Received December 17, 2019, accepted January 5, 2020, date of publication January 14, 2020, date of current version January 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2966665

Bee Foraging Algorithm Based Multi-Level Thresholding For Image Segmentation

ZHICHENG ZHANG AND JIANQIN YIN

School of Automation, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Zhicheng Zhang (zczhang@bupt.edu.cn)

This work was supported in part by the National Nature Science Foundation of China under Grant 61673192 and in part by the Fundamental Research Funds for The Central Universities under Grant 2019RC27.

ABSTRACT Multi-level thresholding is one of the essential approaches for image segmentation. Determining the optimal thresholds for multi-level thresholding needs exhaustive searching which is time-consuming. To improve the searching efficiency, a novel population based bee foraging algorithm (BFA) for multi-level thresholding is presented in this paper. The proposed algorithm provides different flying trajectories for different types of bees and takes both single-dimensional and multi-dimensional search aiming to maintain a proper balance between exploitation and exploration. The bee swarm is divided into a number of sub-swarms to enhance the diversity. A neighbourhood shrinking strategy is applied to mitigate stagnation and accelerate convergence. Experiments have been performed on eight benchmark images using between-class variance as the thresholding criterion. The performance of the proposed algorithm is compared with some state-of-art meta-heuristic algorithms. The results show that BFA is efficient and robust, produces excellent results with few control parameters, and outperforms other algorithms investigated in this consideration on most of the tested images.

INDEX TERMS Bee foraging algorithm, image segmentation, meta-heuristic, multi-level thresholding.

I. INTRODUCTION

Image segmentation is an essential technique for image processing, which is aiming to partition an image into a number of congeneric regions with similar characteristics using some pre-defined measurement criterions. Among various popular image segmentation methods, thresholding is one of the most efficient and easiest methods which are used commonly and extensively [1]. If a grayscale image is separated into two classes by one threshold value based on the histogram, the process is called bi-level thresholding. However, one threshold is insufficient to segment some complex images such as those containing multiple objects to be separated. The process that divides the pixels into multiple classes and separate the image into multiple regions, which is called multi-level thresholding, becomes necessary to choose at least two threshold values. Multi-level thresholding is helpful to partition the complex images more subtly. As the number of thresholds increases, the computation process for searching the optimal thresholds becomes time-consuming.

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Hao Chen.

For most of the measurement criterions of the threshold values, multi-level thresholding is considered as a multi-dimensional optimization problem which is inefficient for some traditional search methods to solve. Therefore, many meta-heuristic optimization algorithms are applied for improving the efficiency of multi-level thresholding in the past few years. The related work includes particle swarm optimization (PSO) [2]–[4], [6], [9], artificial bee colony (ABC) [2]–[4], [7], genetic algorithm (GA) [3], [4], [6], differential evolution (DE) [3], [6], [7], cuckoo search (CS) [3], [7], bat algorithm (BA) [4], social spider optimization (SSO) [5], [8], flower pollination algorithm (FPA) [5], state transition algorithm (STA) [6], bacterial foraging optimization (BFO) [7], [9], wind driven optimization (WDO) [7], whale optimization algorithm (WOA) [8], moth-flame optimization (MFO) [8], grey wolf optimizer (GWO) [9], etc.. Furthermore, some other modified meta-heuristic algorithms and methods are also proposed to solve multi-level thresholding problem for image segmentations [10]–[16]. All these meta-heuristic algorithms mentioned above have demonstrated their applicability for solving multi-level thresholding problems. However, since different optimization algorithms perform diversely in multi-level thresholding applications and no single algorithm is able to solve all the optimization problems properly [17], researchers have been spending significant efforts on improving the existing algorithms and investigating new meta-heuristic algorithms.

The algorithms which are inspired from the intelligent behaviours of honeybees are relatively new members of meta-heuristic algorithms. Researchers have concentrated on modelling various intelligent aspects of honeybee swarms over the past decade. A few algorithms have been proposed and proved efficient for solving various optimization problems [18]–[24]. However, some of these algorithms such as ABC is slow in convergence rate, while some other algorithm such as the bees algorithm (BA) is complicated to be used for its parameter tuning. In this work, a novel optimization algorithm based on the intelligent foraging behaviour of bee swarm is proposed and applied to improve the thresholding-based image segmentation. The proposed algorithm divides the bee swarm into two groups with three types of bees. The forager bees and onlooker bees constitute the recruit bees group to do the local search. They apply different types of updating strategies to search the neighbourhood of each selected food source thoroughly. While the scout bees search around the searching regions randomly at the end of each iteration, which play the role of global search. In addition, the proposed algorithm utilizes a set of approaches to prevent the premature and stagnation.

The proposed algorithm has some key features to improve the efficiency for multi-level threshold image segmentation:

- The recruit bees are divided into a number of sub-swarms to enhance the diversity of the bee swarm.
- Two rounds of searching are taken for each sub-swarm in an iteration. Foragers search the neighbourhood in full dimensions randomly while onlookers take single-dimensional search with a certain direction.
- Both the neighbourhood shrinking and food source abandonment procedures are applied to avoid the stagnation.

Considering the between-class variance as the objective function, the functionality of the proposed algorithm is discussed in comparison with ABC [18], MFO [25], GWO [26], and WOA [27] on eight benchmark images with different numbers of thresholds. The results show that the proposed mothed is relatively better than other tested algorithms.

The reminder of this paper is organized as follows. Section 2 outlines the Otsu's between-class variance criterion for multi-level thresholding; Section 3 introduces the intelligent foraging behaviour of honeybees and elaborates the proposed bee foraging algorithm (BFA); Section 4 presents the experimental results of the proposed algorithm compared with other algorithms. The influence of different neighbourhood shrinking rates for BFA is also discussed; Section 5 provides some discussions and conclusions.

Initialization:
  The n_s scout bees search the landscape randomly.
  Evaluate the food sources found by scout bees.
  Set upper limit of stagnation using equation (10).
  Choose $\frac{n_s}{2}$ better food sources as selected food sources based on fitness.
  Allocate foragers to each selected food source equally.
Repeat
  for each selected food source **do**
    Distribute foragers uniformly within the neighbourhood.
    if foragers find a better food source. **then**
      The better food source becomes to the new food source.
      Set stagnation accumulator to zero.
    else
      Accumulate the number of stagnations.
      Shrink the neighbourhood according to equation (7).
    end if
  end for
  Allocate onlookers to each selected food source based on equation (8).
  for each selected food source **do**
    Distribute onlookers according to equation (9).
    if onlookers find a better food source. **then**
      The better food source becomes to the new food source.
      Set stagnation accumulator to zero.
    else
      Accumulate the number of stagnations.
    end if
    if Stagnations $> Limit_{st}$ **then**
      Abandon the selected food source.
      Set its stagnation accumulator to zero.
      Set its neighbourhood to initial size.
    end if
  end for
  Scout bees search the space randomly.
  Rank the food sources based on fitness.
  Record the best food source so far.
  Choose $\frac{n_s}{2}$ better food sources as selected food sources for next iteration.
Until the stop criterion is met.

FIGURE 1. Pseudocode of bee foraging algorithm.

II. MULTI-LEVEL THRESHOLDING

For bi-level thresholding, the purpose is to find one optimal threshold value to separate an image into two parts. While the number of thresholds increased, the process becomes to a more complicated multi-level thresholding problem which is help to separate the image more precisely. Among various thresholding methods, Otsu's between-class variance method [28] is one of the most popular and efficient method which is considered as the thresholding criterion in this work.

