Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 974024, 23 pages
http://dx.doi.org/10.1155/2014/974024

Research Article

A Global Multilevel Thresholding Using Differential Evolution Approach

Kanjana Charansiriphaisan, Sirapat Chiewchanwattana, and Khamron Sunat

Department of Computer Science, Faculty of Science, Khon Kaen University, Khon Kaen 40002, Thailand

Correspondence should be addressed to Khamron Sunat; khamron_sunat@yahoo.com

Received 3 October 2013; Revised 23 January 2014; Accepted 3 February 2014; Published 20 March 2014

Academic Editor: Yi-Hung Liu

Otsu's function measures the properness of threshold values in multilevel image thresholding. Optimal threshold values are necessary for some applications and a global search algorithm is required. Differential evolution (DE) is an algorithm that has been used successfully for solving this problem. Because the difficulty of a problem grows exponentially when the number of thresholds increases, the ordinary DE fails when the number of thresholds is greater than 12. An improved DE, using a new mutation strategy, is proposed to overcome this problem. Experiments were conducted on 20 real images and the number of thresholds varied from 2 to 16. Existing global optimization algorithms were compared with the proposed algorithms, that is, DE, rank-DE, artificial bee colony (ABC), particle swarm optimization (PSO), DPSO, and FODPSO. The experimental results show that the proposed algorithm not only achieves a more successful rate but also yields a lower threshold value distortion than its competitors in the search for optimal threshold values, especially when the number of thresholds is large.

1. Introduction

Thresholding is the simplest and most commonly used method of image segmentation. It can be bilevel or multilevel [1]. Both of these types can be classified into parametric and nonparametric approaches [1]. Surveys of thresholding techniques for image segmentation can be found in [2–7]. The surveys revealed that Otsu's method is a commonly used technique [4, 8]. This method finds the optimal thresholds by maximizing the weighted sum of between-class variances (BCV) [9]. The BCV function is also called Otsu's function. However, the solution finding process is an exhaustive search and it is a very time-consuming process because the complexity grows exponentially with the number of thresholds.

Multilevel image thresholding based on Otsu's function has been used as a benchmark for comparing the capability of evolutionary algorithms (EA). The EA is a nongradient based optimization algorithm. Several algorithms have been widely applied to solve multilevel thresholding. A group of successful works were based on a combination of Otsu's function with some state-of-the-art algorithms: PSO [10], DE [11], ABC [12], and FOSPSO [13]. Kulkarni and Venayagamoorthy [14] showed that PSO was faster than Otsu's method in searching the optimal thresholds of multilevel image thresholding. Akay [15] presented a comprehensive comparative study of the ABC and PSO algorithms. The results showed that the ABC algorithm with both the between-class variance and the entropy criterion can be efficiently used in multilevel thresholding. Hammouche et al. [16] focused on solving the image thresholding problem by combining Otsu's function with metaheuristic techniques, that is, genetic algorithm (GA), PSO, DE, ant colony, simulated annealing, and Tabu search. Their results revealed that DE was the most efficient with respect to the quality of solution. Osuna-Enciso et al. [17] presented an empirical comparative study of the ABC, PSO, and DE algorithms to perform image thresholding using a mixture of Gaussian functions. The results showed that the DE algorithm was superior in performance in minimizing the Hellinger distance and used less evaluations of the Hellinger distance. Ghamisia et al. [18] showed that a global optimal search for optimal threshold values of Otsu's function was essential for the multilevel segmentation of multispectral and hyperspectral images.

The DE algorithm was selected for multilevel image thresholding. It is simple to implement and produces good results. However, based on our experiments, DE could not reach an optimal solution when it was applied to a very difficult problem. Therefore, a better DE algorithm is required. We noticed that the mechanism of vector selection and the size of the higher ranked population are an important criterion for success.

The contribution of this paper is as follows.

DE with the onlooker and ranking-based mutation operation, named $O(\beta)R$-DE, is proposed to overcome the drawback of the DE algorithm for multilevel image thresholding, especially when the number of thresholds is large. The proposed algorithm homogenizes the onlooker phase of the ABC algorithm and the ranking-based mutation operator of the rank-DE [19]. The main advantage of the proposed algorithm is that a user can adjust the balancing of the exploitation and exploration capabilities of the algorithm.

To verify the capabilities of the proposed $O(\beta)R$-DE algorithm, experiments to find the optimal solutions in the multilevel image thresholding, when the number of thresholds ranged from two to 16, were set up. It was found that the optimal solutions could be effectively reached using the proposed $O(\beta)R$-DE algorithm.

The remainder of the paper is organized as follows. Section 2 describes the multilevel thresholding problem. Section 3 presents a brief review of the differential evolution algorithm (DE). In Section 4, the proposed new version of the DE algorithm with the onlooker and ranking-based mutation operator algorithm, $O(\beta)R$-DE, is described in detail. Section 5 shows the experimental results of applying the proposed method to multilevel segmentation in different images. Finally, the conclusion of the paper is discussed in Section 6.

2. Multilevel Thresholding Problem Formulation

Otsu's method [9] is based on the maximization of the between-class variance. Consider a digital image having the size $H \times W$, where W is the width and H is the height. The pixels of a given picture are represented in L gray levels and they are in $\{0, 1, 2, \ldots, L-1\}$. The number of pixels at level i is denoted by n_i and the total number of pixels by $N = n_1 + n_2 + \cdots + n_L$. The gray-level histogram is normalized and regarded as a probability distribution and is written as follows:

$$p_i = \frac{n_i}{N}, \quad p_i \geq 0, \quad \sum_{i=1}^{L} p_i = 1. \tag{1}$$

The total mean of the image can be defined as

$$\mu_T = \sum_{i=1}^{L} i \times p_i. \tag{2}$$

The multilevel thresholding with respect to the given $n-1$ threshold values t_j, $j = 1, \ldots, n-1$ can be performed as follows:

$$F(x,y) = \begin{cases} 0, & f(x,y) \leq t_1, \\ \frac{1}{2}(t_1 + t_2), & t_1 < f(x,y) \leq t_2, \\ \vdots & \\ \frac{1}{2}(t_{n-2} + t_{n-1}), & t_{n-2} < f(x,y) \leq t_{n-1}, \\ L, & f(x,y) > t_{n-1}, \end{cases} \tag{3}$$

where (x, y) is the coordinate of a pixel and $f(x, y)$ denotes the intensity level of a pixel. The pixels of a given image will be divided into n classes $D_1, \ldots, D_n$ in this regard.

The optimal threshold can be determined by maximizing the between-class variance function (BCV), σ_B^2, which can be defined by

$$\sigma_B^2 = \sum_{j=1}^{n} w_j \left(\mu_j - \mu_T\right)^2, \tag{4}$$

where j represents a specific class in such a way that w_j and μ_j are the probability of occurrence and the mean of class j, respectively. Equation (4) is also called Otsu's function. The probabilities of occurrence w_j of classes $D_1, \ldots, D_n$ are defined by

$$w_j = \begin{cases} \sum_{i=1}^{t_j} p_i, & j = 1, \\ \sum_{i=t_{j-1}+1}^{t_j} p_i, & 1 < j < n, \\ \sum_{i=t_{j-1}+1}^{L} p_i, & j = n. \end{cases} \tag{5}$$

The mean of each class μ_j can be given by

$$\mu_j = \begin{cases} \sum_{i=1}^{t_j} \frac{i \times p_i}{w_j}, & j = 1, \\ \sum_{i=t_{j-1}+1}^{t_j} \frac{i \times p_i}{w_j}, & 1 < j < n, \\ \sum_{i=t_{j-1}+1}^{L} \frac{i \times p_i}{w_j}, & j = n. \end{cases} \tag{6}$$

Thus, the n-level thresholding problem is transformed to an optimization problem. The process is to search for $n-1$ thresholds t_j that maximize the value φ, which is generally defined as

$$\varphi = \max_{1 < t_1 < \cdots < t_{n-1} < L} \sigma_B^2 (t_j). \tag{7}$$

3. Differential Evolution Algorithm

The DE algorithm is an evolutionary optimization technique proposed by Storn and Price [11]. The main procedures of DE are briefly described as follows.

3.1. Initialization. The DE algorithm starts with a population of initial solutions, each of dimension D, $X_{i,g} = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$, $i = 1, \ldots, \mathrm{NP}$, where the index i denotes the ith solution, or vector, of the population, g is the generation, and NP is the population size. The initial population (at $g = 0$) is randomly generated to be within the search space constrained by the minimum and maximum bounds, $X_{\min} = \{x_{1,\min}, x_{2,\min}, \ldots, x_{D,\min}\}$ and $X_{\max} = \{x_{1,\max}, x_{2,\max}, \ldots, x_{D,\max}\}$. The ith vector x_i is initialized as follows:

$$x_{j,i,0} = x_{j,\min} + \mathrm{rndreal}_{i,j}[0,1) \cdot \left(x_{j,\max} - x_{j,\min}\right), \quad (8)$$

where $\mathrm{rndreal}_{i,j}[0,1)$ is a uniformly distributed random real number between 0 and 1, $(0 \leq \mathrm{rndreal}_{i,j}[0,1) < 1)$.

3.2. Mutation Operators. The differential mutation operator is one of the three operators of DE. The mutation operator is applied to generate the mutant vector v_i for each target vector x_i in the current population. A mutant vector is generated according to

$$v_{i,g+1} = x_{r_1,g} + F \cdot \left(x_{r_2,g} - x_{r_3,g}\right), \quad (9)$$

where the randomly chosen indexes, random indexes, $r_1, r_2, r_3 \in \{1, 2, \ldots, \mathrm{NP}\}$ are mutually different random integer indices and they are also different from the running index i. Further, i, r_1, r_2, and r_3 are different so that $\mathrm{NP} \geq 4$. F is a real and constant factor, $F \in [0, 2]$, which controls the amplification of the differential variation; $x_{r_1,g}$ is called the base vector, $x_{r_2,g}$ is called the terminal vector, $x_{r_3,g}$ is called the other vector, and $(x_{r_2,g} - x_{r_3,g})$ is called the difference vector.

There have been many proposed mutation strategies for DE [20, 21]. Each different strategy has different characteristics and is suitable for a set of problems. However, the choice of the best mutation operators for DE is difficult for a specific problem [22–24]. The "DE/rand/1/bin" strategy has been widely used in DE literature [25–28]. It is more reliable than the strategies based on the best-so-far solution such as "DE/best/1" and "DE/current-to-best/1". However, "DE/rand/1/bin" has slower convergence. Simply put, it has high exploration but low exploitation abilities.

3.3. Crossover. DE utilizes the crossover operation to generate new solutions by shuffling competing vectors and to increase the diversity of the population. The classical version of the DE (DE/rand/1/bin) uses the binary crossover. It defines the following trial vector:

$$u_{i,g+1} = \left(u_{1i,g+1}, u_{2i,g+1}, \ldots, u_{Di,g+1}\right), \quad (10)$$

where $j = 1, \ldots, D$ (D = problem dimension) and

$$u_{ji,g+1} = \begin{cases} v_{ji,g+1} & \text{if } (\mathrm{randb}(j) \leq \mathrm{CR}) \text{ and } j = \mathrm{rnbr}(i) \\ x_{ji,g} & \text{if } (\mathrm{randb}(j) > \mathrm{CR}) \text{ and } j \neq \mathrm{rnbr}(i). \end{cases} \quad (11)$$

CR is the crossover rate $\in [0,1]$, $\mathrm{randb}(j)$ is the jth evaluation of a uniform random number generator with outcome $\in [0,1]$, and $\mathrm{rnbr}(i)$ is a randomly chosen index $\in 1, 2, \ldots, D$ that ensures $u_{i,g+1}$ will get at least one parameter from $v_{i,g+1}$.

3.4. Selection. Selection determines whether the target or the trial vector survives to the next generation. The selection operation is described as

$$x_{i,g+1} = \begin{cases} u_{i,g}, & \text{if } f\left(u_{i,g}\right) \leq f\left(x_{i,g}\right) \\ x_{i,g}, & \text{if } f\left(u_{i,g}\right) > f\left(x_{i,g}\right), \end{cases} \quad (12)$$

where $f(x)$ is the objective function to be minimized. Therefore, if the objective of the new trial vector, $f(u_{i,g})$, is equal to or less than the objective of the old trial vector, $f(x_{i,g})$, then $x_{i,g+1}$ is set to $u_{i,g}$; otherwise, the old value $x_{i,g}$ is retained.

The pseudocode of basic DE with "DE/rand/1/bin" strategy is shown in Algorithm 1.

The function $\mathrm{rndint}[1, D]$ returns a uniformly distributed random integer number between 1 and D. $\mathrm{rndreal}_j[0,1)$ is a uniformly distributed random real value of $[0,1)$. The word "better" in line 17 means "less than" if the problem requires minimization, see (12) and its explanation, and it means "greater than," if the problem requires maximization. The best $X_{i,G}$, where G is the maximum number of generations, is the solution of the algorithm. The word "best" also depends on the type of problem.

