, PSNR, FSIM, and SSIM as well as the Wilcoxon rank sum and convergence performance are performed in the experiments.

The rest of the paper is organized as follows: Section 2 shows the theory of multilevel thresholding based on the Otsu method. In Sect. 3, the cuckoo search algorithm is described. The improved cuckoo search algorithm is shown in Sect. 4. Experimental results are discussed in Sect. 5. In Sect. 6, conclusions are drawn.

2 Segmentation based on between-class variance

A multilevel threshold value image segmentation method based on the image one-dimensional gray histogram is proposed. Based on that, the image can be segmented into different regions. The process of thresholding image segmentation is to obtain the best objective function value by employing an intelligent optimization algorithm and then obtaining approximate optimal thresholds. The multilevel thresholding image segmentation method is utilized to find the best possible threshold in the segmented histogram by meeting some criteria. In 1979, image thresholding based on the Otsu method was proposed. This method obtains the optimal solution by maximizing the objective function [32]. In the present work, Otsu's nonparametric segmentation method, called between-class variance, is considered. In addition, some nonparametric criteria, such as Kapur's [33] and Tsallsi entropy [34], are used for image thresholding segmentation. A detailed description of the between-class variance method can be found in bilevel thresholding image segmentation. An image can be segmented into two classes, A_1 and A_2 (objects and background), by a threshold at a level "t." Class A_1 encloses the gray levels in the range 0–$t-1$, and class A_2

encloses the gray levels from t to $L-1$. The probability distributions for gray levels A_1 and A_2 can be expressed as:

$$A_1 = \frac{p_0}{\omega_1(t)} \dots \frac{p_{t-1}}{\omega_1(t)}, \quad A_2 = \frac{p_t}{\omega_2(t)} \dots \frac{p_{L-1}}{\omega_2(t)} \tag{1}$$

where p_i is the gray-level probability,$\omega_1(t) = \sum_{i=0}^{t-1} p_i$, $\omega_2(t) = \sum_{i=t}^{L-1} p_i$, and $L = 256$, and the mean levels μ_1 and μ_2 for A_1 and A_2 can be measured by:

$$\mu_1 = \sum_{i=0}^{t-1} \frac{ip_i}{\omega_1(t)}, \quad \mu_2 = \sum_{i=t}^{L-1} \frac{ip_i}{\omega_2(t)} \tag{2}$$

The average intensity (μ_T) of the entire image can be calculated by:

$$u_T = \omega_1 \mu_1 + \omega_2 u_2, \qquad \omega_1 + \omega_2 = 1 \tag{3}$$

The objective function for the bilevel thresholding problem can be defined as:

$$F_t^{\text{opt}} = \arg\max \left(\delta_1 + \delta_2 \right) \tag{4}$$

where $\sigma_1 = \omega_1 \left(u_1 - u_T \right)^2$ and $\sigma_2 = \omega_2 \left(u_2 - u_T \right)^2$.

Bilevel thresholding can be extended to a multilevel thresholding problem by increasing the number of threshold values "m" as follows. Assume that there are "m" thresholds (t_1, t_2…,t_m), which divide the image into "$m+1$" classes: A_1 with gray levels in the range 0–$t-1$, A_2 with enclosed gray levels in the range t_1–$t_{2\_}1$…, and A_{m+1} with gray levels from t_m to $L-1$. The objective function for the multilevel thresholding problem can be measured by [35]:

$$F_t^{\text{opt}} = \arg\max \left(\delta_1 + \delta_2 + \dots + \sigma_{m+1} \right) \tag{5}$$

where $\sigma_1 = \omega_1 \left(u_1 - u_T \right)^2, \sigma_2 = \omega_2 \left(u_2 - u_T \right)^2, \dots, \sigma_{m+1} = \omega_{m+1} \left(u_{m+1} - u_T \right)^2$.

3 Cuckoo search algorithm

The cuckoo search algorithm is a nature-inspired algorithm that was proposed by Yang in 2009. The CS mimics the process of cuckoo egg-laying. Cuckoos normally lay their fertilized eggs in host nests with the hope of their offspring being raised by proxy parents. Sometimes, the host identifies that the eggs in their nests do not belong to them. Under these circumstances, the foreign eggs are thrown out of the nests, or the whole nests are discarded. The CS optimization algorithm is generally based on the following three principles:

1. Interestingly, each cuckoo bird lays one egg at a time and randomly places its egg in a host bird's nest.
2. Usually, the best nests containing high-quality eggs are carried over to the next generations.

3. The number of available host nests is fixed. The host bird discovers foreign eggs with a probability p_α, and the range of p_α is from 0 to 1. Note that the best nests are selected for further calculations. For simplicity, principle 3 can be explained as follows: the n nests will be replaced by new nests with a probability p_α.

Based on these three principles, the CS process can be summarized as follows: While generating new solution x_i^{t+1} for cuckoo i, a Lévy flight is performed [36]:

$$x_i^{t+1} = x_i^t + \alpha_0\left(x_i^t - x_{\text{best}}\right) \oplus \text{Levy}(s, \lambda) \tag{6}$$

where α_0 is the step size, $\alpha_0 > 0$, and x_{best} represents the current optimal solution. Lévy flights are drawn from a Lévy distribution, which can be defined by:

$$\text{Levy}(s, \lambda) \sim u = t^{-\lambda}, \quad (1 < \lambda \leq 3) \tag{7}$$

where

$$\text{Levy}(s, \lambda) = \frac{\lambda\Gamma(\lambda)\sin(\pi\lambda/2)}{\pi}\frac{1}{s^{1+\lambda}}, \quad \left(s >> s_0 > 0\right) \tag{8}$$

where $\Gamma(\lambda)$ is the standard gamma function with an index λ.

In the CS algorithm, the worst nest is abandoned with a probability p_α, and a new nest is built with random walks by the following formula:

$$x_i^{t+1} = x_i^t + r\left(x_j^t - x_k^t\right) \tag{9}$$

where r represents a random number and x_j^t and x_k^t are the random solutions at iteration t. The pseudocode of the CS algorithm is summarized in Table 1.

4 The improved cuckoo search algorithm

CS is also a metaheuristic global search algorithm that is widely used to solve different optimization problems, such as gray image segmentation [37] or color image segmentation [38]. The CS has fewer parameters compared to other algorithms. It is easy to set the parameter of the algorithm. Hence, CS may be useful for nonlinear problems and many-objective optimizations. Although CS is efficient, it has some drawbacks, such as time consumption and premature convergence for a number of real-world optimization problems. Therefore, the basic structure of the CS has been modified to improve its performance.