The aim of multi-level thresholding is to find m $(m \geq 2)$ threshold values for image segmentation. Assuming that the given image I consists $m+1$ classes which are segmented by m optimal threshold values, as in the following equations:

$$\begin{aligned} M_0 &= \{g(x,y) \in I \mid 0 \leq g(x,y) \leq t_1 - 1\} \\ M_1 &= \{g(x,y) \in I \mid t_1 \leq g(x,y) \leq t_2 - 1\} \\ M_j &= \left\{g(x,y) \in I \mid t_j \leq g(x,y) \leq t_{j+1} - 1\right\} \\ &\cdots \\ M_m &= \{g(x,y) \in I \mid t_m \leq g(x,y) \leq L - 1\} \end{aligned} \tag{1}$$

where $t_j (j = 1, 2, \cdots, m)$ represents the j-th threshold value; M_j represents the j-th class of image I; $g(x, y)$ is the gray level value of pixel (x, y); L is the number of gray levels of image I. The gray levels are in the range $[0, 1, \cdots, L-1]$, then the probability of i-th gray level is defined as the following equation:

$$p_i = h(i)/N, \quad (i = 0, 1, \cdots, L-1) \tag{2}$$

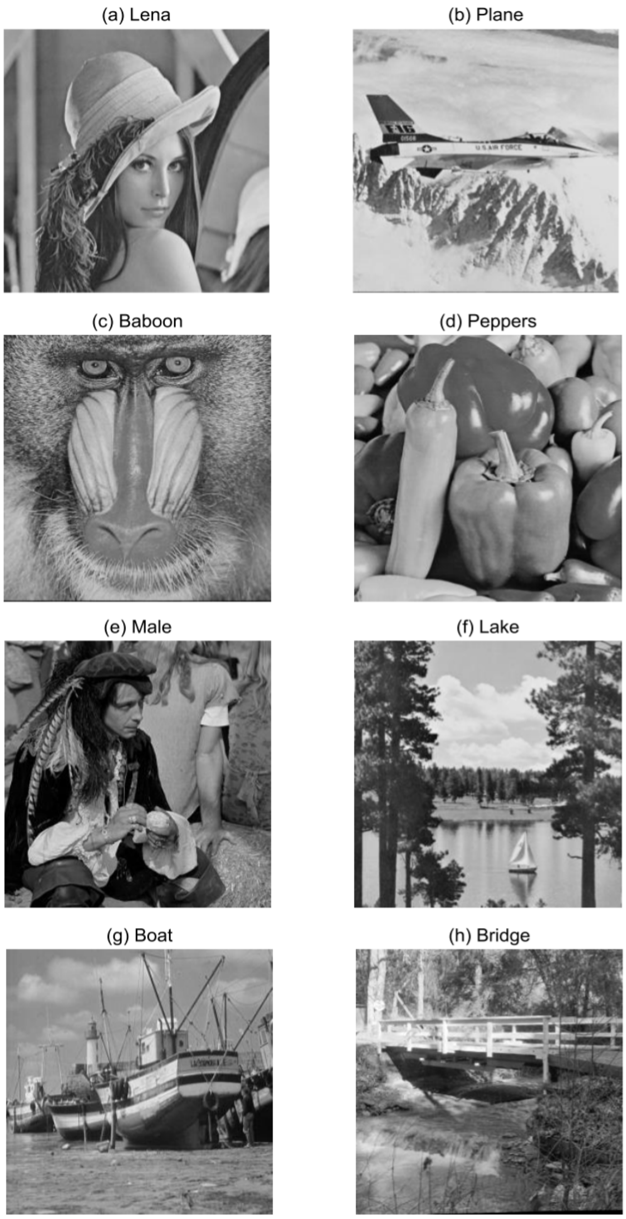


FIGURE 2. Benchmark Images tested in the experiments.

where $h(i)$ represents the number of pixels in the *i-th* gray level; N represents the total number of pixels in image I.

Define $t_0 = 0$ and $t_{m+1} = L$. The cumulative probability of each class for image I is calculated as follows:

$$\omega_j = \sum_{k=t_j}^{t_{j+1}-1} p_k, \quad (j = 0, 1, \cdots, m) \tag{3}$$

The mean grayscale level of each class is defined by the following equation:

$$\mu_j = \sum_{k=t_j}^{t_{j+1}-1} \frac{kp_k}{\omega_k}, \quad (j = 0, 1, \cdots, m) \tag{4}$$

Let μ_T be the total mean level of image I, which is calculated by the following equation:

$$\mu_T = \sum_{k=0}^{L-1} kp_k \tag{5}$$

Then the total between-class variance of the $m+1$ classes is defined as follows:

$$f(t) = \sum_{j=0}^{m} \omega_j(\mu_j - \mu_T)^2 \tag{6}$$

The optimal threshold values are obtained by maximizing the between-class variance presented in (6). Therefore, finding the multi-level thresholds becomes to solving an optimization problem, in which (6) is selected as the objective function to be optimized.

III. BEE FORAGING ALGORITHM

Bee Foraging Algorithm (BFA) is inspired by the foraging behaviour of honeybees in nature. In this section, the proposed BFA is presented and its biological motivation is outlined.

A. FORAGING BEHAVIOUR OF HONEYBEES

Honeybees are typical social insects living as swarms. A swarm of honeybees is able to perform complex tasks using relatively simple behaviour of individual bees. The foraging behaviour is one of the distinctive behaviours of honeybee swarm. This behaviour is the link between the bee swarm and the ambient environment, which helps bee swarm adapt to environmental changes quickly and search for food sources efficiently.

TABLE 1. Control parameters setting for the tested algorithms.

Algorithms	Control Parameters	
ABC	Swarm Size	20
	n_o	50% of the colony
	n_e	50% of the colony
	n_s	1
	Limit	$n_e \times$ Dimension
MFO	Swarm Size	20
	b	[-1,1]
	l	1
GWO	Swarm Size	20
WOA	Swarm Size	20
	a	[0,2]
	b	1
	l	[-1,1]
BFA	Swarm Size	20
	n_s	10% of the colony
	n_f	45% of the colony
	n_o	45% of the colony
	n_{sc}	0.7

TABLE 2. The mean values and standard deviations of the objective functions obtained by the tested algorithms.