4. The Proposed DE with Onlooker Ranking-Based Mutation Operator

In 2013 Gong and Cai [19] proposed a rank-DE algorithm. They claimed that probabilistically selecting the vectors x_{r_1} and x_{r_2} in the mutation operator from the better population can improve the exploitation ability of basic DE. To the best of the authors' knowledge, rank-DE may, however, also lead to premature convergence (this will be shown in the experiments). That means that the rank-DE has too much exploitation ability. Furthermore, it cannot balance between the exploration and the exploitation abilities. In order to balance between the two abilities, we propose DE with the onlooker and ranking-based mutation operator, named $O(\beta)R$-DE. The proposed algorithm is an improvement of the rank-DE by homogenizing the rank-DE with the onlooker phase of ABC algorithm. The detail of the $O(\beta)R$-DE algorithm is described as follows.

4.1. Ranking Assignment. To perform the maximization, the fitness of each vector is sorted in ascending order (i.e., from worst to best). Then, the rank of the ith vector, R_i, is assigned based on its sorted ordering as follows:

$$R_{\mathrm{order}} = \mathrm{order}, \quad \mathrm{order} = 1, 2, \ldots, \mathrm{NP}. \quad (13)$$

As a result, the best vector in the current population will obtain the highest ranking, that is, NP.

Generate the initial population randomly
Evaluate the fitness for each individual in the population
while the maximum generation G is not reached do
  for $i =$ 1 to NP do
    Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq i$
    $j_{\text{rand}} = \text{rndint}\ [1, D]$
    for $j = 1$ to D do
      if $\text{rndreal}_j[0,1) \leq \text{CR}$ or j is equal to j_{rand} then
        $u_{i,j} = x_{r_1,j} + F \cdot \left(x_{r_2,j} - x_{r_3,j}\right)$
      else
        $u_{i,j} = x_{i,j}$
      end if
    end for
  end for
  for $i = 1$ to NP do
    Evaluate the offspring u_i
    if $f(u_i)$ is better than or equal to $f(x_i)$ then
      Replace x_i with u_i
    end if
  end for
end while

Algorithm 1: The DE algorithm with "DE/rand/1/bin" strategy.

4.2. Probabilistic Selection. After assigning the ranking for each vector, the selection probability p_i of the ith vector x_i is calculated as

$$p_i = \frac{R_i}{\text{NP}}, \quad i = 1, 2, \ldots, \text{NP}. \tag{14}$$

4.3. A New Strategy for Base Vector, Terminal Point, and the Other Vector Selections

Definition 1 (a worse population and a better population). Let ζ be a real value and $0 \leq \zeta < 1$. A population having probability less than ζ is called a worse population and a population having probability greater than or equal to ζ is called a better population.

In the rank-DE, the base vector x_{r_1} and the terminal point x_{r_2} were based on their selection probabilities. The other vector in the mutation operator, x_{r_3}, is selected randomly as in the original DE algorithm. The vectors with higher rankings (higher selection probabilities) are more likely to be chosen as the base vector or the terminal point in the mutation operator.

Our investigation revealed that if both x_{r_1} and x_{r_2} vectors of rank-DE were chosen from better vectors, then the distribution of the target vector may collapse quickly and possibly lead to premature convergence. Accordingly, when the rank-DE was applied to a very difficult problem, it could not reach the optimal solution.

If the steps of the DE algorithm are compared with the ABC algorithm, the population in the current generation can be considered as the employed bees and the population in the next generation can be considered as the onlooker bees. To follow the concept of ABC, a new vector, x_{r_1}, which is called the base vector, chooses a food source with respect to the probability that is computed from the fitness values of the current population. The probability value, p_t, of which x_t is chosen by a base vector x_{r_1} can be calculated by using the expression given in (14). After a base source x_{r_1} for a new vector is probabilistically chosen, both x_{r_2} and x_{r_3} are also chosen in the same manner as the terminal point and the other vector selections in the rank-DE. The target vector is created by a mutation formula of DE. The mutant vector u_i is created after the target vector is crossed with a randomly selected vector, and then the fitness value is computed. As in the ordinary DE, a greedy selection is applied between u_i and x_i. Hence, the new population contains better sources and positive feedback behavior appears. This idea can be expressed as pseudocode, as in Algorithm 2. Since the selection of x_{r_1} is the onlooker selection and the selections of x_{r_2} and x_{r_3} are brought from the rank-DE, then the algorithm is called onlooker and ranking-based vector selection.

The pseudocode of onlooker and ranking-based vector selection is shown in Algorithm 2. The differences between the original ranking-based and onlooker and ranking based selection are highlighted by "$\Leftarrow$".

The function minprop(β) is added to generalize the algorithm. Its output depends on the parameter β. The outcome can be either a constant value of $[0, 1)$ or a value of the uniform random function rndreal$[0, 1)$. The balance of the exploration and exploitation ability can be set by the parameter β. And the function is defined by

$$\text{minprop}(\beta) = \begin{cases} \beta, & \text{if } \beta \text{ is a constant and } 0 \leq \beta < 1 \\ \text{rndreal}[0,1), & \text{otherwise.} \end{cases} \tag{15}$$

Input: The target vector index i, the last index of onlooker r_1, and β ⇐
Output: The selected vector indexes r_1, r_2, r_3
$r_1 = r_1 + 1$; if $r_1 > \mathrm{NP}$ then $r_1 = 1$; end if ⇐
while minprop(β) $>$ p_{r_1} //onlooker-like selection ⇐
    $r_1 = r_1 + 1$; if $r_1 > \mathrm{NP}$ then $r_1 = 1$; end if ⇐
end while
Randomly select $r_2 \in \{1, \mathrm{NP}\}$ //terminal vector index
while rndreal$[0, 1) > p_{r_2}$ or $r_2 == i$ or $r_2 == r_1$ do
    Randomly select $r_2 \in \{1, \mathrm{NP}\}$
end while
Randomly select $r_3 \in \{1, \mathrm{NP}\}$ //the other vector index
while $r_3 == r_2$ or $r_3 == r_1$ or $r_3 == i$ do
    Randomly select $r_3 \in \{1, \mathrm{NP}\}$
end while

Algorithm 2: Onlooker and ranking-based vector selection for DE.

Randomly generate the initial population
Evaluate the fitness for each individual in the population
while the maximum generation G is not reached do
    Sort and rank the fitness values of population according to (13)
    Calculate the selection probability for each individual according to (14)
    $r_1 = 0$ ⇐
    for $i = 1$ to NP do
        Select r_1, r_2, r_3 as shown in **Algorithm** 2 based on the current r_1 and β ⇐
        $j_{\text{rand}} = \text{rndint}[1, D]$
        for $j = 1$ to D do
            if rndreal$_j[0, 1) \leq \mathrm{CR}$ or j is equal to j_{rand} then
                $u_{i,j} = x_{r_1,j} + F \cdot \left(x_{r_2,j} - x_{r_3,j}\right)$
            else
                $u_{i,j} = x_{i,j}$
            end if
        end for
    end for
    for $i = 1$ to NP do
        Evaluate the offspring u_i
        if $f(u_i)$ is better than or equal to $f(x_i)$ then
            Replace x_i with u_i
        end if
    end for
end while

Algorithm 3: DE with onlooker and ranking-based mutation.

4.4. The DE with Onlooker-Ranking-Based Mutation Operator. The procedures in Sections 4.1, 4.2, and 4.3 are combined together to create a better DE algorithm. The parameter $0 \leq \beta < 1$ determines the fraction of the worse population to be eliminated. When $\beta = 0$ there is no worse population; each single vector in the current population will act as the base vector. If $0 < \beta < 1$, then each single vector having a probability less than β is a worse vector and will not be selected as the base vector. If β is not a constant or is outside $[0, 1)$, each single base vector is an onlooker bee. Accordingly, the name of the algorithm is Onlooker(β) Ranking-Base Differential Evolution ($O(\beta)R$-DE). To achieve the global solution, a user can set a proper value for β to control the balance of the exploration and exploitation abilities of the algorithm. The pseudocode of $O(\beta)R$-DE is shown in Algorithm 3 and the differences between the rank-DE and $O(\beta)R$-DE are highlighted by "⇐".

5. Experiments and Results

5.1. Experimental Setup. The global multilevel thresholding problem deals with finding optimal thresholds within the range $[0, L - 1]$ that maximize the BCV function. The dimension of the optimization problem is the number of thresholds, n, and the search space is $[0, L-1]^n$. The parameter

β of $O(\beta)R$-DE is rndreal$[0,1)$ or is set to be one of 0.0, $0.1,\ldots,0.9$. The variation of the proposed $O(\beta)R$-DE was implemented and compared with the existing metaheuristics that performed image thresholding, that is, PSO, DPSO, FODPSO, ABC, and several variations of DE algorithms. All the methods were programmed in Matlab R2013a and were run on a personal computer with a 3.4 GHz CPU, 8 GB RAM with Microsoft Windows 7 64-bit operating system. The experiments were conducted on 20 real images. The 19 images, namely, starfish, mountain, cactus, butterfly, circus, snow, palace, flower, wherry, waterfall, bird, police, ostrich, viaduct, fish, houses, mushroom, snow mountain, and snake, were taken from the Berkeley Segmentation Dataset and Benchmark [29]. The last image, namely, Riosanpablo, is a satellite image "New ISS Eyes see Rio San Pablo", March 1, 2013 (http://visibleearth.nasa.gov/view.php?id=80561). Each image has a unique gray level histogram. These original images and their histograms are depictedin Figure 1. An experiment of an image with a specific number of thresholds is called a "*subproblem.*" The number of thresholds investigated in the experiments was $2,3,\ldots,16$. Thus, there are 20×15 subproblems per algorithm. Each subproblem was repeated 50 times and each time is called a run.

To compare with PSO, ABC, and DEs algorithms, the objective function evaluation is computed for $\mathrm{NP}\times N_i$, where NP is population size and N_i is the number of generations. A population of PSO and the DEs calls Otsu's function one time per generation. The population size in the PSO and DEs algorithms was set to 50. A bee in the ABC calls Otsu's function two times per generation; therefore their number of food sources were set to a half of the PSO's size, that is, 25. The stopping criteria were set by the maximum amount of generations G. In this experiment, G was set to 50, 100, 150, 200, 300, 400, 600, 800, 1000, 1500, 2000, 3000, 4000, 5000, and 6000 when n was 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, and 16, respectively. For the PSO, DPSO, and FODPSO algorithms, the parameters were set as per the suggestion in [30] and is shown in Table 1. The other control parameter of the ABC algorithm, *limit*, was set to 50 [15]. The control parameters F and CR of the DE algorithms were set to 0.5 and 0.9, respectively [31, 32].

5.2. Comparison Strategies and Metrics. To compare the performance of different algorithms, there are three metrics: (1) the convergence rate of algorithms was compared by the average of generations ($\overline{\mathrm{NG}}$), a lower $\overline{\mathrm{NG}}$ means a faster convergence rate; (2) the stability of algorithms was compared by the average of the success rate, ($\mathrm{SR}_{\mathrm{HM}}$), a higher $\mathrm{SR}_{\mathrm{HM}}$ means higher stability; (3) the reliability was compared by the threshold value distortion measure (TVD), a lower TVD means higher reliability. The details of the three metrics are described as follows.