4.1 Adaptive control parameters

The control parameters are sensitive to the performance of the metaheuristic algorithms; a lot of parameter strategies are proposed to improve the performance. The control parameters like linear, piece-wise, or curve decrease with the generation, which is called adaptive parameter strategy [29], if the control parameters

Table 1 Pseudocode for the cuckoo search algorithm

Cuckoo Search Algorithm
Begin
1. Objective function $f(x)$, $x=(x_1,x_2...,x_d)$;
2. Initialize a population of n host nests, $x_i(i=1,2...,n)$;
3. For $G=1:G_m$
4. Get a cuckoo (m) randomly by Lévy flights, $Levy(s,\lambda)\sim u=t^{-\lambda}$;
5. Generate new nests, $x_i^{t+1}=x_i^t+\alpha_0\left(x_i^t-x_{best}\right)\oplus Levy(s,\lambda)$;
6. Evaluate its quality/fitness F_m;
7. Choose a nest among n (n) randomly;
8. if ($F_m>F_n$)
9. Replace j with the new solution;
10. end
11. If ($r>p_\alpha$)
12. Abandon a fraction of the worst nests;
13. and build new ones at new locations via $x_i^{t+1}=x_i^t+r\left(x_j^t-x_k^t\right)$;
14. end
15. Keep the best solutions (or nests with quality solutions);
16. Rank the solutions and find the current best;
17. end
End

changed with the fitness value or optimization problem named self-adaptive strategy such as fuzzy parameter strategy and parameter archiving mechanism [16]. In CS, the parameters p_α and α_0 are introduced to help the algorithm find globally locally improved solutions. The parameters p_α and α_0 are very important parameters and can potentially be utilized in controlling the convergence rate. The original CS algorithm adopts a fixed value for both p_α and α_0. In facing different problems, the parameters should be adjusted based on personal experience. If the value of p_α is small and the value of α_0 is large, the performance of the algorithm will be poor and lead to a considerable increase in the number of iterations. If the value of p_α is large and the value of α_0 is small, the speed of convergence is high, but it may be unable to find the best solutions [31]. To improve the performance of the CS algorithm, we propose adaptive variables p_α and α_0. At the beginning of the iteration, the values of p_α and α_0 must be large enough to increase the diversity of the population. However, these values should decrease in the final generations. In this paper, we propose an adaptive control parameter strategy. The values of p_α and α_0 dynamically change with the number of generations and are expressed in the following equations:

$$p_\alpha = p_{\alpha i} \cdot 2^\tau,\ \tau = e^{1-\frac{G_m}{G_m+1-G}} \tag{10}$$

$$\alpha_0 = 0.5 * \exp\left(-\frac{G-1}{G_m}\right) \tag{11}$$

where $p_{\alpha i}$ is a predefined constant, G_m is the maximum number of iterations, and G is the current number of iterations. Figure 1 shows the parameter values changed with the number of generation when $G_m = 1000$, $p_{ai} = 0.25$.

4.2 A dynamic weighted random-walk strategy

A new nest built with random walks often leads to a slow convergence rate and vibration. To enhance the local search, we propose that a larger ω leads to greater control of exploration or exploitation of host nest positions (solutions). Based on formulas 12 and 13, ω linearly decreases from a relatively large value to a small value throughout the course so that the CS can effectively enhance the local search ability.

$$x_i^{t+1} = \omega x_i^t + r\left(x_j^t - x_k^t\right) \tag{12}$$

$$\omega = \omega_{\max} - \frac{G\left(\omega_{\max} - \omega_{\min}\right)}{G_m} \tag{13}$$

where ω is a weighted coefficient. $\omega_{\max}$ and $\omega_{\min}$ are user-defined constants. Correspondingly, the pseudocode of the improved CS algorithm is summarized in Table 2.

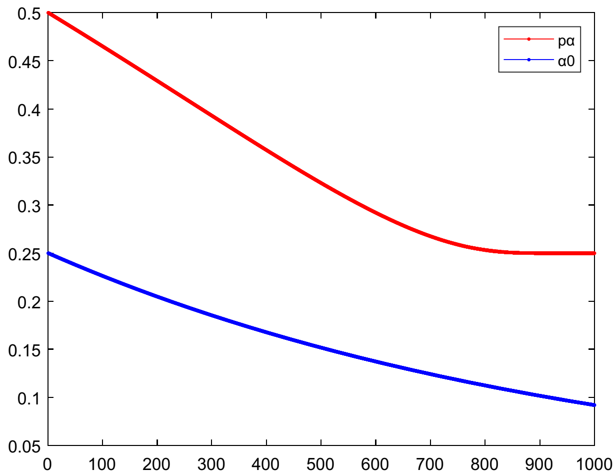


Fig. 1 The parameter values changed with the number of generations

Table 2 Pseudocode for an improved cuckoo search algorithm

Improved Cuckoo Search Algorithm
Begin
1. Objective function $f(x)$, $x=(x_1,x_2...,x_d)$;
2. Initialize parameter p_{ai},ω_{max},ω_{min},
3. Initialize a population of n host nests, x_i(i=1,2...,n);
4. **For** G=1:G_m
5. Generate p_a, α_0,ω;
6. Get a cuckoo (m) randomly by Lévy flights, $\mathrm{Levy}(s,\lambda)\sim u=t^{-\lambda}$;
7. Generate new nests, $x_i^{t+1}=x_i^t+\alpha_0\left(x_i^t-x_{best}\right)\oplus Levy(s,\lambda)$;
8. Evaluate its quality/fitness F_m;
9. Choose a nest among n (n) randomly;
10. **if** (F_m>F_n)
11. Replace j with the new solution;
12. **end**
13. **If** (r>p_a)
14. Abandon a fraction of the worst nests;
15. and build new ones at new locations via $x_i^{t+1}=\omega x_i^t+r\left(x_j^t-x_k^t\right)$;
16. **end**
17. Keep the best solutions (or nests with quality solutions);
18. Rank the solutions and find the current best;
19. **end**
End

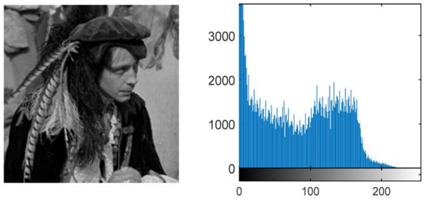
5 Experiments and analysis

In this section, the performance of the proposed algorithm was evaluated by six test images. In the following, the objective function value, image segmentation quality, and Wilcoxon rank sum are compared with FCS [27], MCS [28], CS [10], FICS [29], ACSS [30], FAGSA [16], and IEPO [11].