Images	Nt	ABC		MFO		GWO		WOA		BFA	
		Mean	StdDev	Mean	StdDev	Mean	StdDev	Mean	StdDev	Mean	StdDev
Lena	2	1961.71	3.0E-01	1961.78	1.1E-01	1961.67	4.4E-01	1961.80	9.6E-02	**1961.82**	**7.5E-03**
	3	2128.05	4.1E-01	2128.20	2.1E-01	2127.52	1.5E-00	2128.17	7.1E-01	**2128.30**	**9.2E-03**
	4	2190.57	3.1E-01	2191.78	2.3E-01	2190.87	2.0E-00	2189.90	1.1E+01	**2191.87**	**7.1E-03**
	5	2217.52	3.1E-01	2217.05	7.3E-01	2216.30	3.0E-00	2215.00	7.3E-00	**2217.73**	**2.7E-01**
Plane	2	1948.66	1.1E-01	1948.70	7.7E-02	1948.63	1.6E-01	**1948.72**	2.1E-02	**1948.72**	**1.2E-02**
	3	2024,52	5.2E-01	2024.67	3.2E-01	2024.53	7.0E-01	2024.80	1.2E-01	**2024.82**	**1.8E-02**
	4	2069.46	7.3E-01	2069.69	7.0E-01	2069.02	1.8E-00	2069.62	1.5E-00	**2070.07**	**1.8E-02**
	5	2095.95	2.0E-01	2095.98	4.0E-01	2095.19	1.7E-00	2093.80	6.6E-00	**2096.13**	**2.1E-02**
Baboon	2	1549.07	3.0E-02	1549.06	6.1E-02	1548.91	4.2E-01	1548.94	6.8E-01	**1549.08**	**2.8E-02**
	3	1639.17	5.0E-01	1639.31	3.7E-01	1638.80	1.5E-00	1638.09	1.0E+01	**1639.53**	**1.6E-02**
	4	1692.68	4.7E-01	1692.82	1.5E-00	1692.45	1.2E-00	1689.78	1.1E+01	**1693.18**	**3.6E-02**
	5	1718.76	2.3E-01	1718.65	6.6E-01	1718.49	1.0E-00	1717.76	5.2E-00	**1719.02**	**4.9E-02**
Peppers	2	2532.39	5.7E-02	2532.30	5.5E-02	2532.21	4.0E-01	2532.28	2.6E-01	**2532.32**	**1.9E-02**
	3	2703.35	3.4E-01	2703.43	5.5E-01	2702.77	2.7E-00	2703.21	2.4E-00	**2703.57**	**7.4E-03**
	4	2765.53	1.0E-00	2765.58	1.4E-00	2765.69	1.4E-00	2765.94	1.3E-00	**2766.40**	**2.3E-01**
	5	2810.53	3.5E-01	2810.67	4.4E-01	2809.84	2.2E-00	2810.24	4.4E-00	**2810.84**	**9.8E-03**
Male	2	2997.59	3.3E-01	2997.64	5.3E-02	2997.61	2.8E-01	**2997.66**	4.8E-02	**2997.66**	**1.7E-02**
	3	3179.96	7.8E-01	3180.35	3.8E-01	3179.99	2.3E-00	3180.34	4.4E-01	**3180.51**	**3.3E-03**
	4	3265.37	5.4E-01	3265.67	6.9E-01	3265.18	2.5E-00	3265.88	1.4E-01	**3265.91**	**2.2E-02**
	5	3312.27	3.4E-01	3312.31	7.3E-01	3312.30	1.0E-00	3312.31	1.4E-00	**3312.62**	**2.7E-02**
Lake	2	3975.91	1.0E-01	3975.97	7.1E-02	3975.88	3.1E-01	**3976.00**	**5.9E-03**	3975.98	5.5E-02
	3	4113.49	4.2E-01	4113.63	2.9E-01	4113.19	1.5E-00	4113.63	3.1E-01	**4113.76**	**7.7E-03**
	4	4182.00	5.0E-01	4182.34	4.4E-01	4181.73	1.6E-00	4182.32	4.3E-01	**4182.51**	**2.0E-02**
	5	4218.20	2.5E-01	4218.05	1.4E-00	4217.27	2.8E-00	4217.28	6.1E-00	**4218.40**	**2.3E-02**
Boat	2	1863.31	8.3E-02	1863.33	5.4E-02	1863.24	3.2E-01	**1863.35**	**3.8E-03**	1863.34	1.6E-02
	3	1994.24	3.8E-01	1994.41	2.2E-01	1994.18	7.1E-01	1994.41	3.9E-01	**1994.52**	**2.5E-02**
	4	2059.41	5.1E-01	2059.63	4.7E-01	2058.75	2.7E-00	2058.94	5.4E-00	**2059.87**	**5.4E-03**
	5	2092.55	1.6E-01	2092.37	9.1E-01	2091.96	1.6E-00	2091.69	5.3E-00	**2092.76**	**2.1E-02**
Bridge	2	2532.46	6.0E-02	2532.46	6.0E-02	2532.34	5.4E-01	2532.07	1.2E-00	**2532.46**	**4.1E-12**
	3	2721.93	7.1E-01	2722.24	4.1E-01	2722.00	1.5E-00	2721.66	1.7E-00	**2722.34**	**6.5E-02**
	4	2821.97	9.5E-01	2822.19	1.2E-00	2821.86	2.1E-00	2819.37	9.4E-00	**2822.58**	**2.6E-01**
	5	2873.97	4.0E-01	2873.56	1.1E-00	2873.38	1.9E-00	2871.16	6.0E-00	**2874.14**	**2.4E-01**

A bee swarm is made up of different types of honey bees such as (employed or unemployed) foragers, onlookers, and scout bees, etc. [29]. Different types of bees play different roles in their foraging process. The Scout bees, in general, occupy about 10% of the total honeybee population when they need to find some new food sources [30]. They spontaneously explore the fields surrounding the hive looking for new food sources rich in nectar or pollen. When they return to the hive, scout bees share the information of the food sources they found with forager bees through a special action named 'waggle dance' to recruit foragers exploiting the food sources [30]. The waggle dance takes place in a particular space in the hive called 'dance floor', and conveys three items of information for each food source including the quality of the food source, the direction of the food source, and distance between the food source and the hive. Forager bees select several rich food sources according to the information from scout bees to collect nectar or pollen. Once a forager returns to the hive, it brings back the nectar and reports the information of the food source through waggle dance to the onlooker bees. Onlookers estimate the profitability of each food source based on the information conveyed by the foragers. They choose more profitable food sources to do the exploitation using a probabilistic approach, since more profitable food sources may provide more valuable resources. Hence, more profitable food sources recruit more bees to collect nectar.

The intelligent behaviour of bee swarm makes the foraging process more efficient and dynamic in a collective intelligent manner. The proposed algorithm in this paper takes inspiration from the foraging strategy of bee swarm to search for the best solution for a given optimization problem.

B. THE PROPOSED ALGORITHM

The bee swarm in BFA is divided into two groups: scout bees and recruit bees. Scout bees explore the searching space randomly for new food sources, while recruit bees wait in the hive for making decisions to choose food sources according to the information from scout bees. The most profitable food

TABLE 3. The mean values and standard deviations of the objective functions obtained by BFA with different neighbourhood shrinking rates.

Images	Nt	n_{sc} =0.9		n_{sc} =0.8		n_{sc} =0.7		n_{sc} =0.6		n_{sc} =0.5	
		Mean	StdDev	Mean	StdDev	Mean	StdDev	Mean	StdDev	Mean	StdDev
Lena	2	1961.69	1.6E-01	1961.81	4.4E-02	**1961.82**	**7.5E-03**	1961.81	2.5E-02	1961.81	3.5E-02
	3	2128.10	2.2E-01	**2128.30**	2.3E-02	**2128.30**	**9.2E-03**	**2128.30**	3.9E-02	2128.30	1.3E-02
	4	2191.85	4.0E-02	**2191.87**	**3.7E-03**	**2191.87**	7.1E-03	**2191.87**	1.7E-02	2191.86	2.5E-02
	5	2217.39	6.0E-01	2217.67	3.6E-01	2217.73	2.7E-01	**2217.75**	**8.9E-02**	2217.74	2.1E-01
Plane	2	1948.63	1.5E-01	1948.71	4.9E-02	**1948.72**	1.2E-02	1948.71	4.0E-02	**1948.72**	**2.4E-03**
	3	2024.72	9.0E-02	**2024.82**	3.3E-02	**2024.82**	1.8E-02	**2024.82**	**1.7E-02**	**2024.82**	2.8E-02
	4	2070.03	2.0E-01	**2070.07**	**1.3E-02**	**2070.07**	1.8E-02	2070.06	4.1E-02	2070.04	8.1E-02
	5	2096.13	1.2E-02	2096.12	2.2E-02	**2096.13**	**2.1E-02**	2096.11	4.6E-02	2096.09	5.4E-02
Baboon	2	1548.94	2.8E-01	1549.07	5.7E-02	**1549.08**	**2.8E-02**	1549.07	6.9E-02	1549.07	5.2E-02
	3	1639.36	1.6E-01	**1639.53**	3.7E-02	**1639.53**	**1.6E-02**	**1639.53**	3.4E-02	1639.52	4.1E-02
	4	1693.16	7.9E-02	**1693.18**	**2.8E-02**	**1693.18**	3.5E-02	1693.17	3.8E-02	1693.17	3.3E-02
	5	**1719.03**	3.3E-02	**1719.03**	**2.9E-02**	1719.02	4.9E-02	1719.00	7.7E-02	1718.98	9.2E-02
Peppers	2	2532.20	1.9E-01	2532.30	5.2E-02	**2532.32**	1.9E-02	**2532.32**	**9.1E-03**	**2532.32**	1.8E-02
	3	2703.41	1.8E-01	2703.56	3.4E-02	**2703.57**	**7.1E-03**	**2703.57**	1.1E-02	**2703.57**	7.4E-03
	4	2766.29	6.4E-01	2766.38	4.7E-01	**2766.40**	**2.0E-01**	2766.29	7.3E-01	2766.39	2.2E-01
	5	**2810.84**	9.8E-03	**2810.84**	1.1E-02	**2810.84**	**9.0E-03**	2810.83	2.0E-02	2810.82	3.0E-02
Male	2	2997.52	2.3E-01	2997.65	3.7E-02	**2997.66**	1.7E-02	2997.65	7.1E-02	**2997.66**	**6.6E-03**
	3	3180.32	2.0E-01	**3180.51**	2.7E-02	**3180.51**	**3.3E-03**	**3180.51**	1.0E-02	**3180.51**	2.4E-02
	4	3265.89	4.1E-02	**3265.91**	2.6E-02	**3265.91**	**2.2E-02**	3265.90	4.3E-02	3265.89	4.7E-02
	5	**3312.63**	**1.5E-02**	**3312.63**	2.1E-02	3312.62	2.7E-02	3312.62	2.9E-02	3312.60	4.5E-02
Lake	2	3975.85	1.7E-01	3975.98	2.9E-02	3975.98	5.5E-02	**3975.99**	**2.7E-02**	3975.98	3.0E-02
	3	4113.60	1.7E-01	4113.75	6.1E-02	**4113.76**	**7.7E-03**	**4113.76**	1.6E-02	**4113.76**	2.2E-02
	4	4182.48	5.6E-02	**4182.51**	2.6E-02	**4182.51**	**2.0E-02**	4182.50	3.2E-02	4182.48	4.6E-02
	5	**4218.41**	1.4E-02	**4218.41**	**1.3E-02**	4218.40	2.3E-02	4218.40	2.7E-02	4218.37	5.8E-02
Boat	2	1863.23	1.7E-01	1863.33	7.5E-02	**1863.34**	**1.6E-02**	**1863.34**	**1.6E-02**	**1863.34**	3.0E-02
	3	1994.38	1.7E-01	**1994.53**	**2.0E-02**	**1994.53**	2.2E-02	**1994.53**	2.3E-02	**1994.53**	2.3E-02
	4	2059.85	2.7E-02	2059.86	7.1E-03	**2059.87**	**5.4E-03**	2059.86	8.5E-03	2059.86	2.1E-02
	5	**2092.76**	**1.5E-02**	**2092.76**	1.8E-02	**2092.76**	2.1E-02	2092.75	2.9E-02	2092.73	3.9E-02
Bridge	2	2532.45	8.4E-02	**2532.46**	4.3E-02	**2532.46**	**4.1E-12**	**2532.46**	6.0E-02	2532.43	2.9E-01
	3	2722.25	3.0E-01	2722.31	1.6E-01	**2722.34**	**6.5E-02**	2722.32	1.7E-01	2722.30	2.2E-01
	4	2822.50	3.0E-01	2822.50	3.9E-01	**2822.58**	**2.6E-01**	2822.55	2.8E-01	2822.46	3.4E-01
	5	2874.13	2.7E-01	2874.14	2.6E-01	**2874.14**	**2.4E-01**	2874.08	2.9E-01	2874.04	3.4E-01