When all 50 runs of an algorithm performing on an image with a specific number of thresholds are terminated, the outcomes will be analyzed. Run r'th is called a successful run if there is a generation of $t\le \mathrm{G}$ such that $\mathrm{BCV}_r(t)\ge \mathrm{VTR}$ and the number of generations (NG) of the successful run is recorded. Thus, the number can be defined by

$$\mathrm{NG}_r=\operatorname*{ArgMin}_t\left(\mathrm{BCV}_r(t)\ge \mathrm{VTR}\right),\quad \text{if } r \text{ is a successful run and otherwise undefined.} \tag{16}$$

The average of NG_r from those successful runs is represented by $\overline{\mathrm{NG}}$ as follows:

$$\overline{\mathrm{NG}}=\frac{1}{\text{number of successful runs}}\sum_{\text{All successful runs}}\mathrm{NG}_r. \tag{17}$$

The ratio of success rate (SR) for which the algorithm succeeds to reach the VTR for each subproblem is computed as

$$\mathrm{SR}=\frac{\text{number of successful runs}}{\text{total number of runs}}. \tag{18}$$

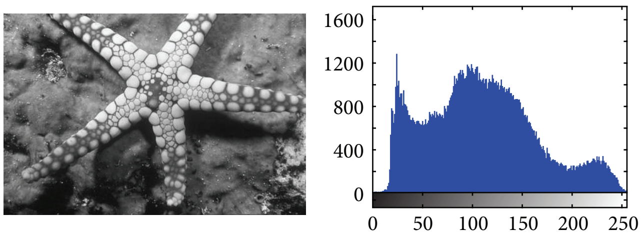
The experiments were conducted on 20 images. The arithmetic mean (AM) of $\overline{\mathrm{NG}}$ ($\overline{\mathrm{NG}}_{\mathrm{AM}}$) over the entire set of images with a specific number of thresholds is calculated as

$$\overline{\mathrm{NG}}_{\mathrm{AM}}=\frac{1}{N}\sum_{\text{All images}}\overline{\mathrm{NG}}, \tag{19}$$

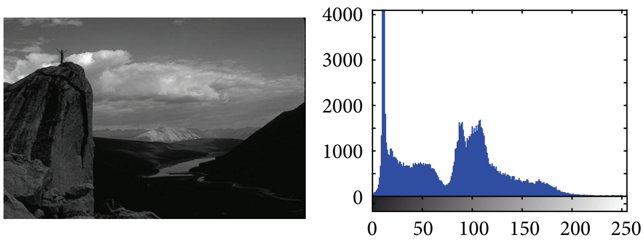
where N is the total number of images. $\overline{\mathrm{NG}}_{\mathrm{AM}}$ is shown in Table 3. The worst-case scenario is that there is no successful run for a subproblem; this subproblem is called an "unsuccessful subproblem." If an algorithm encounters this scenario, the subproblem will be grouped by its number of thresholds and the number of images in the group will be counted and assigned to x. These scenarios will be represented by $\mathrm{NA}(x)$, as shown in Tables 3 and 4.

Table 1: Essential parameters of the PSO, DPSO, and FODPSO taken from [30].

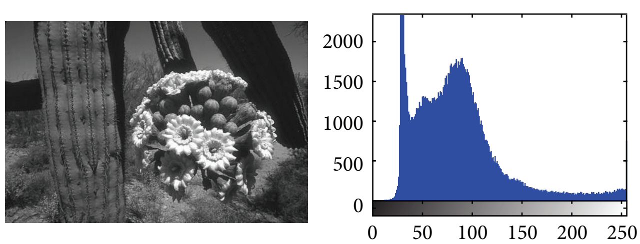
Parameter	PSO	DPSO	FODPSO
Population	50	50	50
ρ_1	1.5	1.5	1.5
ρ_2	1.5	1.5	1.5
W	1.2	1.2	1.2
$V_{\max}$	2	2	2
$V_{\min}$	−2	−2	−2
$x_{\max}$	255	255	255
$x_{\min}$	0	0	0
Min population	—	10	10
Max population	—	50	50
No. of swarms	—	4	4
Min swarms	—	2	2
Max swarms	—	6	6
Stagnancy	—	10	10
Fractional coefficient	—	—	0.75



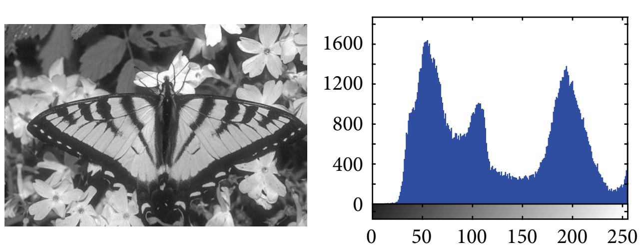
(1) Starfish (481 × 321)



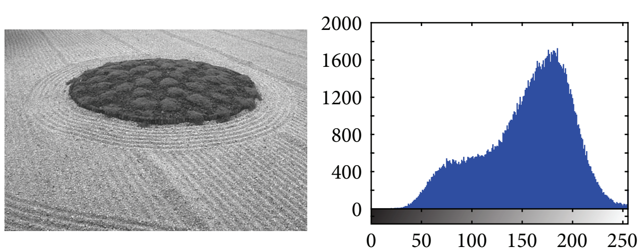
(2) Mountain (481 × 321)



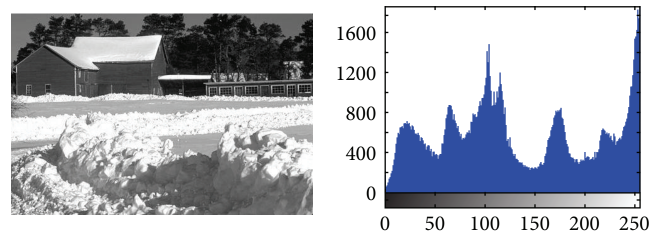
(3) Cactus (481 × 321)



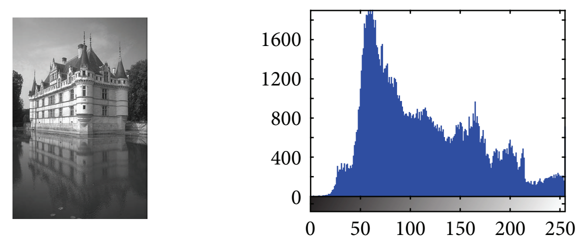
(4) Butterfly (481 × 321)



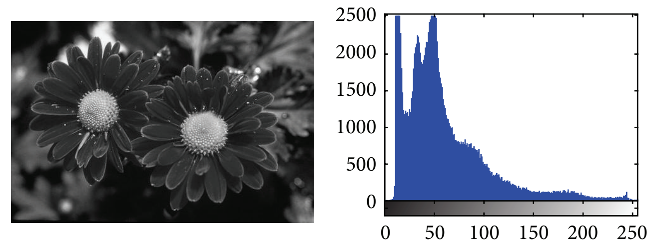
(5) Circus (481 × 321)



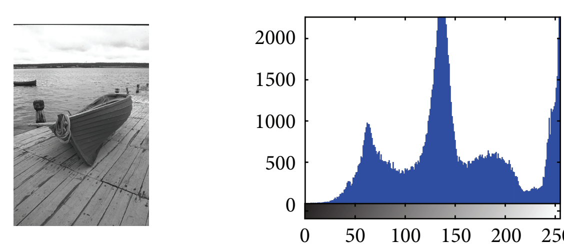
(6) Snow (481 × 321)



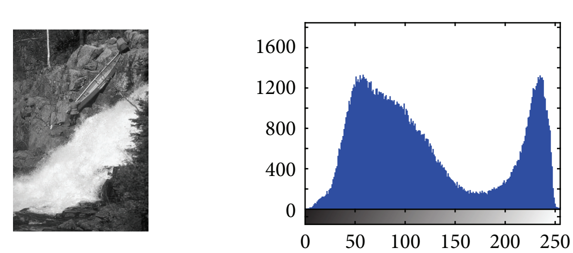
(7) Palace (321 × 481)



(8) Flower (481 × 321)



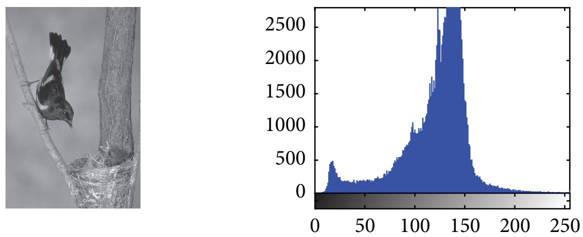
(9) Wherry (321 × 481)



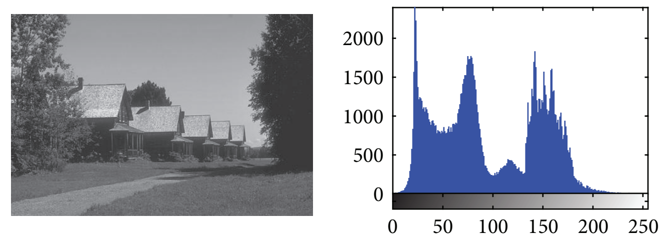
(10) Waterfall (321 × 481)

(a)

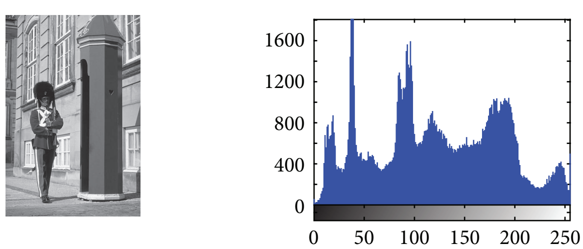
FIGURE 1: Continued.



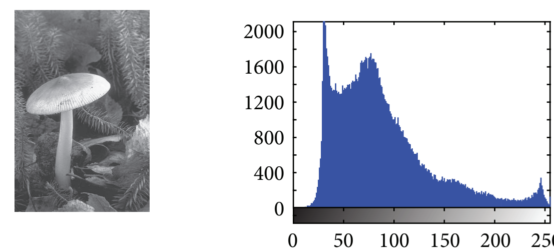
(11) Bird (321 × 481)



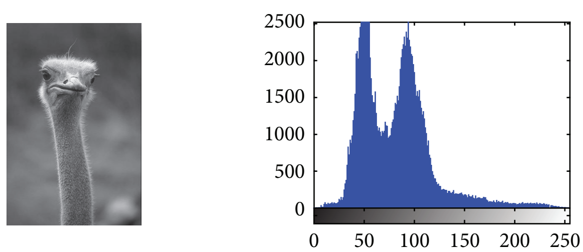
(16) Houses (481 × 321)



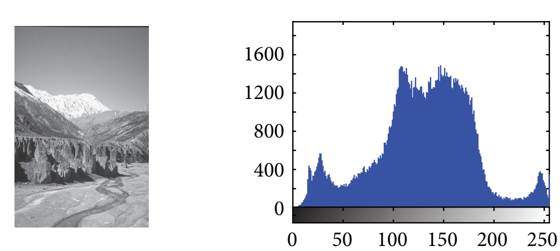
(12) Police (321 × 481)



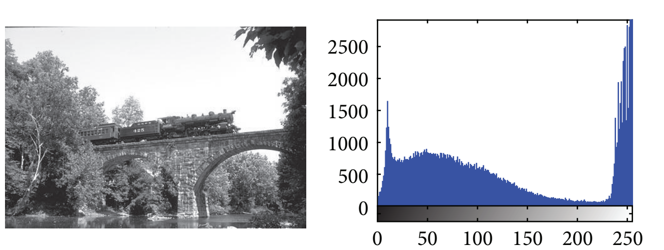
(17) Mushroom (321 × 481)



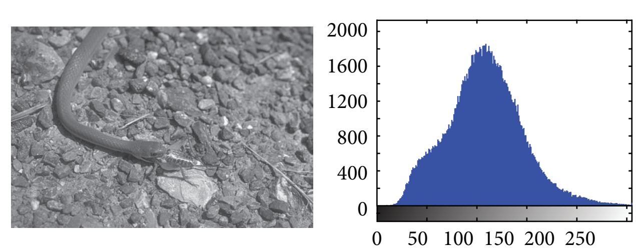
(13) Ostrich (321 × 481)



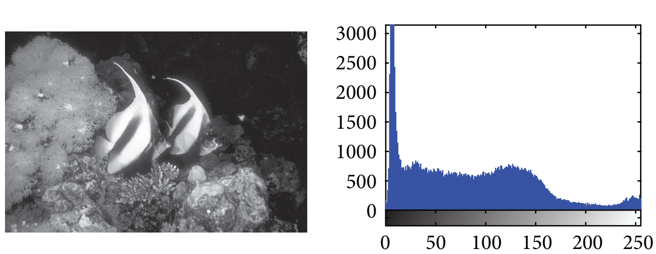
(18) Snow mountain (321 × 481)



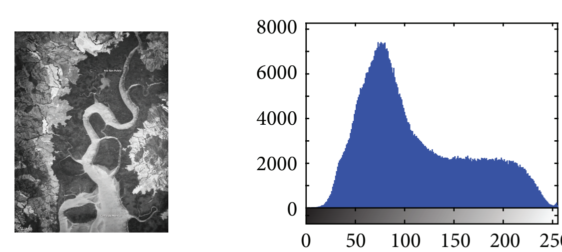
(14) Viaduct (481 × 321)



(19) Snake (481 × 321)



(15) Fish (481 × 321)



(20) Riosanpablo (720 × 944)

(b)

Figure 1: The test images and corresponding histograms.

The average of the success rate over the entire dataset with a specific number of thresholds (SR_{HM}) is averaged by the Harmonic mean, HM, as follows:

$$SR_{HM} = \frac{N}{\sum_{\text{All images}} (1/SR)}. \quad (20)$$

The SR_{HM} is very important in measuring the stability of an algorithm and it means the ratio of runs that are achieving the target solution. Because the evolutionary methods are based on stochastic searching algorithms, the solutions are not the same in each run of the algorithm and depend on the search ability of the algorithm. Therefore, the SR_{HM} is vital in evaluating the stability of the algorithms. The comparison of the stability gives us valuable information in terms of the ratio representing the success rates (SR_{HM}). A higher SR_{HM} means better stability of the algorithm.