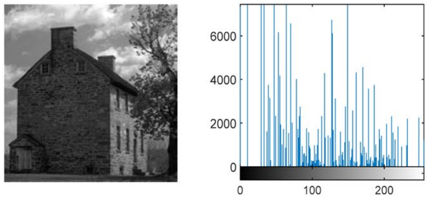
5.1 Experimental setting

A maximum between-class variance based on the proposed ICS was tested under a set of benchmark images. Six classic images from the Benchmarks 500 named "Hunter," "House," "Baboon," "Couple," "Lena," and "Peppers" are used in the

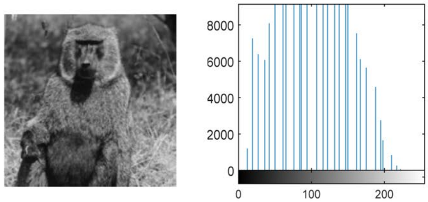
Fig. 2 The test images and corresponding histograms



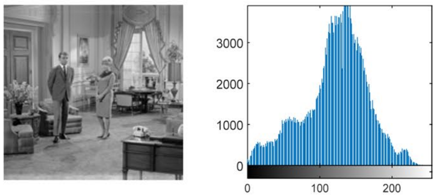
(a) Hunter



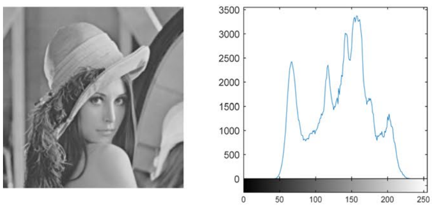
(b) House



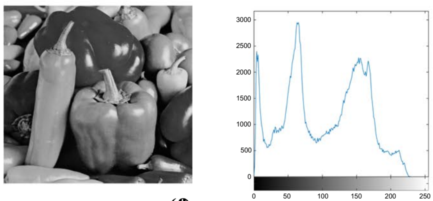
(c) Baboon



(d) Couple



(e) Lena



(f) Peppers

experiment. The size of all images is 512×512. The test images and corresponding histograms are shown in Fig. 2.

The experiments were performed on a Lenovo Laptop with an Intel Core i7 processor and 8 GB memory, running the Windows 10 operating system. The algorithm was carried out by MATLAB R2019a. In the next subsections, the ICS will be compared with CS variants such as CS, FCS, MCS, FICS, ACSS and other two newly proposed metaheuristic algorithms FAGSA and IEPO. The comparison with different algorithms was mainly used to test the advantages of the ICS. For the sake of fairness, all the algorithms were performed under the same conditions. Generally, the thresholds were set as $m=2$, 3, 4, 5. The number of maximum iterations was 300; the population size was 30. All experiments were repeated 25 times. The corresponding parameters used for the presented four algorithms are listed in Table 3, which originate from related references.

5.2 Results on the objective function value

As the aim of multilevel thresholding is to maximize the given objective function, the objective function value obtained by the involved algorithms directly shows the algorithm's performance. In detail, the mean objective function values are shown in Table 4 (the optimal value is marked in bold) and Fig. 3 (the average objective function value), where "*m*" stands for the number of thresholds. The corresponding standard deviation values are given in Table 5.

From Table 4 and Fig. 3, it is clear that our algorithm outperformed the other methods, and the ICS obtained the best results in 22 cases (total 30 cases), whereas MCS was the second-best but did not show an advantage in any case when compared meticulously with the other six algorithms. However, the differences in the maximum objective function values obtained by the eight algorithms are not more than 10. Correspondingly, the standard deviation of the objective function values is

Table 3 Parameters of the heuristic algorithms

Algorithms	Parameters	Value or range
FAGSA	Inertia weight (ω)	[0.2, 1.2]
	Control parameter (α)	[10, 50]
IEPO	Levy (L)	1.5
FICS	Probability (p_α)	0.5
	Scale factor (a)	1.5
	Variable (Ne)	3
ACSS	Probability (p_α)	0.1
	Fly constant	1.9
CS	Probability (p_α)	0.25
	Variable α_0	0.01
ICS	Variable $p_{\alpha i}$	0.25
	Weighted values $\omega_{\max}$,$\omega_{\min}$	0.9, 0.2
FCS	Variable α_0	0.01
MCS	Probability (p_α)	0.75

Table 4 Comparison of the mean objective function values

Images	*m*	ICS	FCS	MCS	CS	FICS	ACSS	FAGSA	IEPO
Hunter	2	**3064.2**	3064.1	**3064.2**	3063.8	**3064.2**	3064.1	3064.0	3064.1
	3	3213.4	3212.5	**3213.5**	3212.3	3212.3	3211.8	3210.1	3211.4
	4	**3269.5**	3268.1	3269.3	3267.9	3269.1	3268.6	3268.5	3268.2
	5	**3308.1**	3305.5	3307.8	3304.1	3307.6	3306.4	3305.2	3307.6
House	2	**3543.9**	3542.6	**3543.9**	3542.1	3543.8	3542.3	3542.1	3543.5
	3	**3755.6**	3753.4	3755.4	3752.8	3755.1	3754.6	3753.6	3754.9
	4	3846.9	3845.7	3846.5	3845.0	3845.9	3844.9	3843.2	**3847.2**
	5	**3902.6**	3899.3	3898.4	3896.5	3901.4	3901.7	3901.3	3901.1
Baboon	2	**1609.2**	1604.1	**1609.2**	1603.7	1608.2	1608.8	1607.2	1608.5
	3	1751.2	1740.2	1751.0	1738.1	1750.9	1750.2	1749.8	**1751.4**
	4	1798.4	1794.6	1798.2	1787.0	1794.3	**1798.6**	1796.3	1796.6
	5	**1850.1**	1848.8	1847.8	1845.6	1849.5	1848.7	1849.6	1848.7
Couple	2	1616.5	**1616.6**	1616.4	1614.7	1616.5	1615.4	1615.2	1615.7
	3	**1758.9**	1752.4	1758.6	1748.7	1757.7	1757.2	1750.6	1758.5
	4	1828.6	**1828.9**	1828.4	1827.5	1827.4	1827.9	1827.9	1826.2
	5	**1870.6**	1867.5	1869.3	1867.0	1868.9	1866.2	1868.5	1867.7
Lena	2	**1416.5**	1415.3	1416.2	1415.1	1415.3	1415.7	1415.8	1415.9
	3	1604.9	1603.2	1604.3	1602.5	**1605.1**	1602.8	1603.1	1603.2
	4	**1646.1**	1645.7	1645.7	1644.3	1445.7	1644.2	1644.7	1645.5
	5	**1668.1**	1667.3	1667.8	1666.2	1667.2	1666.5	1666.9	1667.8
Peppers	2	**2803.2**	2802.1	2802.6	2801.8	2803.1	2802.6	2802.3	2802.0
	3	**3066.1**	3065.4	3065.9	3064.4	3065.7	3065.5	3064.7	3064.5
	4	3139.8	3137.9	**3140.2**	3136.7	3140.1	3139.1	3136.4	3139.2
	5	**3194.5**	3193.4	3193.8	3191.6	3193.3	3193.4	3192.8	3193.7