sources attract the largest number of recruit bees to exploit for the resources (nectar or pollen). The recruit bees group contains two types of bees: foragers and onlookers. They apply different strategies to select the food sources with higher quality, and use diverse tactics to update their positions in the searching space. The searching space where recruit bees exploit for better resources around the food sources is called 'neighbourhood'. If the number of bees recruited by a food source is confirmed, the size of the neighbourhood determines the searching intensity within the neighbourhood. Shrinking the size of the neighbourhood could help the bees finding the profitable food sources rapidly [23]. In BFA, a neighbourhood shrinking procedure is tuned and applied to increase the efficiency for foragers searching around a food source. The propose algorithm can be divided into three stages: initialization phase, local search phase, and global search phase.

In initialization phase, a number of scout bees search the landscape randomly with a uniform distribution. Then these bees go back to the hive to recruit more bees for the local search. The food sources reported by scout bees are ranked according to their profitability. The half of the food sources with better nectar resources become to the selected food sources, around which recruit bees carry out the local search. After the initialization phase, the algorithm enters the main loop which composed of local search phase and global search phase.

In local search phase, the recruit bees are divided into sub-swarms according to the number of selected food sources. Each selected food source recruits a group of forager bees and a group of onlooker bees to exploit its surrounding fields for better nectar resources. The forager neighbourhood search is applied firstly. Foragers, the first half of the recruit bees, are allocated equally to different selected food sources, and distributed uniformly within the neighbourhood around each selected food source. If a forager finds a better nectar surrounding the selected food source it belongs to, the better nectar replaces the selected food source. Then the foragers

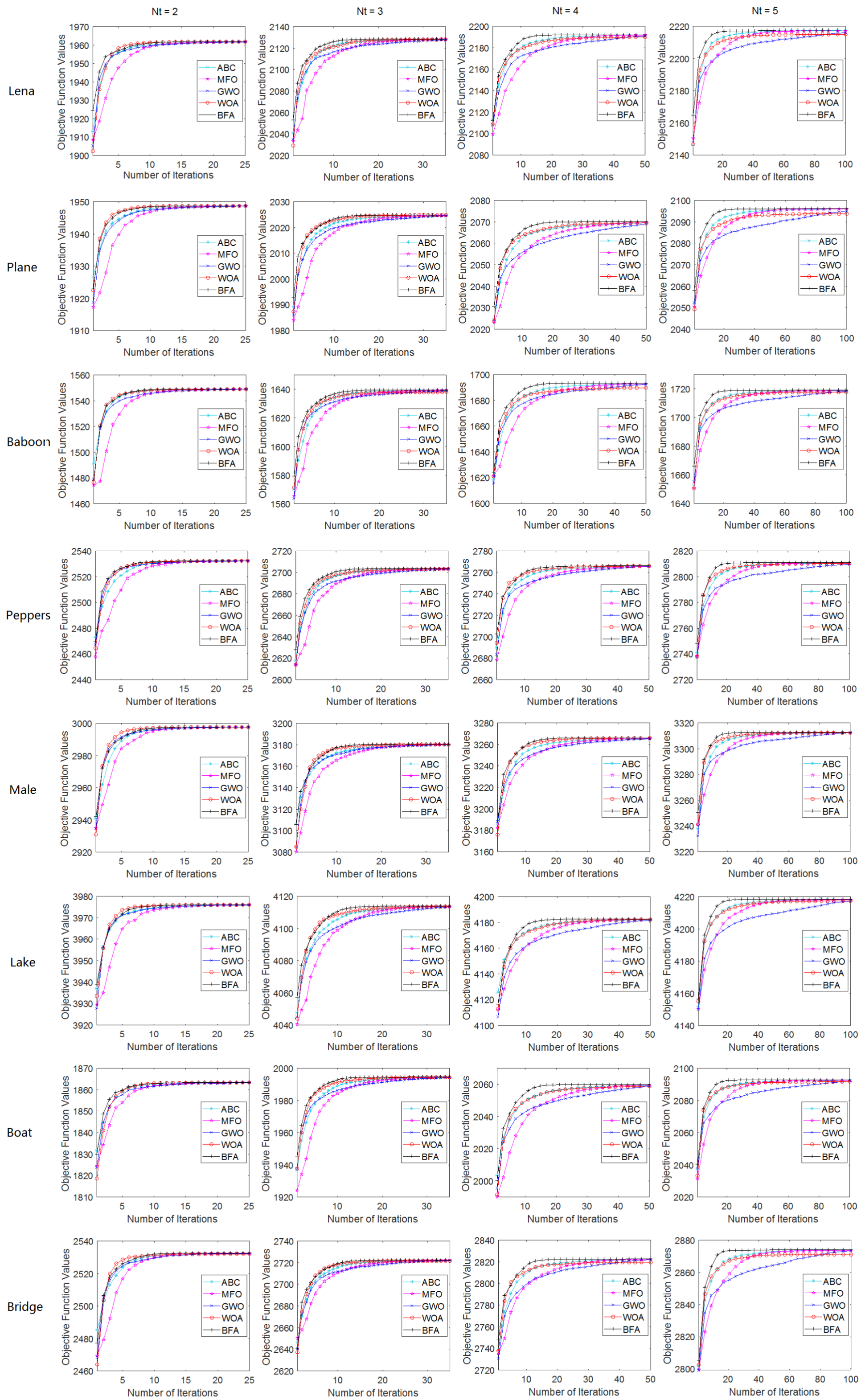


FIGURE 3. **Convergence curves of objective functions for multi-level thresholding obtained by the tested algorithms.**

allocated to it will search around the new selected food source in the next iteration. If foragers fail to improve some of the selected food sources, the unimproved food sources are recorded as stagnations. Furthermore, the neighbourhood shrinking procedure is applied to the unimproved food sources to increase the intensity of neighbourhood search for foragers. The updating of the neighbourhood size is described as the following equation:

$$S_{nh}(n+1) = n_{sc} \cdot S_{nh}(n) \tag{7}$$

where $S_{nh}(n)$ is the current size of the neighbourhood; $S_{nh}(n + 1)$ is the neighourhood size in the next iteration; n_{sc} is the neighbourhood shrinking coefficient.

