applied sciences

MDPI

Article

An Adaptive Bi-Mutation-Based Differential Evolution Algorithm for Multi-Threshold Image Segmentation

Yu Sun † **and Yingying Yang** *,†

School of Computer and Electronics and Information, Guangxi University, Nanning 530004, China; sunyu@gxu.edu.cn
* Correspondence: berlinyang1034@163.com
† These authors contributed equally to this work.

Abstract: Aiming at solving the problems of large calculation, time-consuming, and low segmentation accuracy of multi-threshold image segmentation, an adaptive threshold value based differential evolution algorithm is proposed in this paper. Firstly, an opposite learning strategy is introduced into the initial population to improve the quality of the initial population; secondly, a threshold-value-based mutation strategy is proposed to balance the exploration and development capabilities of the algorithm, and the number of successfully evolved individuals is considered as a threshold value to adaptively adjust the evolution of superior and inferior individuals. Experiments demonstrate that the proposed algorithm has better performance in enhancing accuracy and speeding up the convergence.

Keywords: differential evolution; OTSU; multi-threshold segmentation

check for updates

Citation: Sun, Y.; Yang, Y. An Adaptive Bi-Mutation-Based Differential Evolution Algorithm for Multi-Threshold Image Segmentation. *Appl. Sci.* **2022**, *12*, 5759. https://doi.org/10.3390/app12115759

Academic Editors: Erika Ottaviano, Jose Machado, Katarzyna Antosz, Dariusz Mazurkiewicz, Yi Ren, Pierluigi Rea, Rochdi El Abdi, Marina Ranga, Vijaya Kumar Manupati and Emilia Villani

Received: 20 April 2022
Accepted: 1 June 2022
Published: 6 June 2022

1. Introduction

Image segmentation is a classic problem in the field of computer vision, and it is also the main step of image analysis. Image segmentation refers to the segmentation of an image into several disjoint parts according to grayscale, color, spatial texture, geometric shapes, etc., so that the parts with similar characteristics in the same area are clearly different from the parts in different areas. It is widely used in machine vision, medical imaging, object detection and other fields [1]. Segmentation accuracy is an important indicator for detecting the quality of image segmentation, but in the process of research, the main challenge faced by many researchers is to find a perfect solution to obtain higher segmentation accuracy. At present, segmentation algorithms are mainly divided into threshold-based method, region growth-based method, histogram-based method and edge detection-based method and so on [2]. Among them, the threshold segmentation method is widely used in image processing due to its simplicity, efficiency, and fast segmentation speed. The use of threshold-based methods means that it is necessary to find an optimal threshold or a set of optimal threshold combinations [3]. The traditional image thresholding algorithm is effective for image binarization, but for multi-level image thresholding, the computational complexity of exhaustive search will increase significantly due to the increase in the number of thresholds, resulting in lower computational efficiency. Therefore, in order to reduce the computational cost, many research scholars integrate the meta-heuristic intelligent optimization algorithm such as Genetic Algorithm (GA) [4], Differential Evolution algorithm (DE) [5], Particle Swarm Optimization algorithm (PSO) [6], Artificial Bee Colony Algorithm (ABC) [7], etc., into the problem of multi-threshold segmentation of images. A better solution was obtained, which improved the accuracy of image segmentation to a certain extent and reduced the time complexity. Kurban et al. [8] took Kapur entropy as the objective function, and applied it to the task of color image segmentation using ABC, and achieved certain results in terms of computing efficiency and image segmentation quality. In [9], DE is employed for solving thresholding based image segmentation problem using Otsu objective function. Qi Qianhui et al. [10] studied several segmentation algorithms, and concluded that the threshold based segmentation method has a good segmentation performance on images with obvious peaks and valleys. Cerfa et al. [11] applied the fuzzy C-means (ARKFCM) clustering algorithm based on the adaptive regular kernel to the magnetic resonance (MR) brain image segmentation problem, initialized the sensitive cluster center value and combined the PSO algorithm to optimize the cluster centroid, and obtained more Good to achieve the segmentation effect. In the research of Wang et al. [12] they proposed an improved Otsu method and an improved GA for image segmentation. First, a morphological weighted adaptive algorithm is used to reduce morphological noise, and then the improved GA is used to optimize the largest Otsu globally. The image segmentation function is the between-class variance function to obtain the optimal image segmentation threshold. In fact, image segmentation based on the combination of threshold method and optimization algorithm can not only reduce the computational time complexity but also improve the segmentation accuracy and obtain higher segmentation quality pictures. After conducting a lot of research on the segmentation results, combining intelligent algorithms with Otsu's method seems to be a feasible and worthy method [3].