Bold values represent the optimal solution

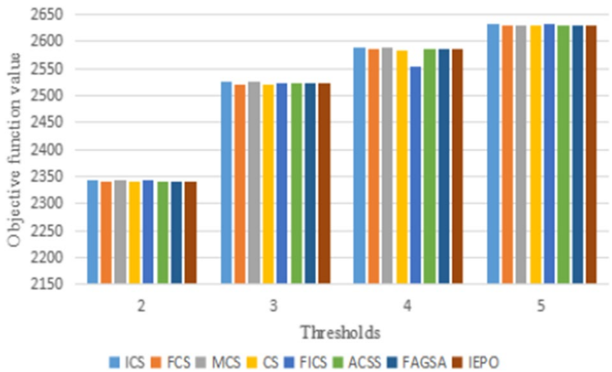


Fig. 3 The average objective function values of the algorithms over all images

shown in Table 5; the results show that the stability of the ICS was the best, which obtained the best results in 21 cases (total 30 cases). However, the stability of ACSS, FICS, FCS, and MCS was worse than that of the ICS. From the result analysis, the proposed algorithm is better than other algorithms in most case, but for the image "Baboon," the algorithm did not perform well, but the difference of the objective function values is small.

Table 5 Results of the standard deviation of objective function values

Images	m	ICS	FCS	MCS	CS	FICS	ACSS	FAGSA	IEPO
Hunter	2	0.01196	0.14925	**0.01194**	0.01540	0.01430	0.01463	0.01526	0.01356
	3	0.03054	0.03083	0.03061	0.03143	0.03057	0.03014	0.03128	**0.02986**
	4	**0.03219**	0.03456	0.32255	0.03571	0.03124	0.03456	0.03345	0.03247
	5	**0.04316**	0.05687	0.04423	0.05723	0.04965	0.05429	0.05229	0.05143
House	2	0.01844	0.02064	**0.01832**	0.01983	0.01895	0.01862	0.01912	0.01899
	3	**0.02029**	0.02625	0.02159	0.02413	0.02354	0.02153	0.02564	0.02453
	4	0.02981	0.03542	0.03012	0.03036	0.03452	**0.02864**	0.03141	0.03246
	5	0.04143	0.04764	0.04151	0.04154	0.04219	0.04256	0.04216	**0.04029**
Baboon	2	**0.01646**	0.01754	**0.01646**	0.01861	0.01658	0.01753	0.01821	0.01784
	3	**0.02421**	0.03767	0.02516	0.03847	0.02546	0.02654	0.03414	0.02536
	4	**0.03317**	0.34671	0.03421	0.03914	0.03452	0.03452	0.03692	0.03799
	5	**0.04251**	0.04778	0.04627	0.04472	0.04339	0.04443	0.04671	0.04561
Couple	2	0.01208	**0.01177**	0.01206	0.01842	0.01425	0.01354	0.01539	0.01638
	3	**0.02230**	0.02751	0.02341	0.02561	0.02356	0.02351	0.02286	0.02452
	4	0.02992	0.03064	0.03015	0.03374	**0.02920**	0.03129	0.03412	0.03157
	5	**0.03842**	0.03956	0.03945	0.04221	0.04113	0.04101	0.04216	0.03976
Lena	2	**0.01012**	0.15698	0.01357	0.02138	0.01121	0.01246	0.01356	0.01324
	3	0.01894	0.02654	0.02395	0.03135	**0.01813**	0.02134	0.02265	0.02013
	4	**0.02596**	0.03365	0.03124	0.04312	0.03252	0.02861	0.02934	0.02968
	5	**0.03568**	0.04269	0.04031	0.05321	0.03754	0.03875	0.03961	0.03754
Peppers	2	**0.01354**	0.01965	0.01652	0.02013	0.01510	0.01946	0.01893	0.01689
	3	0.02631	0.03652	0.02985	0.03651	**0.02623**	0.02864	0.02768	0.25897
	4	**0.02963**	0.03845	0.03125	0.04569	0.02865	0.03218	0.03429	0.31569
	5	**0.03784**	0.04564	0.04023	0.05612	0.03869	0.04123	0.04326	0.04326

Bold values represent the optimal solution

5.3 Quality measures of segmented images

To further verify the performance of the proposed algorithm, the time consumption (seconds) and appropriate performance indicators, including the peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and feature similarity (FSIM), were calculated to evaluate the corresponding image segmentation performance of the results. Image quality measurement was performed using PSNR, which is defined as [39]:

$$\mathrm{PSNR} = 20\log_{10}\left(\frac{255}{\mathrm{RMSE}}\right), \qquad (\mathrm{dB}) \tag{14}$$

where

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(I(i,j) - C(i,j))^2}{M \times N}} \tag{15}$$

and C and I are original and segmented images of size $M \times N$, respectively. A higher value of PSNR indicates a better quality segmented image.

The SSIM was used to compare the structures of the original and segmented images. The SSIM index is described as follows [40]:

$$\mathrm{SSIM}(C,I)=\frac{\left(2\mu_C\mu_I+C_1\right)\left(2\delta_{CI}+C_2\right)}{\left(\mu_C^2+\mu_I^2+C_1\right)\left(\delta_C^2+\delta_I^2+C_2\right)} \tag{16}$$

where μ_C and δ_C represent the pixel mean and variance, respectively, of the original image; μI and δI represent the pixel mean and variance, respectively, of the segmented image; δ_{CI} is the covariance of the original image and the segmented image; and C_1 and C_2 are constants, where $C_1=C_2=6.5025$. A higher value of SSIM indicates better performance. In addition, the FSIM is employed to measure the feature similarity between two images. It is calculated between two images C and I as [41]:

$$\mathrm{FSIM}=\frac{\sum_{x\in\Omega}S_L(x)PC_m(x)}{\sum_{x\in\Omega}PC_m(x)} \tag{17}$$

where

$$\begin{aligned}
S_L(x)&=S_{\mathrm{PC}}(x)S_G(x);\\
S_{\mathrm{PC}}(x)&=\frac{2\mathrm{PC}_1(x)\mathrm{PC}_2(x)+T_1}{\mathrm{PC}_1^2(x)+\mathrm{PC}_2^2(x)+T_1};\\
S_G(x)&=\frac{2G_1(x)G_2(x)+T_2}{G_1^2(x)+G_2^2(x)+T_2};\\
\mathrm{PC}_m(x)&=\max\left\{\mathrm{PC}_1(x),\mathrm{PC}_2(x)\right\}.
\end{aligned} \tag{18}$$

T_1 and T_2 are constants. Here, we choose $T_1=0.85$ and $T_2=160$ in the experiments, $G(x)$ represents the gradient magnitude of an image, and $\mathrm{PC}(x)$ is the phase congruency of an image [12]. A higher value of FSIM indicates better performance.