After the forager neighbourhood search, all the selected food sources are sorted according to the quality of their nectar. Then the onlooker neighbourhood search, which use the other half of the recruit bees, is implemented. The onlooker search is a further local search according to the results of the forager search. The number of onlookers allocated to each selected food source in their neighbourhood search depends on the quality of each selected food source. That means the better selected food sources recruit more onlookers to do the neighbourhood search and have more chance to exploit their nectar, while the worse selected food sources recruit less onlookers and have less chance to exploit their nectar. In addition, it is necessary to make sure that each selected food source recruits at least one onlooker bee to keep its chance being searched for better nectar. A selection technique based on fitness function is used for allocating onlookers to each selected food source using the following equation:

$$N_j = fix\left[\frac{fitness_j}{\sum_{k=1}^{K} fitness_k} \cdot (n_o - K)\right] + 1 \tag{8}$$

where N_j is the number of onlookers recruited by the *j-th* selected food source; $fitness_j$ is the fitness function value of *j-th* selected food source; n_o is total number of onlookers; K is the number of selected food sources which equals to the half of scout bee numbers; *fix* represents taking an integer. According to (8), more onlookers are recruited to search around the selected food sources with higher fitness values. As the local search develops, some of the searching dimensions may achieve the position near optimum while others are not. At this moment, update the position of recruit bees in one dimension may help the algorithm to find optimum more quickly. Therefore, a different update strategy from forager search is applied in onlooker search. After choosing the selected food sources, onlookers collect neighbourhood resources using the equation as below:

$$W_i^d(n) = Z_j^d(n) + \alpha\left[Z_j^d(n) - Z_r^d(n)\right] \tag{9}$$

where d is a randomly selected dimension; $W_i(n)$ is the updated position of the *i-th* onlooker bee allocated to the *j-th* selected food source; $Z_\mathrm{j}(n)$ is the current position of the *j-th* selected food source; α is a random number within the range (-1,1); $Z_r(n)$ is the current position of a randomly choosed food source with the condition $j \neq r$. Similar to forager neighbourhood search, if onlookers find a more profitable food source, the new food source becomes to the selected food source instead of the current one for the next iteration. If a food source is not improved, its number of stagnations is accumulated.

The issue should be further discussed is the neighbourhood shrinking procedure, which is applied to make forager neighbourhood search more efficient. As local search progresses, the neighbourhood may shrink to a quite small size if there is no criterion to stop shrinking. Furthermore, if a selected food source hasn't been improved for a certain number of iterations, this food source may fall into a local optimum and should be abandoned to mitigate stagnation. In BFA, the stagnation number of a selected food source accumulates in both forager search and onlooker search, hence this parameter is used for estimating the stagnation of a selected food source and determine which food source should be abandoned. The limitation number of stagnations for a food source is obtained by the following equation:

$$Limit_{st} = 2 \cdot fix\left[\frac{\log(0.1)}{\log(n_{sc})}\right] \tag{10}$$

where $Limit_{st}$ represents the upper limit of stagnations for a selected food source without improvement. If the stagnation number is larger than $Limit_{st}$, the corresponding selected food source is ranked to the bottom of all the food sources and will be abandoned in global search phase.

In global search phase, the worse half of food sources other than the selected food sources are abandoned. Scout bees always explore the whole landscape randomly. Then the new nectar sources found by scout bees will be ranked together with the local search results. As the selected food sources in current iteration and the new food sources obtained from random search by scout bees are all reserved, the number of food sources after global search becomes to three times as many selected food sources. Then the best one third of all these food sources becomes to the selected food sources for the next iteration. The pseudocode of the proposed BFA is presented in Figure (1).

For SI based algorithms, it is important to establish a proper balance between exploitation and exploration in the swarm. It is also important to minimize the control parameters to be tuned and make the algorithm easy to be applied to solve engineering optimization problems. In BFA, different types of bees use different strategies to search for better nectars, and should be allocated with proper ratio of population to keep the searching balance. Therefore, we take 10% of bee swarm population as scout bees for global search and 90% of the population as recruit bees for local search. Forager search and onlooker search are different kinds of neighbourhood search with equal importance in local search, so recruit bees are divided into two parts equally. That means foragers and onlookers occupy 45% of the total population respectively. This kind of population allocation is inspired by some biological research and hypothesis of bee swarms in nature [30], which helps to reduce the number of control parameters for the algorithm.

TABLE 4. The optimum PSNR (dB) and SSIM values obtained by the tested algorithms.

Images	Nt	ABC		MFO		GWO		WOA		BFA	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Lena	2	20.3791	0.24481	20.4031	0.24485	20.3969	0.24471	20.3990	0.24480	**20.4085**	**0.24489**
	3	23.7475	0.31414	23.7615	0.31423	23.7296	0.31406	23.7649	0.31418	**23.7666**	**0.31426**
	4	25.3245	0.37346	25.3341	0.37465	25.3282	0.37454	25.3163	0.37315	**25.3390**	**0.37474**
	5	26.1386	0.42998	26.0890	0.42352	26.1133	0.42737	26.0791	0.42668	**26.1552**	**0.43168**
Plane	2	22.2995	0.28152	22.3125	0.28174	22.2938	0.28125	22.3292	0.28184	**22.3375**	**0.28201**
	3	22.9710	0.34353	22.9564	0.34357	22.9144	0.34307	22.9556	0.34364	**23.0121**	**0.34402**
	4	24.1483	0.40416	24.1453	0.40398	24.1069	0.40382	24.2009	0.40453	**24.2147**	**0.40492**
	5	25.3174	0.43353	25.3359	0.43458	25.3294	0.43357	25.2805	0.43361	**25.3863**	**0.43554**
Baboon	2	20.1202	0.57192	20.1223	0.57195	20.1151	0.57184	20.1119	0.57183	**20.1185**	**0.57203**
	3	21.6434	0.66948	21.6483	0.66960	21.6524	0.66955	21.6537	0.66897	**21.6722**	**0.67108**
	4	23.0513	0.72836	23.0435	0.72820	23.0368	0.72752	22.9932	0.72511	**23.0542**	**0.72839**
	5	24.2292	0.78132	24.1957	0.77998	24.2239	0.78113	24.2428	0.78044	**24.2828**	**0.78300**
Peppers	2	20.1035	0.22232	20.1167	0.22234	20.0962	0.22232	**20.1196**	**0.22236**	20.1189	0.22235
	3	22.6052	0.30382	22.6075	0.30385	22.5901	0.30371	22.6047	0.30381	**22.6169**	**0.30398**
	4	23.6554	0.36646	23.6414	0.36414	23.6447	0.36463	23.6509	0.36512	**23.6643**	**0.36793**
	5	25.1498	0.42382	25.1473	0.42375	25.1385	0.42378	25.1477	0.42348	**25.1659**	**0.42451**
Male	2	18.8459	0.38116	18.8528	0.38122	18.8493	0.38119	**18.8687**	**0.38132**	18.8603	0.38127
	3	21.3297	0.49386	21.3091	0.49359	21.2939	0.49342	21.3039	0.49405	**21.3338**	**0.49462**
	4	24.9487	0.56368	24.9122	0.56106	24.9308	0.56334	24.9486	0.56362	**24.9546**	**0.56371**
	5	26.7973	0.62634	26.7943	0.62629	26.7865	0.62528	26.8167	0.62707	**26.8232**	**0.62742**
Lake	2	21.5653	0.35703	21.6040	0.35715	21.5758	0.35723	**21.6188**	**0.35726**	21.6095	0.35724
	3	23.3771	0.41241	23.3763	0.41311	23.3727	0.41323	23.3867	0.41347	**23.3961**	**0.41397**
	4	24.8692	0.47317	24.8665	0.47329	24.8313	0.47332	24.8582	0.47323	**24.8762**	**0.47347**
	5	26.0517	0.53785	26.0233	0.53777	25.9724	0.53646	26.0424	0.53642	**26.0748**	**0.53839**
Boat	2	19.7766	0.34191	19.7741	0.34201	19.7759	0.34128	**19.7798**	**0.34289**	19.7774	0.34272
	3	23.5891	0.42725	23.5939	0.42690	23.5781	0.42716	23.5830	0.42719	**23.5968**	**0.42745**
	4	25.9992	0.55679	25.9702	0.55746	25.9657	0.55621	25.9627	0.55655	**26.0019**	**0.55799**
	5	27.4879	0.59918	27.5132	0.60149	27.5047	0.59989	27.4650	0.59994	**27.5407**	**0.60116**
Bridge	2	20.2639	0.48436	20.2589	0.48357	20.2629	0.48431	20.2582	0.48359	**20.2645**	**0.48449**
	3	22.8919	0.60661	22.8918	0.60663	22.5503	0.60631	22.8892	0.60639	**22.8968**	**0.60682**
	4	25.1489	0.69359	25.1462	0.69355	25.1253	0.69256	25.1146	0.69159	**25.1553**	**0.69366**
	5	26.7636	0.75312	26.7464	0.75211	26.7398	0.75147	26.6925	0.75004	**26.7669**	**0.75314**