An algorithm producing $SR_{HM} < 0.5$ means that more than 50 percent of the independent runs of the algorithm cannot reach the global solution. Thus, the algorithm that yields $SR_{HM} < 0.5$ should not be selected to solve the problem. The experiments were conducted for the number of thresholds varying from 2 to 16. These experiments contained the maximum number of thresholds such that the algorithm yields $SR_{HM} \geq 0.5$, which is represented by $n_{0.5}$ in Table 4. Furthermore, the experiments also contained the maximum number of thresholds that the algorithm can solve; and above this value there was the case such that all 50 runs of some subproblems missed the VTR. This number is represented by $n_{\max}$. In this case the success rate was zero and the associated SR_{HM} was zero too. And the definitions of the two values are presented in (21)

$$\begin{aligned} n_{0.5} &= \max\left(\{n \mid n = \text{number of thresholds that} \right. \\ &\qquad \left. \text{has } SR_{HM} \geq 0.5\}\right) \\ n_{\max} &= \min\left(\{n \mid n = \text{number of thresholds that} \right. \\ &\qquad \left. \text{has } SR_{HM} = 0\}\right) - 1. \end{aligned} \quad (21)$$

Let n be the number of thresholds. The reliability of a solution is measured by threshold value distortion measure (TVD) and is computed as

$$\begin{aligned} \text{TVD} &= \frac{\sum_{\text{All images}} \sum_{r=1}^{\text{run}} \sum_{i=1}^{n} \left| T_{ri}^{*} - T_{ri}^{m} \right|}{1 + \sum_{\text{All images}} \sum_{r=1}^{\text{run}} \sum_{i=1}^{n} 1_{\{T_{ri}^{*} \neq T_{ri}^{m}\}}} \\ &\quad \times (1 - SR) \times 100, \end{aligned} \quad (22)$$

where T^{*} is the threshold value producing the VTR, T^{m} is the threshold value obtained from the algorithm, and $1_{\{T_{ri}^{*} \neq T_{ri}^{m}\}}$ is the indicator function, which is equal to 1 when $T_{ri}^{*} \neq T_{ri}^{m}$ and is zero otherwise. TVD is zero if the algorithm can reach the VTR in every run. The lower the TVD the more reliable the algorithm is.

5.2.1. The Value to Reach (VTR). Following the completion of all of the experiments the best values of the between-class variance and the corresponding thresholds were collected and are shown in Table 6. The results are shown image by image and the numbers of thresholds vary from 2 to 16. The between-class variance values in column 3 are used as the VTR values.

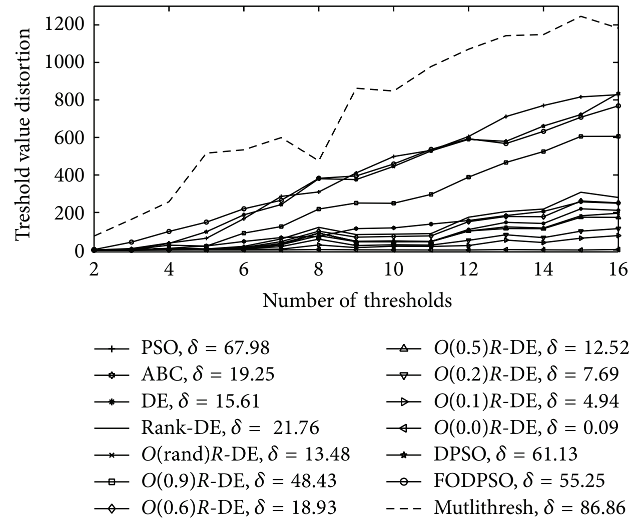


Figure 2: The threshold value distortion (TVD) of algorithms versus number of thresholds.

5.2.2. Results Produced by Local Search Method. The multithresh function of the Matlab toolbox was conducted on the same images and number of thresholds as the other search methods. The capabilities of solving the optimal solution between a local search and a global search will be discussed here. This is the reason we focused on the global search, that is, the proposed $O(0.0)R$-DE algorithm. Table 2 shows the between-class variances and threshold values of the "mountain" image. These values were the best outcomes of 50 runs produced by the multithresh function in the Matlab R2013a toolbox and by the proposed $O(0.0)R$-DE algorithm. The terminated condition of the multithresh function was set by "MaxFunEvals" = 500000. That is the multithresh function performs more function calls than that of the $O(0.0)R$-DE algorithm. It can be seen from columns 3 and 5 that all the BCVs produced by the $O(0.0)R$-DE algorithm are better than the BCVs produced by the multithresh function; the difference of the BCVs is shown in column 7. The differences in the thresholds from the two algorithms, shown in column 8, tended to be large if the number of thresholds increased.

Figure 2 shows the graph of the TVD of all the images and thresholds. These results are in the same pattern of the results of the "mountain" image in Table 2. That means the ability to search for the optimal solution of the proposed global search algorithm is higher than that of the multithresh function, especially when the number of thresholds is large. This goes to illustrate the difficulty of the problem. The problem with this kind is that it can be multimodal [33] or can be a nearly flat top surface [34]. The multithresh function solves the problem by performing the Nelder-Mead Simplex

Table 2: The best values of the between-class variance and thresholds of the "mountain" image produced by the multithresh function of the Matlab toolbox and by the proposed $O(0.0)R$-DE algorithm. The number of thresholds varies from 2 to 16.

		Produced by multithresh		Produced by the $O(0.0)R$-DE algorithm			
Image (1)	n (2)	Between-class variance (3)	Thresholds (4)	Between-class variance (5)	Thresholds (6)	Objdiff (7) = (5) − (3)	Threshdiff (8) = $\sum \lvert (4)-(6) \rvert$
	2	2372.886	60, 127	2372.923	61, 128	0.037	2
	3	2495.852	32, 76, 130	2496.113	33, 77, 131	0.261	3
	4	2551.778	32, 72, 109, 145	2551.955	33, 73, 109, 147	0.177	4
	5	2580.154	31, 68, 98, 124, 158	2580.336	32, 69, 99, 125, 159	0.182	5
	6	2588.264	35, 75, 98, 118, 152, 175	2596.956	24, 46, 74, 101, 126, 160	8.692	122
	7	2598.800	33, 70, 98, 118, 141, 166, 207	2608.807	24, 46, 73, 98, 119, 145, 175	10.007	153
	8	2605.785	31, 68, 91, 105, 121, 140, 163, 192	2616.294	24, 46, 73, 97, 115, 135, 160, 191	10.509	70
Mountain	9	2619.512	27, 52, 75, 95, 111, 129, 148, 168, 197	2622.314	20, 37, 54, 76, 98, 116, 136, 160, 191	2.802	114
	10	2620.392	23, 46, 73, 97, 116, 135, 157, 175, 204, 232	2627.194	20, 36, 53, 74, 93, 106, 121, 140, 163, 194	6.802	258
	11	2625.275	23, 45, 72, 93, 105, 119, 137, 156, 175, 204, 228	2630.496	20, 36, 53, 74, 93, 105, 118, 134, 152, 172, 201	5.221	199
	12	2629.641	22, 39, 58, 78, 95, 109, 123, 139, 156, 177, 207, 247	2633.189	18, 31, 45, 59, 76, 93, 105, 118, 134, 152, 172, 201	3.548	246
	13	2631.372	20, 38, 54, 74, 94, 107, 121, 139, 157, 173, 193, 224, 248	2635.290	18, 31, 45, 59, 76, 92, 103, 115, 129, 144, 161, 179, 207	3.918	283
	14	2633.865	19, 36, 53, 74, 91, 103, 115, 129, 144, 160, 174, 190, 217, 245	2637.088	17, 29, 42, 56, 72, 86, 96, 106, 117, 130, 145, 161, 179, 207	3.223	307
	15	2635.190	19, 33, 50, 69, 85, 96, 107, 121, 137, 152, 167, 178, 190, 212, 240	2638.449	17, 28, 39, 50, 61, 75, 88, 97, 107, 118, 131, 146, 162, 179, 207	3.259	351
	16	2636.357	16, 30, 45, 59, 77, 93, 105, 118, 133, 148, 162, 176, 192, 209, 230, 244	2639.616	17, 27, 38, 49, 60, 74, 86, 95, 104, 113, 123, 135, 149, 164, 181, 209	3.259	415

Table 3: The average of mean number of generation ($\overline{\mathrm{NG}}_{\mathrm{AM}}$) and ranks of the methods.

No. maxGen.		FODPSO	DPSO	PSO	ABC	Rank-DE	DE	O(rand)R-DE	$O(0.9)R$-DE	$O(0.8)R$-DE	$O(0.7)R$-DE	$O(0.6)R$-DE	$O(0.5)R$-DE	$O(0.4)R$-DE	$O(0.3)R$-DE	$O(0.2)R$-DE	$O(0.1)R$-DE	$O(0.0)R$-DE
$n = 2$ $G = 50$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	18.337	15.713	13.218	13.202	10.538	14.713	10.117	6.394	6.903	7.636	8.208	9.147	9.261	9.901	10.709	11.631	20.850
	Rank	16	15	13	12	9	14	8	1	2	3	4	5	6	7	10	11	17
$n = 3$ $G = 100$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	35.200	36.210	27.479	33.210	19.488	29.049	18.958	10.492	12.296	13.587	14.807	16.969	17.107	18.100	19.546	21.749	44.192
	Rank	15	16	12	14	9	13	8	1	2	3	4	5	6	7	10	11	17
$n = 4$ $G = 150$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	45.391	58.683	49.596	63.251	31.060	48.130	29.595	14.733	18.526	20.885	22.815	26.632	26.456	28.058	30.175	33.499	72.100
	Rank	12	15	14	16	10	13	8	1	2	3	4	6	5	7	9	11	17
$n = 5$ $G = 200$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	55.664	81.877	87.441	108.131	44.157	69.088	41.283	19.508	24.524	28.097	30.963	36.935	36.748	38.647	41.455	45.631	100.693
	Rank	12	14	15	17	10	13	8	1	2	3	4	6	5	7	9	11	16
$n = 6$ $G = 300$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(2)	100.664	141.937	157.328	61.935	99.700	56.405	24.623	32.687	38.167	42.608	51.003	50.281	52.089	55.729	61.748	138.716
	Rank	17	13	15	16	11	12	9	1	2	3	4	6	5	7	8	10	14
$n = 7$ $G = 400$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(1)	NA(1)	192.267	236.928	86.938	142.096	75.340	29.763	41.691	49.408	56.193	67.619	67.063	69.696	74.389	83.326	184.371
	Rank	17	16	14	15	11	12	9	1	2	3	4	6	5	7	8	10	13
$n = 8$ $G = 600$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(7)	NA(6)	NA(2)	NA(1)	117.092	198.986	100.944	33.967	51.573	62.263	74.523	91.836	89.666	95.198	102.552	115.583	244.880
	Rank	17	16	15	14	11	12	8	1	2	3	4	6	5	7	9	10	13
$n = 9$ $G = 800$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(11)	NA(13)	NA(12)	NA(3)	151.004	264.253	125.299	41.228	63.742	79.432	90.201	113.530	110.416	114.397	122.555	137.538	299.492
	Rank	17	16	15	14	11	12	9	1	2	3	4	6	5	7	8	10	13
$n = 10$ $G = 1000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(11)	NA(16)	NA(16)	NA(7)	NA(1)	321.373	147.770	NA(3)	73.564	91.121	105.020	132.968	128.999	132.691	141.825	157.997	356.546
	Rank	17	16	15	14	12	10	8	13	1	2	3	6	4	5	7	9	11
$n = 11$ $G = 1500$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(16)	NA(17)	NA(19)	NA(9)	NA(1)	405.090	185.020	NA(3)	NA(2)	NA(2)	127.135	162.822	157.184	160.123	175.152	198.522	434.877
	Rank	17	16	15	14	12	8	6	13	10	10	1	4	2	3	5	7	9
$n = 12$ $G = 2000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(18)	NA(20)	NA(20)	NA(9)	278.743	535.542	229.965	NA(4)	NA(5)	135.266	160.047	205.266	197.386	203.814	222.094	251.535	574.104
	Rank	17	16	15	14	12	8	6	13	11	10	1	4	2	3	5	7	9
$n = 13$ $G = 3000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(20)	NA(19)	NA(20)	NA(14)	NA(3)	NA(3)	275.860	NA(12)	NA(5)	NA(1)	NA(1)	NA(1)	NA(1)	261.522	278.082	316.468	726.521
	Rank	17	16	15	14	12	9	2	13	11	10	6	6	6	1	3	4	5
$n = 14$ $G = 4000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(20)	NA(20)	NA(20)	NA(14)	NA(4)	NA(4)	330.194	NA(11)	NA(8)	NA(4)	NA(1)	NA(2)	296.414	NA(1)	331.626	385.951	889.711
	Rank	17	16	15	14	12	9	1	13	11	10	7	8	6	5	2	3	4
$n = 15$ $G = 5000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(20)	NA(20)	NA(20)	NA(19)	NA(13)	NA(5)	399.228	NA(11)	NA(10)	NA(8)	NA(6)	363.506	362.665	383.788	417.958	481.830	1086.001
	Rank	17	16	15	14	13	8	1	12	11	10	9	7	6	5	2	3	4
$n = 16$ $G = 6000$	$\overline{\mathrm{NG}}_{\mathrm{AM}}$	NA(20)	NA(20)	NA(20)	NA(18)	NA(10)	NA(8)	NA(4)	NA(18)	NA(11)	NA(9)	NA(5)	NA(1)	NA(2)	447.570	491.772	552.856	1374.438
	Rank	17	16	15	14	12	9	4	13	11	10	8	6	7	5	1	2	3
Avg for $n = 2$ to 16	$n_{\max}$, NA(x) $\overline{\mathrm{NG}}_{\mathrm{AM}}$	5, NA(146), 38.648	6, NA(152), 58.629	7, NA(149), 85.323	7, NA(94), 102.008	9, NA(32), 88.995	12, NA(20), 193.456	15, NA(4), 144.713	9, NA(62), 22.589	10, NA(41), 36.167	10, NA(24), 52.586	12, NA(13), 66.593	12, NA(4), 106.519	12, NA(3), 119.204	13, NA(1), 143.971	16, NA(0), 167.708	16, NA(0), 190.391	16, NA(0), 436.499
	Rank	17	16	15	14	12	9	4	13	11	10	8	7	6	5	1	2	3

Table 4: The average success rate (SR_{HM}) and ranks of the methods.