Furthermore, Table 6 and Fig. 4 display the PSNR index values, Table 7 and Fig. 5 show the SSIM index values, and Table 8 and Fig. 6 show the FSIM index values. The experimental results show that the ICS obtained the best values in 23 cases, 25 cases, and 25 cases for PSNR, SSIM, and FSIM, respectively. The experimental results show that the ICS outperforms the other seven algorithms in image segmentation quality, but IEPO, MCS, ICS, and FICS have little difference on PSNR, SSIM, and FSIM.

5.4 Statistical results analysis

A Wilcoxon rank sum is performed to verify the experimental results, which has been conducted with 5% significance level [42]. The null hypothesis is expressed as: There is no significant difference between the ICS algorithm and other seven algorithms. Nevertheless, the alternative hypothesis seems a significant difference among them. Based on the Wilcoxon rank sum, the p value is used to judge the null hypothesis whether to accept the null hypothesis or not. If the p value is greater

Table 6 Comparison of the mean PSNR

Images	*m*	ICS	FCS	MCS	CS	FICS	ACSS	FAGSA	IEPO
Hunter	2	**18.9237**	18.9235	**18.9237**	18.9128	**18.9237**	18.9124	18.9131	18.9134
	3	20.8898	20.8886	20.8771	20.7541	**20.8914**	20.8861	20.7698	20.7759
	4	**22.5912**	22.2013	22.5346	21.9999	22.4865	22.0253	22.5273	22.5894
	5	**23.9406**	23.8985	23.9316	23.8061	23.9342	23.8179	23.8124	23.9347
House	2	21.0346	21.0342	**21.0350**	21.0339	21.0336	21.0319	21.0321	21.0368
	3	**23.0367**	23.0345	23.0351	22.9629	23.0289	22.9897	22.9754	23.0425
	4	25.0831	24.9532	25.0830	24.9214	25.0726	24.9982	24.9876	**25.0969**
	5	**26.3729**	26.3610	26.3657	26.3444	26.3621	26.1254	26.3510	26.2713
Baboon	2	**19.3728**	**19.3728**	19.3725	19.3714	19.3721	19.3714	19.3684	19.3685
	3	22.0215	21.9872	22.0210	21.8219	22.0126	22.0292	22.0193	**22.0235**
	4	24.8939	24.7519	24.8812	24.3198	24.8836	**24.8967**	24.5692	24.8213
	5	**26.9548**	26.8614	26.9431	25.8035	26.9125	26.8387	26.6014	26.9401
Couple	2	**20.0459**	**20.0459**	**20.0459**	19.9824	20.0133	20.0351	20.0362	20.0364
	3	**22.9763**	22.9673	22.9684	22.9742	22.9567	22.9682	22.9015	22.9325
	4	**25.1360**	25.1248	25.1159	24.8227	25.1129	25.1351	25.1352	25.1246
	5	**26.9398**	26.9261	26.9395	26.5745	26.8453	26.8955	26.9384	26.9012
Lena	2	**19.8467**	19.7865	19.8356	19.7234	19.8365	19.8278	19.8312	19.8322
	3	22.9805	22.9012	22.3024	22.7658	**22.9814**	22.8647	22.8965	22.9175
	4	**24.2904**	24.1026	24.1523	23.9865	24.2698	23.9846	24.2013	24.2013
	5	**24.7540**	24.6243	24.6952	24.1356	24.6846	24.5275	24.7361	24.4731
Peppers	2	**18.7187**	18.6128	18.6985	18.5962	18.7169	18.7012	18.7023	18.7125
	3	**21.3342**	21.2236	21.1256	21.0986	21.3326	21.3267	21.3269	21.3325
	4	25.1712	24.9755	**25.2322**	24.9358	25.1682	25.1025	25.1476	24.1477
	5	**26.4906**	25.8937	26.3982	25.8659	26.4239	26.4192	26.4307	25.4093

Bold values represent the optimal solution

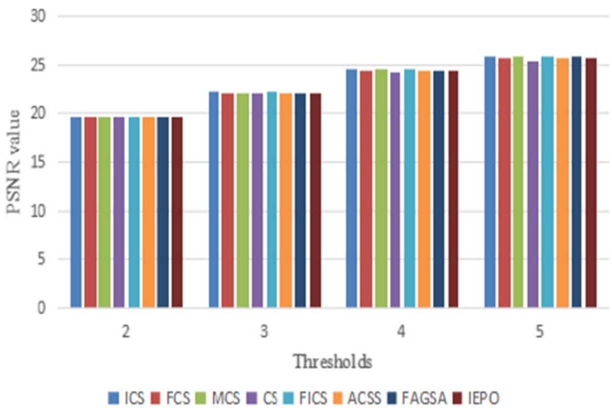


Fig. 4 The average PSNR values of the algorithms over all images

than 0.05, the null hypothesis will be rejected; it implies there is no significant difference between all algorithms. On the contrary, if the *p* value is less than 0.05, the alternative hypothesis will be accepted. Table 9 shows the *p* value results, which is executed on the objective function value, PSNR, SSIM, and FSIM. It can be summarized from Table 9 that there is no significantly difference between ICS and FCS