Therefore, in this article, DE is used in Otsu's method to improve the quality of image segmentation. Common threshold segmentation algorithms, including the maximum entropy method, balanced histogram threshold method, Otsu method, and k-means clustering, are widely used in industrial applications. Among them, the Otsu method is one of the most famous global threshold methods for binary segmentation problems, which was proposed by the Japanese scientist Otsu in 1979 [13]. The main idea of this method is to maximize the variance function between classes. Because of its strong adaptability and high segmentation accuracy, it is often used in the field of image segmentation. Due to the difference in image complexity, when single-threshold segmentation is extended to multi-threshold segmentation, the search space increases, and the time complexity and computational complexity increase accordingly.

2. Otsu Multi-Threshold Image Segmentation Principle

Multi-threshold image segmentation is mainly to find several optimal thresholds that can accurately segment the image into different segments.

The Otsu method takes the maximum variance function between classes as the objective function. It is the most commonly used method and has the advantages of simple application, simple calculation, and is easy to understand [14]. When solving the problem of image segmentation, this method eliminates the interference of contrast and brightness, thus realizing the effective separation of background and target information. With the inter-class variance between foreground and background becoming larger, the difference between the two parts also becomes larger. The threshold obtained by Otsu will be inclined to the type with a large variance when the difference between the inter-class variance of the two parts is large, some pixels with a large inter-class variance will be classified into those with small inter-class variance, resulting in incorrect segmentation. When the threshold value could maximize the inter-class variance function, then the probability of misclassification reaches the minimum.

The basic principle of Otsu is as follows:

Let $f(x, y)$ be a grayscale image with a pixel grayscale value ranging from 0 to $L-1$, where L represents the total number of gray levels, which represents the total number of pixels whose gray level value is equal to, then:

$$N = \sum_{i=0}^{L-1} n_i. \tag{1}$$

n_i represents the number of pixels with pixel value i in the image. N is the total number of pixels in the entire image. If p_i represents the probability of a pixel whose gray value is i, it can be approximately expressed as a frequency, which is defined as:

$$p_i = \frac{n_i}{N}. \tag{2}$$

If T is used to represent the threshold, then for the problem of binary image segmentation, pixels can be divided into two different parts, labeled as C_0 and C_1, where C_0 and C_1 contain pixels with different characteristics respectively, Threshold T divides the gray values into two ranges, and their gray values are within the range [0, T] and $[T+1, L-1]$, respectively. For the definition of variance between classes, several data should be obtained first, namely $\mu_0(T), \mu_1(T), \omega_0(T), \omega_1(T)$. The calculation formula for the above statistical data is as follows:

$$\mu_0(T) = \frac{\sum_{i=0}^{T} i p_i}{\omega_0(T)} \tag{3}$$

$$\omega_0(T) = \sum_{i=0}^{T} p_i \tag{4}$$

$$\mu_1(T) = \frac{\sum_{i=T+1}^{L} i p_i}{\omega_1(T)} \tag{5}$$

$$\omega_1(T) = \sum_{i=T+1}^{L} p_i \tag{6}$$

$$\mu = \frac{\sum_{i=0}^{L} i p_i}{\sum_{i=0}^{L} p_i} = \sum_{i=0}^{L} i p_i = \omega_0(T) \cdot \mu_0(T) + \omega_1(T) \cdot \mu_1(T). \tag{7}$$

In Formulas (3)–(7), $\mu_0(T)$ and $\mu_1(T)$ represent the gray value expectation of the background and foreground, respectively, $\omega_0(T)$ and $\omega_1(T)$ represent the sum of inter-class probability of background and foreground respectively. μ is the average gray value of all pixels in the entire image. Based on the above definition, the between-class variance function can be expressed as:

$$\sigma_B(T) = \omega_0(T)(\mu_0(T) - \mu)^2 + \omega_1(T)(\mu_1(T) - \mu)^2. \tag{8}$$

The formula consists of two parts, corresponding to C_0 and C_1.