No		Multi-thresh	FODPSO	DPSO	PSO	ABC	Rank-DE	DE	O(rand)R-DE	$O(0.9)R$-DE	$O(0.8)R$-DE	$O(0.7)R$-DE	$O(0.6)R$-DE	$O(0.5)R$-DE	$O(0.4)R$-DE	$O(0.3)R$-DE	$O(0.2)R$-DE	$O(0.1)R$-DE	$O(0.0)R$-DE
2	SR_{HM}	0	0.977	1	1	1	1	1	1	0.999	1	1	1	1	1	1	1	1	1
	Rank	18	17	1	1	1	1	1	1	16	1	1	1	1	1	1	1	1	1
3	SR_{HM}	0.000	0.635	0.972	0.991	0.979	0.998	1.000	0.997	0.986	0.993	0.999	0.997	0.998	0.999	1.000	1.000	1.000	1.000
	Rank	18	17	16	13	15	8	1	10	14	12	6	10	8	6	1	1	1	1
4	SR_{HM}	0.000	0.379	0.742	0.934	0.833	0.987	0.995	0.986	0.897	0.945	0.972	0.971	0.986	0.993	0.992	0.991	0.998	0.997
	Rank	18	17	16	13	15	7	3	8	14	12	10	11	8	4	5	6	1	2
5	SR_{HM}	0.000	0.185	0.391	0.767	0.706	0.936	0.977	0.960	0.782	0.878	0.874	0.944	0.950	0.959	0.968	0.960	0.985	1.000
	Rank	18	17	16	14	15	10	3	5	13	11	12	9	8	7	4	5	2	1
6	SR_{HM}	0.000	0.000	0.187	0.433	0.453	0.863	0.968	0.900	0.601	0.706	0.826	0.878	0.934	0.922	0.944	0.935	0.970	0.999
	Rank	18	17	16	15	14	10	3	8	13	12	11	9	6	7	4	5	2	1
7	SR_{HM}	0.000	0.000	0.000	0.153	0.164	0.696	0.884	0.835	0.366	0.561	0.675	0.741	0.840	0.851	0.883	0.914	0.949	0.998
	Rank	18	17	16	15	14	10	4	8	13	12	11	9	7	6	5	3	2	1
8	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.300	0.458	0.530	0.105	0.256	0.351	0.502	0.592	0.628	0.724	0.757	0.838	0.972
	Rank	18	17	16	14	14	11	9	7	13	12	10	8	6	5	4	3	2	1
9	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.495	0.695	0.681	0.083	0.274	0.372	0.548	0.721	0.725	0.769	0.815	0.852	0.999
	Rank	18	17	16	14	14	10	7	8	13	12	11	9	6	5	4	3	2	1
10	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.381	0.375	0.000	0.073	0.218	0.311	0.409	0.370	0.460	0.578	0.686	0.998
	Rank	18	17	16	14	14	12	6	7	13	11	10	9	5	8	4	3	2	1
11	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.271	0.267	0.000	0.000	0.000	0.144	0.407	0.316	0.424	0.584	0.704	0.999
	Rank	18	17	16	14	14	12	7	8	13	10	10	9	5	6	4	3	2	1
12	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.090	0.270	0.423	0.000	0.000	0.089	0.177	0.400	0.407	0.501	0.641	0.810	1.000
	Rank	18	17	16	14	14	12	8	5	13	11	10	9	7	6	4	3	2	1
13	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.151	0.000	0.000	0.000	0.000	0.000	0.000	0.254	0.267	0.381	0.974
	Rank	18	17	16	14	14	12	6	5	13	10	10	6	6	6	4	3	2	1
14	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.102	0.000	0.000	0.000	0.000	0.000	0.110	0.000	0.253	0.494	0.993
	Rank	18	17	16	14	14	12	7	4	13	10	10	7	7	6	5	3	2	1
15	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.090	0.000	0.000	0.000	0.000	0.095	0.117	0.149	0.288	0.466	1.000
	Rank	18	17	16	14	14	12	8	4	13	10	10	8	7	6	5	3	2	1
16	SR_{HM}	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.110	0.192	0.288	0.977
	Rank	18	17	16	14	14	12	6	4	13	10	10	6	6	6	5	3	2	1
Avg. for $n = 2$ to 16	$n_{0.5}$,	<2,	3,	4,	5,	5,	7,	9,	9,	6,	7,	7,	9,	9,	9,	9,	12,	12,	16,
	n_{max},	<2,	5,	6,	7,	7,	9,	12,	15,	9,	10,	10,	12,	12,	12,	13,	16,	16,	16,
	SR_{HM}	0.000	0.376	0.443	0.454	0.464	0.665	0.554	0.304	0.263	0.313	0.530	0.420	0.654	0.625	0.619	0.498	0.656	0.994
	Rank	18	17	16	15	14	12	8	4	13	11	10	9	6	7	5	3	2	1

Table 5: The average of threshold value distortion (TVD) and ranks of the methods.

No		Multi-thresh	FODPSO	DPSO	PSO	ABC	Rank-DE	DE	O(rand)R-DE	O(0.9)R-DE	O(0.8)R-DE	O(0.7)R-DE	O(0.6)R-DE	O(0.5)R-DE	O(0.4)R-DE	O(0.3)R-DE	O(0.2)R-DE	O(0.1)R-DE	O(0.0)R-DE
2	TVD	74.153	2.217	0.000	0.000	0.000	0.000	0.000	0.000	0.050	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	Rank	18	17	1	1	1	1	1	1	16	1	1	1	1	1	1	1	1	1
3	TVD	163.333	41.753	2.968	7.424	1.510	0.150	0.000	0.270	1.004	0.519	0.075	0.246	0.171	0.075	0.000	0.000	0.000	0.000
	Rank	18	17	15	16	14	8	1	11	13	12	6	10	9	6	1	1	1	1
4	TVD	257.083	98.848	34.770	36.891	8.716	7.340	1.428	2.184	25.499	13.609	7.318	7.360	7.398	4.303	1.672	0.728	0.130	0.257
	Rank	18	17	15	16	12	9	4	6	14	13	8	10	11	7	5	3	1	2
5	TVD	516.582	148.344	97.216	62.319	21.565	5.373	1.864	3.241	20.441	10.064	10.422	4.584	3.904	3.289	2.441	3.385	1.195	0.000
	Rank	18	17	16	15	14	10	3	5	13	11	12	9	8	6	4	7	2	1
6	TVD	534.226	219.437	187.620	166.775	45.728	19.536	2.528	7.761	89.938	43.666	30.923	13.010	6.524	6.023	4.136	6.129	2.421	0.050
	Rank	18	17	16	15	13	10	3	8	14	12	11	9	7	5	4	6	2	1
7	TVD	599.250	265.029	242.194	285.862	66.259	58.877	27.976	35.481	125.040	75.011	49.851	44.610	28.039	29.841	26.075	17.950	4.776	0.150
	Rank	18	16	15	17	12	11	5	8	14	13	10	9	6	7	4	3	2	1
8	TVD	475.458	382.076	380.124	309.635	72.763	120.734	90.287	89.406	218.266	143.225	130.898	99.991	76.353	82.886	57.251	57.665	26.822	2.103
	Rank	18	17	16	15	5	11	9	8	14	13	12	10	6	7	3	4	2	1
9	TVD	862.458	394.744	375.229	410.710	113.892	82.099	44.407	46.539	250.470	154.662	96.417	68.197	44.223	35.333	26.627	23.406	12.757	0.780
	Rank	18	16	15	17	12	10	7	8	14	13	11	9	6	5	4	3	2	1
10	TVD	847.703	458.536	445.765	498.328	117.664	84.541	43.991	48.474	249.147	136.935	93.935	72.288	43.883	44.110	36.700	27.978	20.213	0.164
	Rank	18	16	15	17	12	10	6	8	14	13	11	9	5	7	4	3	2	1
11	TVD	977.976	536.258	528.329	531.211	137.941	87.267	44.944	44.973	296.513	152.195	104.092	75.115	44.461	47.564	34.944	26.231	18.782	0.083
	Rank	18	17	15	16	12	10	6	7	14	13	11	9	5	8	4	3	2	1
12	TVD	1070.300	592.750	589.931	605.280	157.169	174.580	109.138	101.282	388.491	258.312	196.098	150.932	101.112	82.365	79.966	51.281	23.429	0.000
	Rank	18	16	15	17	10	11	8	7	14	13	12	9	6	5	4	3	2	1
13	TVD	1142.808	567.442	579.495	711.382	183.451	204.665	147.251	123.438	466.508	299.040	216.656	178.394	114.244	114.491	96.104	81.076	52.578	1.975
	Rank	18	15	16	17	10	11	8	7	14	13	12	9	5	6	4	3	2	1
14	TVD	1148.641	632.482	660.908	770.319	205.714	218.408	141.431	116.716	524.783	325.936	241.997	177.121	113.426	104.073	88.593	65.908	39.647	0.564
	Rank	18	15	16	17	10	11	8	7	14	13	12	9	6	5	4	3	2	1
15	TVD	1245.529	707.835	722.126	816.158	256.477	307.873	219.556	182.464	605.623	428.370	343.439	261.522	174.700	166.309	138.710	100.296	62.940	0.000
	Rank	18	15	16	17	9	11	8	7	14	13	12	10	6	5	4	3	2	1
16	TVD	1182.989	768.714	835.747	828.004	249.042	279.753	211.614	195.676	605.790	416.711	311.314	251.408	174.118	167.316	133.922	113.776	76.661	2.539
	Rank	18	15	17	16	9	11	8	7	14	13	12	10	6	5	4	3	2	1
Avg. for $n = 2$ to 16	TVD	739.899	387.764	378.828	402.687	109.193	110.080	72.428	66.527	257.838	163.884	122.229	93.652	62.170	59.199	48.476	38.387	22.823	0.578
	Growth rate (δ)	86.86	55.25	61.13	67.98	19.25	21.76	15.61	13.48	48.43	32.25	23.91	18.93	12.52	11.86	9.99	7.69	4.94	0.09
	Rank	18	15	16	17	10	11	8	7	14	13	12	9	6	5	4	3	2	1

Table 6: The between-class variance criterion and the best thresholds for test images.