Table 7 Comparison of the mean SSIM

Images	m	ICS	FCS	MCS	CS	FICS	ACSS	FAGSA	IEPO
Hunter	2	0.4108	0.4108	**0.4120**	0.4107	**0.4120**	0.4104	0.4100	0.4124
	3	**0.5217**	0.5209	0.5211	0.5131	0.5212	0.5189	0.5142	0.5196
	4	**0.5792**	0.5761	0.5814	0.5620	0.5739	0.5721	0.5626	0.5784
	5	**0.6496**	0.6410	0.6425	0.6441	0.6489	0.6482	0.6451	0.6425
House	2	**0.3722**	0.3722	**0.3722**	0.3694	0.3721	0.3720	0.3721	0.3720
	3	**0.6501**	0.6499	0.6493	0.6495	0.6463	0.6414	0.6452	0.6498
	4	0.7431	0.7420	0.7354	0.7412	0.7396	0.7423	0.7423	**0.7442**
	5	**0.7959**	0.7952	0.7944	0.7955	0.7926	0.7938	0.7942	0.7936
Baboon	2	**0.3849**	0.3847	0.3846	0.3829	0.3845	0.3836	0.3838	0.3842
	3	0.5193	0.5186	0.5190	0.5183	0.5182	0.5191	0.5182	**0.5198**
	4	0.5969	0.5945	0.5970	0.5927	**0.5979**	0.5893	0.5886	0.5971
	5	**0.6847**	0.6840	0.6842	0.6833	0.6826	0.6425	0.6402	0.6837
Couple	2	**0.4496**	0.4482	0.4491	0.4471	0.4482	0.4479	0.4476	0.4489
	3	**0.5435**	0.5424	0.5431	0.5421	0.5412	0.5401	0.5392	0.5431
	4	**0.6179**	0.6153	0.6171	0.6139	0.6168	0.6132	0.6128	0.6170
	5	**0.6848**	0.6788	0.6841	0.6684	0.6843	0.6752	0.6475	0.6842
Lena	2	**0.4384**	0.4273	0.4296	0.4125	0.4256	0.4266	0.4251	0.4296
	3	0.5336	0.5216	0.5247	0.5139	**0.5362**	0.5312	0.5306	0.5312
	4	**0.5636**	0.5489	0.5538	0.5386	0.5596	0.5542	0.5412	0.5624
	5	**0.5987**	0.5845	0.5892	0.5737	0.5821	0.5791	0.5687	0.5943
Peppers	2	**0.3687**	0.3587	0.3598	0.3458	0.3542	0.3641	0.3640	0.3645
	3	**0.4260**	0.4145	0.4186	0.4032	0.4129	0.4130	0.4126	0.4267
	4	**0.4875**	0.4691	0.4769	0.4596	0.4865	0.4793	0.4785	0.4862
	5	**0.5084**	0.4892	0.4988	0.4789	0.5012	0.4905	0.4936	0.5021

Bold values represent the optimal solution

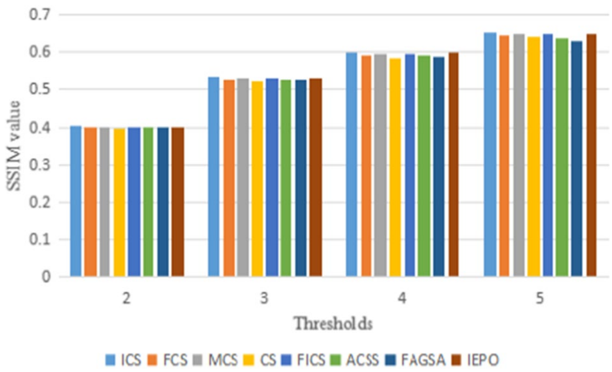


Fig. 5 The average SSIM values of the algorithms over all images

MCS, CS, FICS, ACSS, FAGSA, and IEPO for all metrics. Analyzing the whole data, the ICS has a remarkable improvement, and it is feasible and effective for multilevel thresholding image segmentation.

Table 8 Comparison of the mean FSIM

Images	*m*	ICS	FCS	MCS	CS	FICS	ACSS	FAGSA	IEPO
Hunter	2	**0.7760**	0.7756	**0.7760**	0.7752	**0.7760**	0.7758	0.7757	**0.7760**
	3	**0.8488**	0.8486	0.8487	0.8483	0.8452	0.8445	0.8481	0.8485
	4	0.8906	0.8901	0.8902	0.8888	0.8925	**0.8915**	0.8902	0.8903
	5	**0.9214**	0.9209	0.9211	0.9207	0.9212	0.9210	0.9201	0.9211
House	2	**0.7726**	0.7698	0.7700	0.7692	0.7695	0.7691	0.7682	0.7726
	3	**0.8722**	**0.8722**	0.8721	0.8721	0.8712	0.8718	0.8712	0.8720
	4	0.9093	0.9054	0.9091	0.9049	0.9082	0.9054	0.9041	**0.9097**
	5	**0.9278**	0.9264	0.9273	0.9259	0.9236	0.9285	0.9264	0.9243
Baboon	2	**0.6898**	**0.6898**	0.6894	0.6876	0.6892	0.6895	0.6878	0.6890
	3	0.7892	0.7881	0.7889	0.7878	0.7885	0.7890	0.7852	**0.7902**
	4	0.8454	0.8448	0.8452	0.8352	0.8463	**0.8462**	0.8385	0.8446
	5	**0.9010**	0.9001	0.9008	0.8879	0.8969	0.8964	0.8936	0.8998
Couple	2	**0.7415**	0.7358	0.7415	0.7324	0.7389	0.7358	0.7352	0.7345
	3	**0.8212**	0.8210	0.8211	0.8209	0.8110	0.8223	0.8201	0.8211
	4	**0.8720**	0.8719	0.8716	0.8717	0.8689	0.8642	0.8706	0.8713
	5	**0.9067**	0.9056	0.9061	0.9044	0.9051	0.9045	0.9052	0.9049
Lena	2	**0.7354**	0.7125	0.7322	0.7102	0.7352	0.7263	0.7284	0.7342
	3	0.8269	0.8112	0.8285	0.8024	**0.8294**	0.8215	0.8211	0.8269
	4	**0.8684**	0.8439	0.8568	0.8329	0.8675	0.8569	0.8438	0.8677
	5	**0.8925**	0.8842	0.8913	0.8768	0.8891	0.8814	0.8795	0.8856
Peppers	2	**0.7270**	0.7025	0.7128	0.6986	0.7162	0.7187	0.7168	0.7213
	3	**0.7817**	0.7658	0.7715	0.7524	0.7765	0.7625	0.7603	0.7796
	4	**0.8415**	0.8156	0.8389	0.8025	0.8357	0.8269	0.8569	0.8413
	5	**0.8677**	0.8357	0.8568	0.8169	0.8574	0.8477	0.8365	0.8657