In the Otsu method, when the between-class variance function obtains the maximum value with T^* as the threshold, T^* is used as the threshold. Therefore, the image segmentation problem can be transformed into an optimization problem, which can be described as:

For the multi-threshold image segmentation problem, the weighted sum of variance between categories is calculated. If the image has m categories to be classified, and the categories are divided by m–1 thresholds $(T_1, T_2, \ldots, T_{m-1})$, the objective function is expressed as follows:

$$\sigma(T_1, T_2, \ldots, T_{m-1}) = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2 + \ldots + \omega_{m-1}(\mu_{m-1} - \mu)^2. \tag{9}$$

The corresponding multi-threshold image segmentation problem can be denoted as:

$$T^* = argmax(\sigma(T_1, T_2, \ldots, T_{m-1})). \tag{10}$$

That is, it is necessary to find a set of $(T_1, T_2, \ldots, T_{m-1})$ to maximize the inter-class variance function.

3. Original DE Algorithm

Due to the disaster of dimension, the task becomes more challenging. When the dimension is increased, the search space expands exponentially and the distribution of good solutions is sparse. Differential evolution is an effective population-based meta-heuristic algorithm, which has been successfully used in various applications [15]. DE is based on three main operators, mutation, crossover, and selection. Mutation is the core operator in DE, which generates mutation vectors based on the linear combination of different candidate solutions. The role of the crossover operator is to combine the mutant vector with its target vector to produce a test vector. Finally, the selection operator adopts a greedy strategy to select and enter the formula here. Choose the better one of the test vector and the target vector.

Differential Evolution (DE) is a meta-heuristic algorithm proposed by Price and Storn in 1995. The algorithm has been used to solve multiple applications in various engineering fields. It is similar to other evolutionary algorithms (EA) used for optimization problems, and DE is a population-based algorithm that evolves through differences between individuals and achieves evolution through three operators: mutation, crossover, and selection. The execution of the three operators also contains three control parameters: population size NP, scaling factor F, and crossover rate CR.

Initialization: starting from the candidate solution randomly generated by NP, usually, a DE population is composed of NP individuals, denoted as:

$$x_{i,G} = (x_{1,G}, x_{2,G}, \ldots, x_{D,G}) \qquad i = 1, 2, \ldots, Np, \tag{11}$$

where Np is the population size. Np test individuals are generated from the current population through mutation, crossover, and selection of each generation. This step forms a loop until the termination condition is met. The initial population is randomly generated:

$$x_{i,j,G} = x^{L}_{i,j,G} + rand \times (x^{U}_{i,j,G} - x^{L}_{i,j,G}), \tag{12}$$

where $x_{i,j}$ represents the jth component of the ith individual; L and U in $x^{L}_{i,j}$ and $x^{U}_{i,j}$ represent the lower and upper bounds of the jth component respectively. $rand$ represents a random decimal in the range of [0, 1].

Mutation operator: a mutant individual is generated through the difference between two random individuals and F (scaling factor) and a target vector, DE achieves individual mutation through a differential strategy. This is also the embodiment of DE's central idea, which is based on the current individual in the population and randomly selects several vectors for differential operation and generates a differential v vector. Several commonly used differential mutation strategies are as follows:

(1) DE/rand/1

$$v_{i,G} = x_{r1,G} + F \times (x_{r2,G} - x_{r3,G}) \tag{13}$$

(2) DE/best/1

$$v_{i,G} = x_{best,G} + F \times (x_{r1,G} - x_{r2,G}) \tag{14}$$

(3) DE/current-to-best/1

$$v_{i,G} = x_{i,G} + F \times (x_{best,G} - x_{i,G}) + F \times (x_{r1,G} - x_{r2,G}) \tag{15}$$

(4) DE/current-to-rand/1

$$v_{i,G} = x_{i,G} + rand \times (x_{r1,G} - x_{i,G}) + F \times (x_{r2,G} - x_{r3,G}) \tag{16}$$

where r_1, r_2, and r_3 represent the index subscripts of three different individuals randomly selected from the population. F represents the scaling factor, which controls the strength of the difference. x_{best} represents the individual with the best fitness value.