Image	n	Between-class variance (VTR)	Thresholds
Starfish	2	2546.885	85, 157
	3	2779.925	68, 119, 177
	4	2865.707	60, 101, 138, 187
	5	2912.859	52, 86, 117, 150, 194
	6	2941.728	47, 77, 105, 132, 162, 201
	7	2960.158	44, 71, 95, 118, 142, 170, 206
	8	2972.356	43, 68, 90, 110, 131, 153, 180, 212
	9	2981.138	38, 58, 78, 97, 116, 136, 157, 183, 214
	10	2988.206	37, 56, 75, 93, 110, 128, 146, 167, 192, 219
	11	2993.348	35, 53, 71, 88, 103, 119, 135, 152, 172, 196, 221
	12	2997.352	34, 51, 68, 84, 98, 112, 127, 142, 158, 177, 200, 223
	13	3000.480	33, 48, 64, 79, 93, 106, 119, 133, 147, 162, 181, 203, 225
	14	3003.076	32, 47, 62, 76, 89, 101, 114, 127, 140, 154, 169, 187, 207, 227
	15	3005.235	30, 43, 56, 70, 83, 95, 107, 119, 131, 143, 156, 171, 189, 209, 228
	16	3007.060	30, 42, 55, 68, 80, 92, 103, 114, 126, 138, 150, 163, 178, 195, 213, 230
Mountain	2	2372.923	61, 128
	3	2496.113	33, 77, 131
	4	2551.955	33, 73, 109, 147
	5	2580.336	32, 69, 99, 125, 159
	6	2596.956	24, 46, 74, 101, 126, 160
	7	2608.807	24, 46, 73, 98, 119, 145, 175
	8	2616.294	24, 46, 73, 97, 115, 135, 160, 191
	9	2622.314	20, 37, 54, 76, 98, 116, 136, 160, 191
	10	2627.194	20, 36, 53, 74, 93, 106, 121, 140, 163, 194
	11	2630.496	20, 36, 53, 74, 93, 105, 118, 134, 152, 172, 201
	12	2633.189	18, 31, 45, 59, 76, 93, 105, 118, 134, 152, 172, 201
	13	2635.290	18, 31, 45, 59, 76, 92, 103, 115, 129, 144, 161, 179, 207
	14	2637.088	17, 29, 42, 56, 72, 86, 96, 106, 117, 130, 145, 161, 179, 207
	15	2638.449	17, 28, 39, 50, 61, 75, 88, 97, 107, 118, 131, 146, 162, 179, 207
	16	2639.616	17, 27, 38, 49, 60, 74, 86, 95, 104, 113, 123, 135, 149, 164, 181, 209
Cactus	2	1816.448	73, 151
	3	1970.112	64, 106, 173
	4	2042.275	55, 87, 125, 187
	5	2080.884	49, 76, 102, 138, 196
	6	2104.147	46, 70, 93, 119, 157, 208
	7	2119.670	44, 65, 85, 105, 131, 168, 215
	8	2129.358	42, 60, 77, 94, 113, 138, 174, 218
	9	2136.162	41, 58, 74, 89, 105, 125, 152, 186, 224
	10	2141.579	40, 56, 71, 85, 99, 115, 135, 161, 193, 228
	11	2145.397	39, 53, 66, 79, 91, 104, 119, 139, 165, 196, 229
	12	2148.282	38, 51, 64, 76, 88, 100, 114, 131, 152, 176, 204, 233
	13	2150.740	37, 49, 61, 72, 83, 94, 105, 118, 135, 155, 178, 205, 233
	14	2152.626	36, 47, 58, 68, 78, 88, 98, 109, 122, 138, 158, 181, 207, 234
	15	2154.162	36, 46, 56, 66, 76, 85, 94, 104, 115, 128, 144, 164, 187, 212, 237
	16	2155.471	36, 46, 56, 66, 75, 84, 93, 102, 112, 124, 138, 155, 174, 195, 217, 239

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
Butterfly	2	3873.222	84, 155
	3	3990.855	81, 144, 199
	4	4051.357	77, 121, 167, 207
	5	4092.929	60, 89, 129, 172, 209
	6	4119.448	59, 87, 122, 161, 193, 221
	7	4135.937	53, 74, 98, 128, 165, 195, 222
	8	4148.556	53, 73, 96, 123, 156, 183, 203, 228
	9	4156.528	52, 72, 94, 118, 144, 170, 190, 207, 231
	10	4163.794	48, 63, 80, 99, 121, 147, 172, 191, 208, 231
	11	4168.489	47, 62, 79, 98, 118, 141, 164, 183, 198, 213, 234
	12	4172.517	46, 59, 73, 89, 104, 122, 144, 167, 185, 199, 214, 235
	13	4175.487	46, 59, 72, 87, 102, 118, 137, 157, 175, 189, 202, 216, 236
	14	4177.893	44, 55, 66, 79, 93, 106, 121, 141, 161, 178, 191, 203, 217, 237
	15	4179.908	44, 55, 66, 78, 92, 105, 119, 137, 156, 173, 186, 197, 208, 221, 239
	16	4181.559	42, 52, 61, 71, 83, 95, 107, 121, 139, 157, 173, 186, 197, 208, 221, 239
Circus	2	1651.257	118, 172
	3	1760.512	105, 150, 187
	4	1817.487	93, 132, 165, 195
	5	1850.243	87, 122, 152, 177, 203
	6	1870.083	82, 113, 141, 164, 185, 208
	7	1883.966	77, 104, 129, 151, 171, 190, 212
	8	1893.450	73, 98, 122, 143, 161, 178, 195, 216
	9	1900.592	70, 92, 114, 134, 152, 168, 183, 199, 219
	10	1905.992	68, 89, 109, 128, 145, 160, 174, 188, 203, 222
	11	1909.887	66, 86, 105, 123, 139, 153, 166, 179, 192, 206, 225
	12	1913.079	63, 81, 98, 114, 130, 145, 158, 170, 182, 194, 208, 226
	13	1915.676	62, 79, 95, 111, 126, 140, 152, 164, 175, 186, 197, 210, 228
	14	1917.756	61, 78, 94, 109, 123, 136, 148, 159, 170, 180, 190, 201, 214, 231
	15	1919.524	59, 74, 88, 102, 115, 128, 140, 151, 161, 171, 181, 191, 202, 214, 231
	16	1920.958	58, 73, 87, 100, 113, 126, 138, 149, 159, 169, 178, 187, 196, 206, 218, 234
Snow	2	5261.705	80, 169
	3	5624.289	71, 139, 207
	4	5729.116	50, 92, 144, 208
	5	5785.138	49, 91, 140, 192, 231
	6	5819.333	45, 81, 111, 148, 194, 232
	7	5835.770	43, 76, 101, 127, 159, 196, 232
	8	5850.190	30, 55, 83, 107, 133, 163, 197, 233
	9	5862.437	30, 55, 82, 106, 129, 157, 185, 211, 237
	10	5870.284	29, 52, 75, 94, 111, 132, 159, 186, 212, 237
	11	5875.817	29, 52, 75, 94, 111, 132, 158, 183, 206, 227, 244
	12	5880.335	29, 52, 74, 93, 109, 127, 149, 169, 188, 209, 228, 244
	13	5884.513	23, 39, 57, 76, 94, 110, 128, 150, 170, 188, 209, 228, 244
	14	5887.355	22, 37, 54, 70, 84, 98, 112, 129, 151, 170, 188, 209, 228, 244
	15	5889.953	21, 36, 53, 69, 83, 97, 110, 124, 141, 159, 175, 192, 211, 229, 245
	16	5891.998	21, 36, 53, 69, 83, 97, 110, 124, 141, 159, 174, 189, 207, 222, 236, 248

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
Palace	2	2623.440	99, 165
	3	2791.488	84, 132, 186
	4	2860.98	70, 103, 143, 191
	5	2908.165	69, 101, 138, 177, 218
	6	2934.330	64, 89, 117, 147, 181, 220
	7	2953.745	54, 75, 99, 126, 153, 183, 220
	8	2966.250	50, 69, 89, 111, 134, 158, 185, 221
	9	2974.719	47, 65, 81, 100, 120, 141, 163, 188, 222
	10	2981.875	47, 64, 80, 98, 117, 137, 157, 178, 199, 226
	11	2986.898	45, 61, 75, 90, 106, 123, 141, 159, 179, 200, 227
	12	2990.579	43, 58, 69, 82, 96, 111, 126, 143, 160, 180, 201, 227
	13	2993.809	43, 58, 69, 81, 95, 110, 125, 141, 157, 173, 190, 208, 231
	14	2996.112	43, 57, 68, 80, 94, 108, 123, 138, 153, 167, 182, 199, 218, 239
	15	2998.213	42, 56, 66, 77, 88, 100, 113, 126, 140, 154, 167, 182, 199, 218, 239
	16	2999.918	42, 55, 65, 75, 86, 98, 110, 123, 136, 149, 162, 176, 191, 205, 222, 241
Flower	2	1489.281	61, 130
	3	1627.897	39, 77, 141
	4	1685.956	36, 67, 105, 160
	5	1715.220	28, 49, 75, 111, 164
	6	1736.423	27, 47, 71, 102, 143, 192
	7	1752.512	26, 43, 61, 83, 111, 151, 199
	8	1761.968	25, 42, 58, 77, 99, 126, 161, 204
	9	1768.354	24, 40, 54, 70, 88, 109, 135, 167, 207
	10	1772.903	23, 37, 48, 60, 75, 92, 112, 138, 169, 208
	11	1776.470	23, 36, 47, 58, 72, 87, 104, 124, 149, 177, 211
	12	1779.106	22, 34, 44, 54, 65, 78, 92, 108, 128, 152, 179, 212
	13	1781.251	20, 30, 39, 48, 58, 70, 83, 96, 112, 131, 154, 180, 213
	14	1783.049	19, 29, 38, 47, 56, 66, 78, 90, 104, 120, 139, 161, 185, 215
	15	1784.478	19, 29, 38, 46, 54, 63, 74, 85, 97, 111, 128, 147, 168, 190, 218
	16	1785.641	19, 28, 37, 45, 53, 62, 72, 83, 94, 106, 120, 136, 155, 175, 196, 222
Wherry	2	3313.161	108, 189
	3	3543.272	102, 161, 218
	4	3599.924	83, 121, 163, 218
	5	3634.708	81, 118, 152, 184, 224
	6	3656.048	72, 103, 130, 156, 186, 225
	7	3668.313	68, 94, 120, 139, 161, 189, 226
	8	3678.216	60, 83, 110, 132, 152, 175, 198, 230
	9	3686.001	56, 77, 100, 122, 138, 156, 179, 201, 231
	10	3691.730	56, 76, 98, 120, 136, 153, 174, 194, 219, 243
	11	3696.416	55, 74, 94, 114, 130, 142, 158, 178, 197, 222, 245
	12	3699.866	54, 72, 91, 110, 126, 138, 151, 168, 185, 202, 225, 246
	13	3702.619	52, 68, 83, 100, 117, 130, 140, 153, 169, 185, 202, 225, 246
	14	3705.033	52, 68, 83, 100, 117, 130, 140, 152, 167, 182, 197, 215, 235, 249
	15	3706.937	51, 66, 79, 94, 110, 123, 133, 142, 154, 169, 184, 199, 216, 235, 249
	16	3708.484	49, 63, 75, 89, 104, 118, 129, 138, 147, 159, 173, 186, 200, 217, 235, 249

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
Waterfall	2	4512.801	88, 170
	3	4646.137	72, 115, 182
	4	4711.019	67, 103, 150, 204
	5	4752.267	58, 87, 119, 164, 212
	6	4777.063	53, 78, 104, 134, 176, 217
	7	4793.510	48, 70, 92, 116, 147, 186, 221
	8	4805.756	46, 66, 86, 107, 131, 163, 199, 226
	9	4814.097	43, 61, 79, 97, 117, 141, 173, 205, 228
	10	4820.724	40, 57, 73, 90, 108, 128, 152, 182, 210, 230
	11	4825.739	38, 54, 69, 84, 100, 117, 137, 162, 191, 215, 232
	12	4829.689	37, 53, 67, 81, 96, 111, 128, 148, 173, 198, 218, 233
	13	4832.826	35, 50, 63, 76, 89, 103, 117, 133, 153, 177, 201, 220, 234
	14	4835.371	32, 46, 58, 70, 82, 95, 108, 122, 138, 158, 182, 204, 221, 234
	15	4837.537	32, 46, 57, 68, 80, 92, 104, 117, 131, 148, 169, 191, 210, 224, 236
	16	4839.226	30, 43, 54, 64, 74, 85, 96, 108, 120, 134, 151, 172, 193, 211, 225, 236
Bird	2	901.450	71, 122
	3	975.230	64, 111, 140
	4	1027.509	61, 104, 131, 164
	5	1051.482	54, 93, 119, 138, 169
	6	1067.992	47, 82, 108, 127, 142, 172
	7	1077.304	40, 70, 94, 113, 129, 143, 173
	8	1086.005	39, 69, 93, 112, 128, 141, 159, 192
	9	1091.341	37, 65, 88, 105, 119, 131, 142, 160, 193
	10	1095.355	37, 64, 86, 103, 117, 129, 139, 149, 167, 200
	11	1098.475	33, 56, 77, 93, 107, 119, 130, 140, 150, 169, 202
	12	1100.749	33, 56, 77, 93, 107, 119, 129, 138, 146, 158, 178, 209
	13	1102.639	31, 52, 71, 87, 100, 111, 121, 130, 139, 147, 159, 179, 210
	14	1104.132	29, 48, 66, 82, 95, 107, 118, 127, 134, 141, 149, 161, 181, 211
	15	1105.446	28, 45, 62, 77, 90, 101, 111, 120, 128, 135, 142, 150, 162, 182, 212
	16	1106.500	28, 45, 62, 77, 90, 101, 111, 120, 128, 135, 141, 148, 157, 171, 191, 219
Police	2	3647.353	74, 150
	3	3844.314	70, 135, 192
	4	3966.225	63, 112, 158, 209
	5	4013.875	61, 104, 140, 174, 214
	6	4047.198	32, 67, 106, 141, 175, 214
	7	4067.996	32, 67, 104, 133, 161, 186, 219
	8	4084.933	29, 52, 78, 106, 134, 162, 187, 219
	9	4094.705	29, 52, 78, 103, 125, 147, 169, 190, 220
	10	4101.702	29, 52, 78, 102, 123, 143, 165, 184, 203, 228
	11	4108.018	28, 49, 71, 90, 107, 126, 146, 166, 185, 204, 229
	12	4112.304	28, 49, 71, 89, 105, 122, 139, 157, 174, 189, 207, 231
	13	4115.165	28, 46, 62, 78, 92, 106, 123, 140, 158, 174, 189, 207, 231
	14	4117.926	28, 46, 62, 78, 91, 105, 120, 135, 151, 166, 180, 193, 210, 232
	15	4120.045	28, 46, 62, 78, 91, 103, 116, 129, 143, 158, 172, 185, 197, 214, 235
	16	4121.861	28, 46, 62, 78, 91, 103, 116, 128, 142, 156, 170, 182, 193, 206, 223, 240