Bold values represent the optimal solution

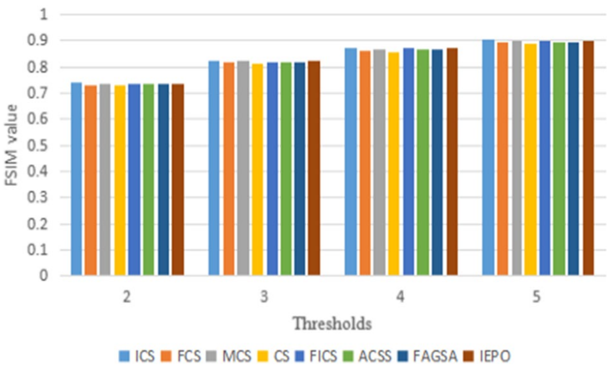


Fig. 6 The average FSIM values of the algorithms over all images

5.5 Convergence performance

The nature-inspired metaheuristic algorithms are stochastic process that execute randomly operations. For this reason, it is not practical to conduct a complexity analysis from a deterministic point of view. However, it is possible to have an idea of this complexity through the mathematical notation called Big O notation [43]. To find the

Table 9 Wilcoxon rank-sum test results

Metrics	Vs.ICS	*p*-value	Metrics	Vs.ICS	*p*-value
Objective function value	FCS	0.8125	*PSNR*	FCS	0.7336
	MCS	0.8366		MCS	0.7965
	CS	0.7807		CS	0.5990
	FICS	0.8123		FICS	0.8366
	ACSS	0.8285		ACSS	0.7966
	FAGSA	0.7966		FAGSA	0.7336
	IEPO	0.8446		IEPO	0.9424
FSIM	FCS	0.6061	SSIM	FCS	0.8045
	MCS	0.8045		MCS	0.8689
	CS	0.5027		CS	0.6725
	FICS	0.8181		FICS	0.7966
	ACSS	0.7763		ACSS	0.7028
	FAGSA	0.6553		FAGSA	0.6279
	IEPO	0.8186		IEPO	0.8608

optimal solutions, the proposed metaheuristic algorithms have an initialization stage and a subsequent stage of iterations. The computational complexity of nature-inspired algorithm depends upon n, Popsize and Maxiter*:*

$$\text{Computation complexity} = \text{O}(n \times \text{Popsize} \times \text{Maxiter}) \tag{19}$$

For all the proposed metaheuristic algorithms, the maximum computational complexity in terms of Big O notation is

$$\text{Computation complexity} = \text{O}(n \times 30 \times 300) = \text{O}(9000n) \tag{20}$$

From the analysis of computational complexity, it may be difficult to judge the performance of each algorithm. To further measure the ICS, the convergence

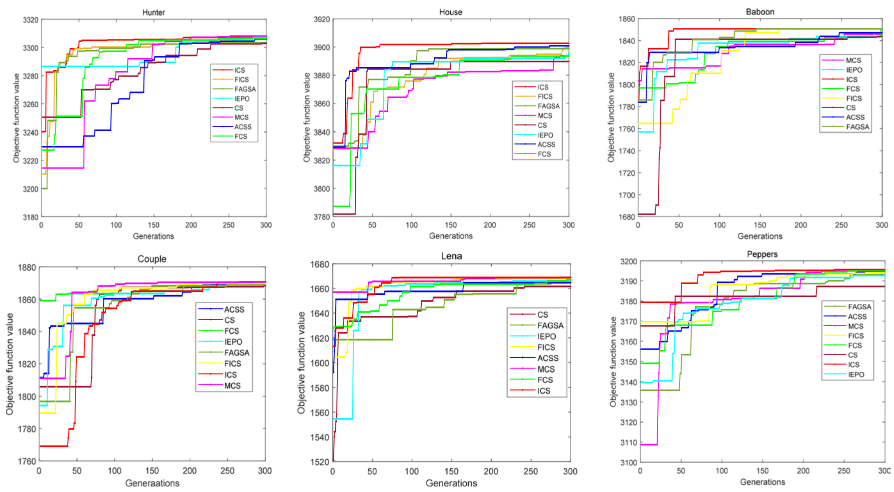


Fig. 7 The convergence performance of all algorithms

performance on 5 levels of thresholding was also studied. For the sake of fairness, all the parameters and settings keep it the same as the previous setting. In the same way, the six test images are tested to exemplify the performance. The results are shown in Fig. 7. It can be found in Fig. 7 that most of the algorithms like FCS, MCS, CS, FICS, ACSS, FAGSA, and IEPO encountered the premature convergence, whereas only ICS overcame the problems and obtained the best objective function values. It also demonstrated that the success of the improved CS is effective. In conclusion, the proposed algorithm achieved significant improvements based on the standard CS, and the other seven algorithms seem a little difference in convergence performance.

6 Conclusion and future work

This paper proposed an improved cuckoo search algorithm for multilevel thresholding image segmentation. Two modifications were adopted. First, an adaptive control parameter method was utilized to perform effective exploration. Second, the dynamic weighted random-walk strategy was used to improve exploitation and enhance the convergence. The experiments demonstrated the success of our modification. In the experiments, the results show that the proposed algorithm is better than other seven state-of-the-art metaheuristic algorithms for multilevel thresholding in terms of the objective function value, standard deviation PSNR, SSIM, FSIM, Wilcoxon rank sum and convergence performance. Future works of this study will focus on studying the performance of the proposed algorithm on remote sensing images.

Acknowledgments This work is supported by National Natural Science Foundation of China (61972184, 61562032, 61662027, 61762042); Modern Agricultural Research Collaborative Innovation Project of Jiangxi (JXXTCXQN201906); Special Fund Project for Graduate Innovation of Jiangxi (YC2017-B065)

References

1. Elaziz MA, Lu S (2019) Many-objectives multilevel thresholding image segmentation using Knee Evolutionary Algorithm[J]. Expert Syst Appl 125:305–316
2. Zheng X, Ye H, Tang Y et al (2017) Image Bi-Level thresholding based on gray level-local variance histogram[J]. Entropy 19(5):191
3. Khairuzzaman AK, Chaudhury S (2017) Multilevel thresholding using grey wolf optimizer for image segmentation[J]. Expert Syst Appl 86:64–76
4. El Aziz MA, Ewees AA, Hassanien AE (2017) Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation[J]. Expert Syst Appl 83:242–256
5. Chakraborty R, Sushil R, Garg ML (2019) Hyper-spectral image segmentation using an improved PSO aided with multilevel fuzzy entropy[J]. Multimed Tools Appl 78(23):34027–34063
6. Hemeida AM, Mansour R, Hussein ME (2019) Multilevel thresholding for image segmentation using an improved electromagnetism optimization algorithm[J]. IJIMAI 5(4):102–112
7. Zhang S, Jiang W, Satoh S et al (2018) Multilevel thresholding color image segmentation using a modified artificial bee colony algorithm[J]. IEICE Trans Inf Syst E101.D:2064–2071
8. Erwin E, Saparudin S, Saputri W et al (2018) Hybrid multilevel thresholding and improved harmony search algorithm for segmentation[J]. Int J Electr Comput Eng 8(6):4593–4602