IV. EXPERIMENTS AND RESULTS

In this section, the experiments and results for evaluating the performance of the proposed BFA applied on multi-level thresholding segmentation for a number of benchmark images are presented. The threshold results obtained by BFA are compared with the results using some state-of-art swarm intelligent optimization algorithms including ABC, MFO, GWO, and WOA. In addition, some discussions are also made in this section.

A. EXPERIMENTAL SETTINGS

To test the performance of BFA for multi-level image segmentation, a set of well-known benchmark images are used here for evaluation and comparison. The set contains 8 grayscale images including 'Lena', 'Plane', 'Baboon', 'Peppers', 'Male', 'Lake', 'Boat', and 'Bridge'. These images are taken from USC-SIPI which are widely applied to evaluate image segmentation methods in the literature. All the images have the same size of 512×512. The tested images are presented in Figure (2).

There are some control parameters for each optimization algorithm to adjust the algorithm suitable for solving specific optimization problem. Common control parameters which all the algorithms needed are population size (colony/swarm size) and the number of maximum generations. The population size is set to 20 in all experiments in this section, while the number of maximum generations is changed according to the increasing of thresholds. The maximum generations are set to 25, 35, 50, 100 respectively as the numbers of thresholds are 2, 3, 4, 5. In addition to common control parameters, there are some diverse parameters to be set for some of the algorithms tested in this section. The control parameters settings for the five algorithms in the experiments are given in Table (1). All the experiments are implemented using Matlab R2018a in a computer running on Windows 7 system with 3.4 GHz Intel core-i7 CPU and 16 GB RAM.

For statistical performance comparison, 100 times of independent experiments are taken for each algorithm finding a certain number of optimal thresholds for one tested image.

TABLE 5. The threshold values obtained by the tested algorithms.

Images	Nt	ABC	MFO	GWO	WOA	BFA
Lena	2	93, 151	93, 151	94, 151	92, 151	93, 151
	3	80, 126, 171	80, 126, 171	81, 126, 170	80, 126, 171	81, 127, 171
	4	74, 113, 145, 180	75, 114, 145, 180	75, 113, 145, 180	73, 112, 144, 180	75, 114, 145, 180
	5	73, 108, 136, 159, 187	70, 102, 130, 155, 185	71, 105, 132, 157, 186	65, 102, 130, 156, 186	73, 109, 136, 160, 188
Airplane	2	113, 173	113, 173	113, 173	113, 173	113, 173
	3	93, 145, 191	93, 145, 191	93, 146, 191	93, 145, 191	93, 145, 191
	4	84, 129, 172, 203	83, 128, 172, 203	84, 128, 172, 203	83, 129, 172, 203	84, 129, 172, 203
	5	69, 107, 142, 179, 205	70, 107, 143, 180, 205	70, 108, 143, 180, 205	65, 106, 142, 179, 205	69, 107, 143, 180, 205
Baboon	2	98, 150	98, 150	97, 150	98, 150	98, 150
	3	85, 125, 161	85, 125, 161	85, 125, 161	84, 124, 161	86, 125, 161
	4	72, 106, 137, 168	72, 107, 137, 168	72, 106, 137, 167	68, 105, 137, 168	72, 106, 137, 168
	5	67, 99, 125, 149, 175	67, 98, 124, 149, 174	68, 99, 125, 149, 174	64, 98, 124, 149, 174	68, 100, 126, 150, 175
Peppers	2	68, 135	68, 135	68, 135	68, 135	68, 135
	3	63, 118, 166	63, 118, 166	63, 118, 165	62, 119, 166	63, 119, 166
	4	47, 87, 126, 169	49, 88, 127, 170	48, 87, 126, 169	47, 86, 126, 169	47, 86, 126, 169
	5	43, 79, 113, 146, 177	43, 79, 113, 146, 177	43, 78, 113, 146, 177	42, 78, 112, 145, 177	43, 79, 113, 146, 177
Male	2	57, 125	58, 125	57, 125	58, 125	58, 125
	3	39, 91, 142	38, 91, 142	38, 92, 142	39, 91, 142	38, 91, 142
	4	35, 82, 125, 166	34, 82, 124, 165	34, 82, 124, 166	34, 82, 125, 166	35, 82, 125, 166
	5	28, 65, 100, 135, 173	28, 66, 101, 135, 173	28, 65, 100, 135, 173	27, 64, 99, 134, 173	28, 66, 100, 134, 173
Lake	2	86, 155	86, 155	86, 155	86, 155	86, 155
	3	78, 140, 195	78, 140, 195	78, 140, 195	79, 140, 195	79, 141, 195
	4	67, 110, 158, 199	67, 110, 158, 199	67, 110, 157, 199	67, 111, 158, 199	67, 110, 158, 199
	5	58, 89, 128, 167, 201	58, 88, 128, 167, 201	58, 89, 129, 167, 200	56, 88, 128, 167, 201	58, 89, 128, 167, 201
Boat	2	93, 155	93, 155	93, 155	93, 155	93, 155
	3	73, 126, 167	73, 126, 167	73, 127, 167	73, 126, 167	73, 126, 167
	4	65, 113, 147, 179	65, 114, 147, 179	65, 113, 146, 179	65, 113, 146, 179	65, 114, 147, 179
	5	52, 92, 128, 153, 184	53, 94, 129, 154, 185	53, 94, 129, 154, 185	50, 91, 127, 153, 184	52, 93, 128, 153, 184
Bridge	2	91, 157	91, 156	91, 157	92, 157	91, 157
	3	74, 123, 180	74, 123, 180	74, 123, 180	74, 123, 179	75, 123, 180
	4	62, 102, 143, 193	62, 102, 143, 193	63, 103, 144, 193	60, 102, 143, 192	62, 103, 144, 193
	5	55, 89, 121, 158, 200	55, 88, 120, 157, 200	55, 89, 121, 157, 200	52, 87, 120, 156, 199	55, 89, 121, 158, 201

The mean values of the results are used for evaluating and comparing the algorithms.

B. EXPLOITATION PERFORMANCE

Exploitation performance is one of the most important indicators for an optimization algorithm, which indicates the ability for finding the optimum solutions. In this section, the exploitation performance of each tested algorithm is evaluated and compared with others in terms of optimum values for the objective functions. The values of mean optimum and standard deviations of the objective functions, which are optimized respectively by ABC, MFO, GWO, WOA, and BFA, are given in Table (2). The best results obtained by the algorithms are shown in bold in order to make it clear to be observed and analyzed.