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
	2	1073.452	75, 135
	3	1139.260	69, 101, 149
	4	1178.650	65, 92, 125, 176
	5	1203.749	56, 78, 100, 131, 179
	6	1218.643	47, 65, 85, 103, 133, 181
	7	1228.925	47, 64, 83, 100, 122, 152, 192
	8	1236.023	45, 59, 75, 90, 104, 125, 155, 194
Ostrich	9	1240.756	40, 52, 65, 80, 94, 107, 128, 157, 195
	10	1244.909	40, 52, 64, 78, 91, 103, 119, 141, 168, 201
	11	1247.879	40, 51, 62, 75, 87, 97, 108, 125, 148, 174, 205
	12	1250.313	37, 48, 57, 68, 80, 91, 101, 112, 128, 150, 175, 206
	13	1252.298	31, 44, 53, 63, 75, 86, 95, 105, 117, 134, 154, 178, 207
	14	1253.956	29, 42, 50, 58, 68, 79, 89, 98, 108, 120, 137, 157, 181, 209
	15	1255.366	29, 42, 50, 58, 67, 77, 86, 94, 102, 111, 124, 141, 160, 183, 210
	16	1256.490	28, 41, 49, 57, 66, 76, 85, 93, 101, 110, 122, 137, 155, 174, 196, 219
	2	7920.458	77, 180
	3	8117.991	54, 109, 193
	4	8203.807	42, 84, 131, 203
	5	8246.806	35, 68, 103, 146, 210
	6	8272.775	31, 59, 88, 120, 160, 216
	7	8287.714	28, 53, 77, 103, 132, 169, 220
	8	8298.322	27, 51, 75, 100, 128, 164, 212, 246
Viaduct	9	8308.240	24, 45, 66, 88, 112, 139, 172, 216, 247
	10	8315.133	22, 41, 60, 79, 99, 121, 146, 178, 219, 247
	11	8320.032	20, 37, 54, 71, 89, 108, 128, 152, 183, 221, 248
	12	8323.799	20, 36, 52, 68, 84, 101, 119, 139, 162, 190, 224, 248
	13	8326.793	18, 33, 48, 62, 77, 92, 108, 125, 144, 167, 195, 226, 248
	14	8329.119	17, 31, 44, 57, 70, 84, 98, 113, 129, 148, 170, 197, 227, 248
	15	8330.983	17, 31, 44, 57, 70, 84, 98, 113, 129, 147, 169, 195, 224, 243, 251
	16	8332.744	16, 29, 42, 54, 66, 78, 91, 104, 118, 133, 151, 172, 196, 224, 243, 251
	2	3593.389	64, 148
	3	3870.456	44, 104, 177
	4	3972.731	34, 81, 127, 188
	5	4024.885	27, 63, 101, 139, 194
	6	4054.836	24, 56, 90, 123, 156, 205
	7	4075.236	22, 49, 78, 107, 135, 168, 213
	8	4088.300	20, 44, 68, 93, 118, 142, 173, 216
Fish	9	4097.101	19, 40, 62, 85, 107, 128, 149, 178, 218
	10	4103.194	18, 38, 58, 78, 98, 118, 137, 158, 185, 222
	11	4107.954	16, 34, 51, 69, 88, 107, 125, 143, 163, 190, 224
	12	4111.678	15, 31, 47, 64, 82, 100, 117, 133, 149, 169, 195, 227
	13	4114.783	14, 28, 43, 58, 74, 90, 106, 121, 136, 152, 172, 198, 228
	14	4116.954	14, 28, 43, 58, 73, 88, 103, 117, 131, 145, 160, 179, 203, 231
	15	4118.931	13, 25, 38, 51, 64, 78, 92, 106, 120, 133, 147, 162, 181, 205, 232
	16	4120.522	13, 25, 37, 49, 62, 75, 89, 102, 115, 127, 139, 152, 168, 188, 211, 235

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
Houses	2	2543.788	56, 116
	3	2627.230	53, 105, 150
	4	2663.698	42, 69, 110, 152
	5	2694.902	39, 65, 96, 130, 158
	6	2708.655	39, 65, 95, 127, 150, 169
	7	2720.263	35, 55, 74, 98, 128, 151, 170
	8	2726.676	35, 55, 74, 98, 127, 148, 164, 184
	9	2732.853	32, 48, 65, 80, 101, 128, 148, 164, 184
	10	2736.510	30, 45, 60, 74, 87, 106, 129, 149, 164, 184
	11	2739.566	30, 45, 60, 73, 86, 105, 128, 145, 156, 168, 187
	12	2742.148	29, 42, 55, 68, 79, 94, 112, 130, 145, 156, 168, 187
	13	2744.071	28, 39, 51, 63, 74, 84, 98, 115, 131, 145, 156, 168, 187
	14	2745.542	28, 39, 51, 63, 74, 84, 98, 115, 131, 145, 155, 165, 176, 193
	15	2746.817	27, 37, 47, 58, 68, 77, 86, 100, 116, 131, 145, 155, 165, 176, 193
	16	2747.991	27, 37, 47, 57, 67, 76, 85, 98, 113, 127, 139, 147, 156, 165, 176, 193
Mushroom	2	1988.328	76, 145
	3	2153.037	65, 110, 174
	4	2237.441	59, 93, 135, 193
	5	2277.427	52, 78, 106, 145, 199
	6	2301.629	48, 71, 94, 122, 157, 205
	7	2317.538	46, 67, 87, 110, 138, 171, 213
	8	2328.526	44, 62, 79, 97, 118, 145, 177, 216
	9	2335.812	42, 58, 74, 90, 107, 127, 152, 181, 218
	10	2340.981	41, 56, 70, 84, 99, 116, 136, 159, 186, 220
	11	2345.160	39, 52, 65, 78, 92, 107, 124, 144, 166, 191, 223
	12	2348.413	38, 51, 63, 75, 87, 100, 115, 132, 152, 174, 199, 227
	13	2350.988	38, 50, 62, 74, 85, 97, 110, 125, 142, 160, 180, 203, 229
	14	2353.119	37, 48, 59, 69, 79, 89, 100, 113, 127, 144, 162, 181, 204, 230
	15	2354.748	36, 46, 56, 66, 75, 84, 94, 105, 117, 130, 146, 163, 182, 205, 230
	16	2356.113	36, 46, 56, 65, 74, 83, 92, 102, 113, 125, 139, 154, 169, 187, 208, 232
Snow mountain	2	1912.613	85, 149
	3	2135.274	79, 137, 197
	4	2234.669	70, 119, 154, 204
	5	2288.799	55, 96, 129, 160, 206
	6	2317.224	52, 90, 118, 142, 167, 209
	7	2333.078	50, 85, 111, 132, 153, 174, 212
	8	2344.629	45, 76, 100, 120, 139, 158, 177, 214
	9	2352.951	44, 74, 97, 116, 133, 150, 167, 187, 220
	10	2359.175	41, 68, 90, 108, 124, 140, 156, 172, 193, 225
	11	2364.304	38, 62, 84, 102, 117, 132, 146, 160, 175, 196, 227
	12	2367.971	34, 56, 78, 96, 111, 125, 139, 152, 165, 179, 199, 228
	13	2371.208	33, 53, 73, 90, 104, 117, 130, 143, 155, 167, 180, 200, 229
	14	2373.567	32, 52, 72, 89, 103, 115, 127, 139, 150, 161, 172, 185, 205, 232
	15	2375.586	31, 49, 68, 84, 98, 109, 120, 131, 142, 153, 164, 175, 188, 208, 233
	16	2377.255	26, 42, 59, 75, 89, 101, 111, 121, 132, 143, 154, 165, 176, 189, 209, 234

Table 6: Continued.

Image	n	Between-class variance (VTR)	Thresholds
Snake	2	1118.615	87, 134
	3	1231.320	76, 114, 154
	4	1286.555	69, 101, 129, 166
	5	1317.027	63, 91, 115, 140, 175
	6	1336.172	59, 84, 105, 126, 149, 182
	7	1348.933	55, 78, 97, 115, 133, 155, 187
	8	1357.665	52, 73, 91, 107, 123, 140, 161, 192
	9	1364.159	50, 70, 87, 102, 116, 131, 148, 169, 198
	10	1368.955	47, 65, 81, 95, 108, 121, 135, 151, 172, 200
	11	1372.626	46, 63, 78, 91, 103, 115, 127, 140, 156, 176, 203
	12	1375.513	45, 61, 75, 88, 99, 110, 121, 133, 146, 162, 182, 208
	13	1377.847	43, 58, 72, 84, 95, 106, 116, 127, 138, 151, 166, 185, 211
	14	1379.736	42, 56, 69, 81, 92, 102, 112, 122, 132, 143, 155, 170, 189, 214
	15	1381.282	41, 54, 66, 77, 87, 97, 106, 115, 124, 134, 145, 157, 172, 191, 215
	16	1382.603	40, 52, 63, 74, 84, 93, 102, 111, 120, 129, 139, 150, 162, 177, 195, 219
Riosanpablo	2	2667.020	95, 160
	3	2818.660	75, 121, 177
	4	2892.439	68, 102, 143, 189
	5	2931.654	62, 89, 121, 158, 197
	6	2957.018	58, 82, 107, 138, 171, 204
	7	2973.269	53, 74, 95, 119, 147, 177, 207
	8	2984.972	50, 69, 87, 108, 133, 159, 185, 212
	9	2993.359	48, 66, 83, 101, 122, 145, 168, 191, 215
	10	2999.615	46, 63, 78, 93, 110, 130, 151, 173, 195, 217
	11	3004.374	45, 61, 75, 89, 104, 121, 140, 160, 180, 200, 220
	12	3008.082	44, 59, 72, 84, 97, 112, 129, 147, 166, 185, 203, 222
	13	3011.101	43, 57, 69, 81, 93, 107, 122, 138, 155, 172, 189, 206, 224
	14	3013.465	41, 54, 66, 77, 88, 100, 113, 128, 144, 160, 176, 192, 208, 225
	15	3015.423	40, 53, 64, 74, 84, 95, 107, 121, 135, 150, 165, 180, 195, 210, 226
	16	3017.081	39, 51, 62, 72, 82, 92, 103, 115, 128, 142, 156, 170, 184, 198, 212, 227

method [35], which is a local search method that cannot guarantee an optimal solution. Thus, its solutions are inferior to the solution produced by the algorithm using a global search.

5.2.3. Convergence Rate Comparison. The number of generations (NG) is a measure used for the convergence rate comparisons. If the target value, VTR, is achieved in a lesser number of generations (NG), it means a faster convergence rate for the algorithm. Table 3 shows the average of $\overline{\mathrm{NG}}$ ($\overline{\mathrm{NG}}_{\mathrm{AM}}$) for each specific number of thresholds. The results of each algorithm are represented in the corresponding column's name. In each column, the cell containing $\overline{\mathrm{NG}}_{\mathrm{AM}}$ starts from the row associated with $n = 2$ until the row associated with $n = n_{\max}$. The cells associated with $n = n_{\max} + 1$ to the row associated with $n = 16$ are filled by $\mathrm{NA}(x)$. The second last row of the table is filled by the triple:

$$\left(n_{\max}, \mathrm{NA}\left(\text{number of unsuccessful subproblem}\right),\right.$$
$$\left.\mathrm{AM}\left(\overline{\mathrm{NG}}_{\mathrm{AM}} \text{ of } n = 2 \text{ until } n = n_{\max}\right)\right). \quad (23)$$

The algorithm with the highest $n_{\max}$, that is, the lowest number of unsuccessful subproblems and the lowest average of generation is the winner. The ranking of the algorithms depends on the ordering of (a_1, b_1, c_1) and (a_2, b_2, c_2) as follows.