9. Upadhyay P, Chhabra JK (2019) Kapur's entropy based optimal multilevel image segmentation using crow search algorithm[J]. Appl Soft Comput 97:105522
10. Suresh S, Lal S (2016) An efficient cuckoo search algorithm based multilevel thresholding for segmentation of satellite images using different objective functions[J]. Expert Syst Appl 58:184–209
11. Xing Z (2020) An improved emperor penguin optimization based multilevel thresholding for color image segmentation[J]. Knowl-Based Syst 194:105570
12. Li K, Tan Z (2019) An improved flower pollination optimizer algorithm for multilevel image thresholding[J]. IEEE Access 7:165571–165582
13. Kotte S, Pullakura RK, Injeti SK (2018) Optimal multilevel thresholding selection for brain MRI image segmentation based on adaptive wind driven optimization[J]. Measurement 130:340–361
14. Bao X, Jia H, Lang C (2019) A novel hybrid harris hawks optimization for color image multilevel thresholding segmentation[J]. Ieee Access 7:76529–76546
15. Tan Z, Zhang D (2020) A fuzzy adaptive gravitational search algorithm for two-dimensional multilevel thresholding image segmentation[J]. J Ambient Intell Humaniz Comput 11:2–12
16. Bansal S (2019) A comparative study of nature-inspired metaheuristic algorithms in search of near-to-optimal Golomb rulers for the FWM crosstalk elimination in WDM systems[J]. Appl Artif Intell 33(14):1199–1265
17. Xiong L, Zhang D, Li K et al (2019) The extraction algorithm of color disease spot image based on Otsu and watershed[C]. Soft Comput 24:1–11
18. Xiong L, Chen RS, Zhou X et al (2019) Multi-feature fusion and selection method for an improved particle swarm optimization[J]. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-019-01624-4
19. Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems[J]. Eng Comput 29(1):17–35
20. Aziz MA, Hassanien AE (2018) Modified cuckoo search algorithm with rough sets for feature selection[J]. Neural Comput Appl 29(4):925–934
21. Thirugnanasambandam K, Prakash S, Subramanian V et al (2019) Reinforced cuckoo search algorithm-based multimodal optimization[J]. Appl Intell 49(6):2059–2083
22. Boushaki SI, Kamel N, Bendjeghaba O et al (2018) A new quantum chaotic cuckoo search algorithm for data clustering[J]. Expert Syst Appl 96:358–372
23. Zhang M, Wang H, Cui Z et al (2018) Hybrid multi-objective cuckoo search with dynamical local search[J]. Memetic Comput 10(2):199–208
24. Wang Z, Li Y (2015) Irreversibility analysis for optimization design of plate fin heat exchangers using a multi-objective cuckoo search algorithm[J]. Energy Convers Manag 101:126–135
25. Wang J, Zhou B, Zhou S (2016) An improved cuckoo search optimization algorithm for the problem of chaotic systems parameter estimation[J]. Comput Intell Neurosci 2016:8
26. Guerrero M, Castillo O, Garcia M (2015) Fuzzy dynamic parameters adaptation in the Cuckoo Search Algorithm using fuzzy logic[C]. In: 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, pp. 441–448. https://doi.org/10.1109/CEC.2015.7256923
27. Walton S, Hassan O, Morgan K et al (2011) Modified cuckoo search: a new gradient free optimisation algorithm[J]. Chaos Solitons Fractals 44(9):710–718
28. Wang G, Deb S, Gandomi AH et al (2016) Chaotic cuckoo search[C]. Soft Comput 20(9):3349–3362
29. Huang X, Shen L, Fan C, et al (2020) Multilevel image thresholding using a fully informed cuckoo search algorithm[J]. arXiv preprint arXiv: 2006.09987
30. Agrawal S, Samantaray L, Panda R et al (2020) A new hybrid adaptive cuckoo search-squirrel search algorithm for brain mr image analysis[m]//hybrid machine intelligence for medical image analysis. Springer, Singapore, pp 85–117
31. Joshi AS, Kulkarni O, Kakandikar GM et al (2017) Cuckoo search optimization-a review[J]. Mater Today Proc 4(8):7262–7269
32. Merzban MH, Elbayoumi M (2019) Efficient solution of Otsu multilevel image thresholding: a comparative study[J]. Expert Syst Appl 116:299–309
33. Manic KS, Priya RK, Rajinikanth V (2016) Image multithresholding based on Kapur/Tsallis entropy and firefly algorithm[J]. Indian J Sci Technol 9(12):89949
34. Zhang Y, Wu L (2011) Optimal multi-level thresholding based on maximum Tsallis entropy via an artificial bee colony approach[J]. Entropy 13(4):841–859
35. Vala HJ, Baxi A (2013) A review on Otsu image segmentation algorithm[J]. Int J Adv Res Comput Eng Technol (IJARCET) 2(2):387–389

36. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, pp. 210–214. https://doi.org/10.1109/NABIC.2009.5393690
37. Pare S, Kumar A, Bajaj V et al (2016) A multilevel color image segmentation technique based on cuckoo search algorithm and energy curve[J]. Appl Soft Comput 47:76–102
38. Agrawal S, Panda R, Bhuyan S et al (2013) Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm[J]. Swarm Evolut Comput 11:16–30
39. Nandy S, Yang X, Sarkar PP et al (2015) Color image segmentation by cuckoo search[J]. Intell Autom Soft Comput 21(4):673–685
40. Jia H, Lang C, Oliva D et al (2019) Hybrid grasshopper optimization algorithm and differential evolution for multilevel satellite image segmentation[J]. Remote Sens 11(9):1134
41. Bhandari AK (2018) A novel beta differential evolution algorithm-based fast multilevel thresholding for color image segmentation[J]. Neural Comput Appl 32:1–31
42. Garcia S, Molina D, Lozano M et al (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization[J]. J Heuristics 15(6):617–644
43. Bansal S (2020) Performance comparison of five metaheuristic nature-inspired algorithms to find near-OGRs for WDM systems[J]. Artif Intell Rev 53:1–47