Considering the searching accuracy of the algorithms which indicated by the mean values of the objective functions. The results show that the proposed BFA finds the best results in 30 cases out of the total 32 cases, which outperforms other four algorithms. While WOA gets the best results in 4 cases only in less number of thresholds. For standard deviation which stands for searching stability, BFA also gets best results in 30 cases while WOA gets 2 best results. It is obvious from the experiment results that BFA performs better in exploitation than other four algorithms under the same number of iterations.

Another issue should be assessed for the proposed BFA is the value of neighbourhood shrinking coefficient (n_{sc}) which is used for improving the exploitation performance by increasing the density of the bees within a certain neighbourhood. As the value of n_{sc} may influence both the exploitation performance and the convergence speed of the algorithm, it is essential to select a proper n_{sc} value for a specific application. The mean values and standard deviations of the objective functions for the tested images obtained by BFA with different n_{sc} values are given in Table (3). The results show that BFA with $n_{sc} = 0.7$ gets the best results in 27 cases. While the algorithm with $n_{sc} = 0.8$ and $n_{sc} = 0.6$ find 16 and 13 best results in the experiment respectively. It is clear that 0.7 is a decent value for n_{sc}, which means the neighbourhood shrinking in a medium rate helping BFA perform better in multi-level threshoding for image segmentation.

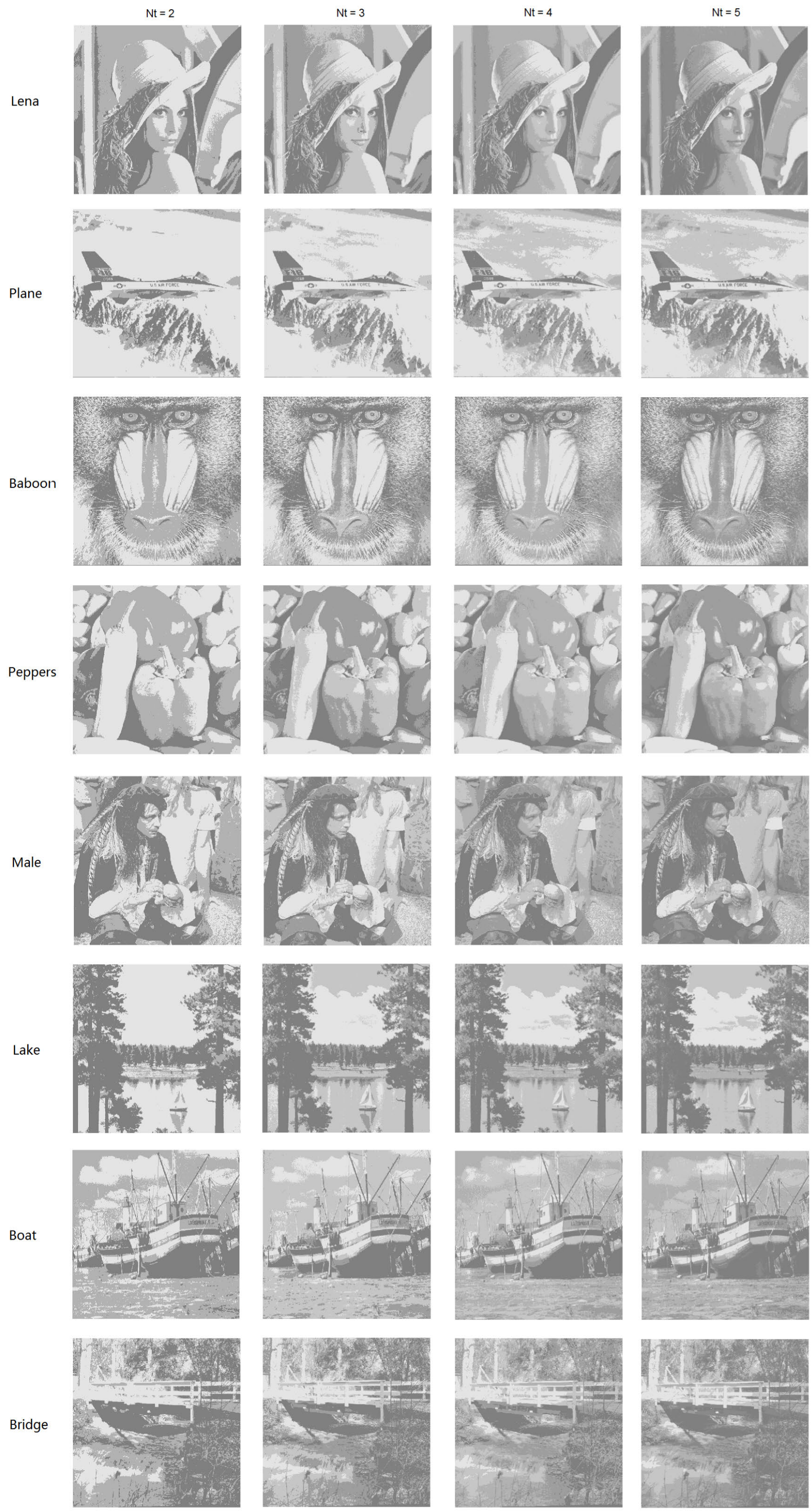


FIGURE 4. Segmented images with different threshold levels obtained by the proposed algorithm.

C. CONVERGENCE PERFORMANCE

Convergence performance is another essential index which represents the efficiency of the algorithms. To further evaluate the proposed algorithm, the convergence performance on different threshold levels are also studied in this section. Figure (3) shows the convergence curves of objective functions obtained by the tested algorithms for image segmentation.

The convergence speed is similar for all the tested algorithms when the optimization process is implemented in lower-levels thresholding. However, as the number of thresholds increases, the advantage of BFA on convergence performance becomes obvious. BFA converges to optimal thresholds faster than other tested algorithms for most of the cases when the number of thresholds is larger than three. That means the proposed algorithm finds the optimum thresholds using less number of iterations and lower computational load than other tested algorithms in higher-levels thresholding.

D. QUALITY MEASUREMENT

To determine the segmentation quality of the benchmark images, two most common used metrics in image quality evaluations, which are peak signal to noise ratio (PSNR) and structural similarity index (SSIM) [31], are selected as the criterion to compare the segmentation results obtained by different algorithms. The definition of PSNR is presented as follows:

$$PSNR = 20 \log_{10} \left(\frac{255}{RMSE} \right) \tag{11}$$

where *RMSE* represents the root mean squared error between the original image I and the segmented image $\tilde{I}$ in size of $X \times Y$. *RMSE* is calculated as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{X} \sum_{j=1}^{Y} \left[I(i,j) - \tilde{I}(i,j) \right]^2}{X \cdot Y}} \tag{12}$$

SSIM is another metric for evaluating the quality of images. It is designed to improve some traditional metrics such as PSNR by considering human visual perception. SSIM is defined as follows:

$$SSIM(I, \tilde{I}) = \frac{\left(2\mu_I \mu_{\tilde{I}} + c_1\right)\left(2\sigma_{I,\tilde{I}} + c_2\right)}{\left(\mu_I^2 + \mu_{\tilde{I}}^2 + c_1\right)\left(\sigma_I^2 + \sigma_{\tilde{I}}^2 + c_2\right)} \tag{13}$$

where μ_I and $\mu_{\tilde{I}}$ are the mean intensity of I and $\tilde{I}$ images respectively; σ_I and $\sigma_{\tilde{I}}$ represent the standard deviation of image I and image $\tilde{I}$ respectively. c_1 and c_2 are two constants used for enhancing the stability when $\mu_I^2 + \mu_{\tilde{I}}^2$ is too close to zero.

Table (4) gives the PSNR and SSIM values for the proposed algorithm and other tested algorithms respectively. While the number of the thresholds increases, the PSNR and SSIM values become larger for all the tested algorithms. As the higher value indicates the better performance for both PSNR and SSIM metrics. It can be seen from Table (4) that the proposed algorithm gets better results in most cases. BFA segments the images with better quality in all experiment cases as the number of threshold levels is larger than three. WOA gets three images of better segmentation quality from the 8 benchmark images with 2 thresholds. These two algorithms occupy all the better results in the experiment.