Crossover operator: The purpose of the crossover is to generate a new candidate solution based on the combination of the mutation vector and the parent vector to generate a test vector. The crossover operation between the individual $x_{g,i}$ of the gth generation and the mutated intermediate individual $v_{g,i}$ can be denoted as:

$$u_{i,j,G} = \begin{cases} v_{i,j,G}, & if\ rand \leq CR\ or\ j = j_{rand} \\ x_{i,j,G}, & otherwise, \end{cases} \quad (17)$$

where j_{rand} is an integer randomly selected from 1, 2, ..., D to ensure that at least one individual is from a variant individual. *CR* is the crossover rate, which determines the ratio of inheritance from variant individuals to test individuals.

Selection operator: the selection operation is performed according to the greedy algorithm, and the individual with the optimal function value is selected from the test individual $u_{i,G}$ and the original individual $x_{i,G}$ to enter the next generation. When the problem is a minimization problem, then the individual with the smallest function fitness value will be chosen to replace the original individual and enter the next generation. The specific formula can be expressed as:

$$x_{i,G+1} = \begin{cases} u_{i,G}, & if\ f(u_{i,G}) \leq f(x_{i,G}) \\ x_{i,G}, & otherwise, \end{cases} \quad (18)$$

where $f(x)$ is the minimization function to be optimized and $x_{i,G+1}$ is the individual entering into the $G + 1$ generation for further evolution.

4. Opposition-Based Optimization after Initialization and Adaptive Threshold-Value-Based Bi-Mutation Strategy

4.1. Opposition-Based Optimization after Initialization

Opposite learning was first proposed by Tizhoosh for the initialization process of intelligent algorithms and subsequently was introduced to DE to find better individuals [16].

Evolutionary optimization methods usually try to improve the initial population toward optimal solutions. The generation of individuals in the initialization process embodies the traits of randomness. Although without prior information about the solution, we can improve our chance of starting with a fitter solution by simultaneously considering the opposite solution [17]. Through this operation, the better individual (random and the opposite of random) can be selected as an initial solution. There is a 50% possibility that the opposite individual is better than the initial solution, so starting with the individual that has the potential to accelerate convergence is quite sensible [17,18]. The opposition optimization is arranged after the initialization process. Supposing that $X = (x_1, x_2, x_3, ..., x_D)$ is a feasible solution to the maximum inter-class variance function and the opposite solution $\widehat{X} = (\widehat{x_1}, \widehat{x_2}, \widehat{x_3}, ..., \widehat{x_D})$ can be defined as follows:

$$\breve{x}_j = a_j + b_j - x_j, \quad (19)$$

where $X = (x_1, x_2, x_3, ..., x_D)$ is an individual in D dimensional space, and $x_1, x_2, x_3, ..., x_D \in R, x_j \in [a_j,\ b_j]$. For each initial individual, calculate the corresponding opposite individual, and the opposite population is generated after initialization.

4.2. Adaptive Threshold-Value-Based Bi-Mutation Strategy

In the evolution of DE, historical prior information often plays a crucial role in the subsequent evolution of the population, and can guide the evolutionary direction of the next generation to a certain extent.

Therefore, designing a mutation strategy that facilitates both exploration (discovering more global optimal regions) and exploitation (refining the accuracy of the solution in each global optimal region) is indispensable for DE-based multi-threshold image segmentation algorithms. For example, in the traditional DE algorithm, the mutation mechanism with high randomness focuses on exploration, while the greedy mutation mechanism guided by local optima focuses on exploitation. When using otsu to solve the multi-threshold image segmentation problem, it works better for images with unimodal between-class variance. However, when the inter-class variance between background and object is small, the multiple peaks of the inter-class variance function are relatively close, which makes it easy for the individual to fall into the local optimum. In this case, the application of a single mutation mechanism will show certain defects. Using mutation strategy with high diversity for sufficient exploration can greatly increase the running time of the algorithm, however, if the greedy mutation strategy guided by the local optima is used for rapid exploration, it would easily fall into local optima and the optimal solution will not be searched.