First, rank on a_1 and a_2. Since both a_1 and a_2 are numeric, the higher value has the higher rank.

Second, rank on b_1 and b_2. If a_1 is 16, then b_1 must be NA(0). If a_1 is less than 16, then b_1 must be NA (number of unsuccessful subproblems) and b_2 has the same characteristics as b_1. Perform the order of the two numeric values in reverse; the lower value has the higher rank.

Third, rank on c_1 and c_2. They are numeric but the lower is better; perform the order of the two numeric values in reverse. The lower value has the higher rank.

When the ordering is finished, assign the numeric value of "1" to the object having the highest rank, assign the numeric value of "2" to the first runner up, and so on. These values are represented in the last row of the table.

If the row having $n = m$ must be ranked, it can be done in the same manner as above with some minor modifications. If m is less than $n_{\max}$, the values of the triple pair will be m, NA(0), $\overline{\mathrm{NG}}_{\mathrm{AM}}$. If m is greater than $n_{\max}$, the values of the triple pair will be ($n_{\max}$, NA(x), ∞). Thus the ranking can now be performed.

From the ranking results, see the second last row of Table 3; the convergence rate can be ranked from best to worst in the following order: $O(0.2)R$-DE, $O(0.1)R$-DE, $O(0.0)R$-DE, $O(\mathrm{rand})R$-DE, $O(0.3)R$-DE, $O(0.4)R$-DE, $O(0.5)R$-DE, $O(0.6)R$-DE, DE, $O(0.7)R$-DE, $O(0.8)R$-DE, rank-DE, $O(0.9)R$-DE, ABC, PSO, DPSO, FODPSO.

As can be seen from Table 3, the DE algorithm cannot complete the task when $n > 12$ and the rank-DE algorithm cannot complete the task when $n > 9$. Thus, rank-DE cannot compete with DE on searching for global multilevel thresholding.

In order for $O(\beta)R$-DE to outperform DE then β must be in the range of $[0.1, 0.6]$ or set to rndreal$[0, 1)$.

5.2.4. Stability Analysis. The harmonic mean of the success rate ($\mathrm{SR}_{\mathrm{HM}}$) for each specific number of thresholds was computed and is presented in Table 4. The results of each algorithm are represented in the corresponding column's name. The second row from the bottom shows the harmonic mean of the success rate of each algorithm for all threshold levels.

In each column, the cells containing $\mathrm{SR}_{\mathrm{HM}}$ start from the row associated with $n = 2$ to the row associated with $n = n_{0.5}$. The cells from the row associated with $n = n_{0.5} + 1$ to the row associated with $n = n_{\max}$ are filled by $\mathrm{SR}_{\mathrm{HM}}$. The cells from the row associated with $n = n_{\max}+1$ to the row associated with $n = 16$ are the cells that have $\mathrm{SR}_{\mathrm{HM}}$; these cells will be excluded from the comparison. The second last row of the table is filled with the triple:

$$\left(n_{0.5}, n_{\max}, \mathrm{HM}\left(\mathrm{SR}_{\mathrm{HM}} \text{ of } n = 2 \text{ until } n = n_{\max}\right)\right). \quad (24)$$

The algorithm with the highest $n_{0.5}$, the highest $n_{\max}$, and the highest average success rate is the winner. The ranking of the algorithms depends on the ordering of (a_1, b_1, c_1) and (a_2, b_2, c_2) as follows.

First, rank on a_1 and a_2.

Second, rank on b_1 and b_2.

Third, rank on c_1 and c_2.

Because they are numeric the higher value has the higher rank. When the ordering is finished, the numeric value of "1" is assigned to the object having the highest rank, the numeric value of "2" is assigned to the first runner up, and so on.

If the row having $n = m$ must be ranked, it can be done in the same manner as above with some minor modifications. If m is less than or equal to $n_{0.5}$, the values of the triple pair will be (m, m, $\mathrm{SR}_{\mathrm{HM}}$). If $n_{0.5} < m \le n_{\max}$, then the values of the triple pair will be ($n_{0.5}$, m, $\mathrm{SR}_{\mathrm{HM}}$). If m is greater than $n_{\max}$, the values of the triple pair will be ($n_{0.5}$, $n_{\max}$, $\mathrm{SR}_{\mathrm{HM}}$). Thus, the ranking can now be performed.

From the ranking results, see Table 4, the success rate can be ranked from best to worst in the following order: $O(0.0)R$-DE, $O(0.1)R$-DE, $O(0.2)R$-DE, $O(\mathrm{rand})R$-DE, $O(0.3)R$-DE, $O(0.5)R$-DE, $O(0.4)R$-DE, DE, $O(0.6)R$-DE, $O(0.7)R$-DE, $O(0.8)R$-DE, rank-DE, $O(0.9)R$-DE, ABC, PSO, DPSO, FODPSO, multithresh.

As can be seen from Table 4, the DE algorithm has an $\mathrm{SR}_{\mathrm{HM}} \ge 0.5$ until $n = 9$ and the rank-DE algorithm has an $\mathrm{SR}_{\mathrm{HM}} \ge 0.5$ until $n = 7$. This result confirms that rank-DE cannot compete with DE on searching for global multilevel thresholding. If the correct β is selected, the proposed algorithm can work very well. For $\beta \le 0.5$, $O(\beta)R$-DE has a higher rank than DE. The multithresh function cannot compete with any of the other algorithms. It can also be seen that the proposed $O(0.0)R$-DE algorithm has the best stability because its $\mathrm{SR}_{\mathrm{HM}}$ is greater than 0.5 when $n = 2$ to 16.

5.2.5. Reliability Comparison. The threshold value distortion a.k.a. TVD for each specific threshold is computed, shown in Table 5 and depicted in Figure 2. The results of each algorithm are illustrated in the corresponding column's name. The second last row of Table 5 shows the slope or the approximated growth rate, δ, of the TVD of each algorithm for all threshold levels. The δ is the slope of the robust linear regression computed by the Matlab function "robustfit." The lower slope exhibits the better reliability. The δ of each algorithm is sorted in descending order. From the results, the reliability can be ranked from best to worst in the following order: $O(0.0)R$-DE, $O(0.1)R$-DE, $O(0.2)R$-DE, $O(0.3)R$-DE, $O(0.4)R$-DE, $O(0.5)R$-DE, $O(\mathrm{rand})R$-DE, DE, $O(0.6)R$-DE, ABC, rank-DE, $O(0.7)R$-DE, $O(0.8)R$-DE, $O(0.9)R$-DE, DPSO, FODPSO, PSO, multithresh.

We can see from these results that rank-DE has a higher approximated TVD growth rate than DE. $O(\beta)R$-DE with $\beta \le 0.5$ and $O(\mathrm{rand})R$-DE are still better than DE. $O(0.0)R$-DE produced the best result with a very flat slope and a very low y-intercept. The multithresh function yielded a higher growth rate of solution distortion and a higher y-intercept than the other algorithms. The higher growth rate of solution distortion means the quality of solution drops very fast if the number of thresholds increases. The higher y-intercept means that the solution distortion at the lowest number of thresholds is high. Thus, the global optimization algorithm is required for solving the multilevel thresholding.

6. Conclusions

The differential evolution with onlooker (β) ranking-based mutation operator $O(\beta)R$-DE algorithm was proposed and applied to the multilevel image thresholding problem. The objective of this proposed algorithm was to increase the ability for adjusting the balance of the exploitation and exploration abilities. Its concept is a combination of the ranking-difference evolution and onlooker selection of the ABC algorithm. The experiments compared the proposed $O(\beta)R$-DE algorithm with six existing algorithms: PSO, DPSO, FODPSO, ABC, DE, and rank-DE on 20 real images of the Berkeley Segmentation Dataset and Benchmark and a satellite image. The stability analysis, convergence speed, and the reliability were measured. The results signified that the proposed $O(\beta)R$-DE algorithm is more efficient than the six tested algorithms. The onlooker ranking-based mutation operator is able to enhance the performance of the proposed algorithm. The $O(\beta)R$-DE not only obtained more stability analysis, but it also achieved faster convergence rates to reach the target BCV, if a proper value of β is set.

For future work based on this paper, the proposed $O(\beta)$R-DE algorithm has one parameter to be set by a user; the mechanism to automatically adapt this parameter is not presented but is required.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Higher Education Research Promotion and National Research University Project of Thailand, Office of the Higher Education Commission, through the Cluster of Research to Enhance the Quality of Basic Education.

References

[1] W. K. Pratt, *Digital Image Processing*, John Wiley & Sons, New York, NY, USA, 1978.

[2] J. S. Weszka, "A survey of threshold selection techniques," *Computer Graphics and Image Processing*, vol. 7, no. 2, pp. 259–265, 1978.

[3] K. S. Fu and J. K. Mui, "A survey on image segmentation," *Pattern Recognition*, vol. 13, no. 1, pp. 3–16, 1981.

[4] P. K. Sahoo, S. Soltani, and A. K. C. Wong, "A survey of thresholding techniques," *Computer Vision, Graphics and Image Processing*, vol. 41, no. 2, pp. 233–260, 1988.

[5] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277–1294, 1993.

[6] A. T. Abak, U. Baris, and B. Sankur, "The performance evaluation of thresholding algorithms for optical character recognition," in *Proceedings of the 4th International Conference on Document Analysis and Recognition*, pp. 697–700, Ulm, Germany, August 1997.

[7] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–168, 2004.

[8] S. U. Lee and S. Yoon Chung, "A comparative performance study of several global thresholding techniques for segmentation," *Computer Vision, Graphics and Image Processing*, vol. 52, no. 2, pp. 171–190, 1990.

[9] N. Otsu, "A threshold selection method from gray level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, Perth, Australia, December 1995.

[11] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," Tech. Rep. TR-95-012, International Computer Sciences Institute, Berkeley, Calif, USA, 1995.

[12] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[13] M. S. Couceiro, R. P. Rocha, N. M. F. Ferreira, and J. A. T. Machado, "Introducing the fractional-order Darwinian PSO," *Signal Image and Video Processing*, vol. 6, no. 3, pp. 343–350, 2012.

[14] R. V. Kulkarni and G. K. Venayagamoorthy, "Bio-inspired algorithms for autonomous deployment and localization of sensor nodes," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 40, no. 6, pp. 663–675, 2010.

[15] B. Akay, "A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding," *Applied Soft Computing Journal*, vol. 3, no. 6, pp. 3066–3091, 2013.

[16] K. Hammouche, M. Diaf, and P. Siarry, "A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 5, pp. 676–688, 2010.

[17] V. Osuna-Enciso, E. Cuevas, and H. Sossa, "A comparison of nature inspired algorithms for multi-threshold image segmentation," *Expert Systems with Applications*, vol. 40, no. 4, pp. 1213–1219, 2013.

[18] P. Ghamisia, M. S. Couceiro, F. Martins, and J. A. Benediktsson, "Multilevel image segmentation based on Fractional-Order Darwinian particle swarm optimization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 1, pp. 1–44, 2013.

[19] W. Gong and Z. Cai, "Differential evolution with ranking-based mutation operators," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2066–2081, 2013.

[20] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer, Berlin, Germany, 2005.

[21] R. Storn and K. Price, *Home Page of Differential Evolution*, International Computer Science Institute, Berkeley, Calif, USA, 2010.

[22] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 1785–1791, September 2005.

[23] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference*, pp. 485–492, July 2006.

[24] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[25] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 38, no. 1, pp. 218–237, 2008.

[26] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[27] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 107–125, 2008.

[28] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[29] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings of the 8th International Conference on Computer Vision*, vol. 2, pp. 416–423, July 2001.

[30] P. Ghamisi, M. S. Couceiro, J. A. Benediktsson, and N. M. F. Ferreira, "An efficient method for segmentation of image based on fractional calculus and natural selection," *Expert Systems with Applications*, vol. 39, pp. 12407–12417, 2012.

[31] R. Gamperle, S. D. Muller, and A. Koumoutsakos, "A parameter study for differential evolution," in *Proceedings of the Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, vol. 10, pp. 293–298, 2002.

[32] M. Saraswat, K. V. Arya, and H. Sharma, "Leukocyte segmentation in tissue images using differential evolution algorithm," *Swarm and Evolutionary Computation*, vol. 11, pp. 46–54, 2013.

[33] J. Kittler and J. Illingworth, "On threshold selection using clustering criteria," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 15, no. 5, pp. 652–655, 1985.

[34] R. Guo and S. M. Pandit, "Automatic threshold selection based on histogram modes and a discriminant criterion," *Machine Vision and Applications*, vol. 10, no. 5-6, pp. 331–338, 1998.

[35] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the Nelder-Mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998.