Table (5) gives the threshold values obtained by all the tested algorithms. While Figure (4) shows the segmentation results of the BFA based method with different threshold levels. It is clear from the images that the images segmented with higher threshold levels contain more details than the ones with lower threshold levels.

V. CONCLUSION

In this work, a novel bee foraging algorithm (BFA) based multi-level thresholding method is presented for image segmentation. BFA is a meta-heuristic optimization algorithm which is inspired from the foraging behaviour of honeybees. The bee swarm is allocated to two groups with fixed proportion for three types of bees so as to reduce the number of control parameters and improve the applicability of the algorithm. The recruit bees are divided into a number of sub-swarms for local search, while the scout bees search around the whole fields for global search. Different kinds of update strategies are implemented by different types of bees in different stages of the optimization process aiming to maintain the proper balance between exploration and exploitation. Neighbourhood shrinking procedure is also introduced into BFA in order to avoid stagnation and improve convergence. The influence of different neighbourhood shrinking coefficient values on the performance of the algorithm is discussed. The functionality of the proposed algorithm is assessed based on eight benchmark images with different number of thresholds. Setting the value of neighbourhood shrinking coefficient to 0.7 seems suitable for most of the involved cases to get acceptable optimization performance. The performance of the proposed algorithm is also compared with four other popular optimization algorithms. The experiment results show that BFA is an efficient and powerful algorithm for multi-level thresholding and outperforms the other four algorithms on most cases of experiments investigated in this paper.

REFERENCES

[1] K. G. Dhal, A. Das, S. Ray, J. Gálvez, and S. Das, "Nature-inspired optimization algorithms and their application in multi-thresholding image segmentation," *Arch. Comput. Methods Eng.*, vol. 1, no. 1, pp. 1–34, Mar. 2019.

[2] B. Akay, "A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding," *Appl. Soft Comput.*, vol. 13, no. 6, pp. 3066–3091, Jun. 2013.

[3] T. Kurban, P. Civicioglu, R. Kurban, and E. Besdok, "Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding," *Appl. Soft Comput.*, vol. 23, pp. 128–143, Oct. 2014.

[4] Z.-W. Ye, M.-W. Wang, W. Liu, and S.-B. Chen, "Fuzzy entropy based optimal thresholding using bat algorithm," *Appl. Soft Comput.*, vol. 31, pp. 381–395, Jun. 2015.

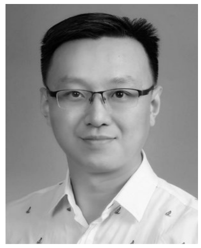
[5] S. Ouadfel and A. Taleb-Ahmed, "Social spiders optimization and flower pollination algorithm for multilevel image thresholding: A performance study," *Expert Syst. Appl.*, vol. 55, pp. 566–584, Aug. 2016.

[6] J. Han, C. Yang, X. Zhou, and W. Gui, "A new multi-threshold image segmentation approach using state transition algorithm," *Appl. Math. Model.*, vol. 44, pp. 588–601, Apr. 2017.

[7] S. Pare, A. Kumar, V. Bajaj, and G. Singh, "An efficient method for multilevel color image thresholding using cuckoo search algorithm based on minimum cross entropy," *Appl. Soft Comput.*, vol. 61, pp. 570–592, Dec. 2017.

[8] M. A. E. Aziz, A. A. Ewees, and A. E. Hassanien, "Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation," *Expert Syst. Appl.*, vol. 83, pp. 242–256, Oct. 2017.

[9] A. K. M. Khairuzzaman and S. Chaudhury, "Multilevel thresholding using grey wolf optimizer for image segmentation," *Expert Syst. Appl.*, vol. 86, pp. 64–76, Nov. 2017.
[10] K. Tang, X. Xiao, J. Wu, J. Yang, and L. Luo, "An improved multilevel thresholding approach based modified bacterial foraging optimization," *Appl. Intell.*, vol. 46, no. 1, pp. 214–226, Jan. 2017.
[11] Y. Jiang, P. Tsai, W.-C. Yeh, and L. Cao, "A honey-bee-mating based algorithm for multilevel image segmentation using Bayesian theorem," *Appl. Soft Comput.*, vol. 52, pp. 1181–1190, Mar. 2017.
[12] L. Shen, C. Fan, and X. Huang, "Multi-level image thresholding using modified flower pollination algorithm," *IEEE Access*, vol. 6, pp. 30508–30519, 2018.
[13] H. Gao, Z. Fu, C.-M. Pun, H. Hu, and R. Lan, "A multi-level thresholding image segmentation based on an improved artificial bee colony algorithm," *Comput. Electr. Eng.*, vol. 70, pp. 931–938, Aug. 2018.
[14] S. Pare, A. Bhandari, A. Kumar, and G. Singh, "A new technique for multilevel color image thresholding based on modified fuzzy entropy and Lévy flight firefly algorithm," *Comput. Electr. Eng.*, vol. 70, pp. 476–495, Aug. 2018.
[15] H. Liang, H. Jia, Z. Xing, J. Ma, and X. Peng, "Modified grasshopper algorithm-based multilevel thresholding for color image segmentation," *IEEE Access*, vol. 7, pp. 11258–11295, 2019.
[16] H. Jia, J. Ma, and W. Song, "Multilevel thresholding segmentation for color image using modified moth-flame optimization," *IEEE Access*, vol. 7, pp. 44097–44134, 2019.
[17] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[18] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Oct. 2007.
[19] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 687–697, Jan. 2008.
[20] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: Artificial bee colony (ABC) algorithm and applications," *Artif. Intell. Rev.*, vol. 42, no. 1, pp. 21–57, Jun. 2014.
[21] X. S. Yang, "Engineering optimizations via nature-inspired virtual bee algorithms," in *Proc. Int. Work-Conf. Interplay Between Natural Artif. Comput.*, Canary Islands, Spain. Berlin, Germany: Springer, Jun. 2005, pp. 317–323.
[22] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm—A novel tool for complex optimisation problems," in *Proc. 2nd Virtual Inter. Conf. Intell. Prod. Mach. Syst.*, Jul. 2006, pp. 454–459.
[23] D. T. Pham and M. Castellani, "The bees algorithm: Modelling foraging behaviour to solve continuous optimization problems," *Proc. Inst. Mech. Eng. C, J. Mech. Eng. Sci.*, vol. 223, no. 12, pp. 2919–2938, Jul. 2009.
[24] D. T. Pham and M. Castellani, "Benchmarking and comparison of nature-inspired population-based continuous optimisation algorithms," *Soft Comput.*, vol. 18, no. 5, pp. 871–903, May 2014.
[25] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[26] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[27] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 46–61, Mar. 2016.
[28] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.
[29] V. Tereshko and A. Loengarov, "Collective decision-making in honey bee foraging dynamics," *J. Comput. Inf. Syst.*, vol. 9, no. 3, pp. 1–7, 2005.
[30] T. D. Seeley, *The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies*. Cambridge, MA, USA, Harvard Univ. Press, 1995.
[31] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.



ZHICHENG ZHANG received the B.S. degree in information engineering, the M.S. degree in pattern recognition, and the Ph.D. degree in control theory and engineering from Jilin University, Changchun, China, in 2005, 2007, and 2011, respectively. From 2011 to 2018, he was a Teacher with the Department of Control Science and Engineering, Jilin University, where he had been an Associate Professor, since 2015. Since 2018, he has been an Associate Professor with the Department of Measurement and Control Technology, Beijing University of Posts and Telecommunications, Beijing, China. His research interests include signal processing, artificial intelligence, and computational intelligence.

JIANQIN YIN received the M.S. and Ph.D. degrees in pattern recognition and intelligent systems from Shandong University, Jinan, China. From 2003 to 2016, she worked as a Teacher with the University of Jinan, Jinan. Since 2016, she has been a Teacher with the Beijing University of Posts and Telecommunications, Beijing, China. Her research interests include pattern recognition, machine learning, and image processing.