To address these shortcomings, a dual mutation strategy DE (named as historical adaptive threshold-based dual mutation strategy DE) is designed. The adoption of this strategy enables each individual to choose a suitable mutation strategy to meet its own needs of exploration and exploitation, which can not only better balance the diversity of the population to improve the accuracy of the solution, but also accelerate the convergence to save the search cost of the algorithm. In this way, the entire population can reach a state of balance between exploration and exploitation. The proposed mutation strategy divides the population of each generation into two parts by using the number of successful mutations in the previous generation as a threshold. The better part uses the individual as the mutation base vector, and combines with a group of differential vector to generate mutant individual, because the individual is superior in this generation, it is sensible to further seek surrounding feasible areas to speed up the exploitation process; while for the poor part, DE/rand/1 is used as the mutation operator so as to fully search the solution space and speed up exploration.

After sorting the whole population, the individuals with better fitness values are chosen and the idea of adaptive bi-mutation strategy is illustrated in Figure 1. The number of successful variant individuals—svi in the last generation—is considered as a threshold value to divide the individuals into superior individuals and inferior individuals [19]. The svi value can well reflect the evolution of the whole population. When fewer individuals are successfully mutated, the entire population is stagnated, and more individuals are required to perform the DE/rand/1 mutation strategy. Therefore, the small svi can exactly adjust the direction of population variation and meet the requirement of exploring better individuals. Instead, when there are quite a lot of individuals successfully mutated, which means the svi is large, the threshold-value-based mutation strategy can be executed to help exploit its neighborhood to speed up the convergence and to refine the accuracy of the solution.

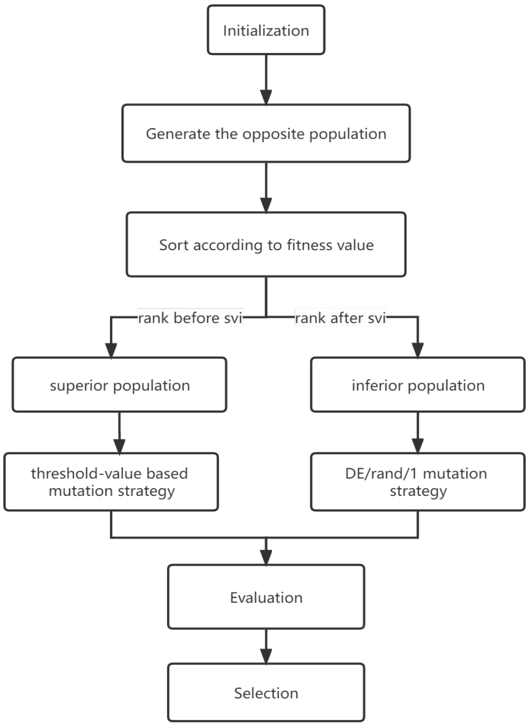

Initialization
Generate the opposite population
Sort according to fitness value
rank before svi
rank after svi
superior population
inferior population
threshold-value based mutation strategy
DE/rand/1 mutation strategy
Evaluation
Selection

Figure 1. Flowchart of Bi-mutation Strategy Framework.

Among the sorted population, individuals within the threshold value are considered as the superior individual, while individuals out of the threshold value are reckoned as inferior individuals. Meanwhile, the threshold value varies adaptively as the numbers of successful mutant variants in each generation are different. If the current individual belongs to the superior region, it should exploit further to accelerate the convergence speed and refine the accuracy of the solution. On the contrary, if the current individual belongs to the inferior individuals, then another strategy is conducted to maintain population diversity.

For superior individuals, the first mutation strategy—the adaptive historical threshold-value-based mutation strategy as shown in Formula (19)—should be conducted to enhance local searchability, while for inferior individuals, exploiting their neighborhoods is unnecessary but they can be utilized to explore more areas. The second mutation strategy—DE/rand/1—shows a good performance in diversity and is suitable for inferior individuals to ensure the population diversity:

$$v_i = \begin{cases} x_i + F \times (x_{r2} - x_{r3}) \\ x_{r1} + F \times (x_{r2} - x_{r3}), \end{cases} \tag{20}$$

where x_i is the current individual and used as base vector, as explained before, to further exploit the individuals around the splendid individuals. x_{r2} and x_{r3} are the random individuals different from the current and best individuals. F is the scaling factor.

The detailed procedure of the adaptive threshold-value-based bi-mutation strategy is presented as follows:

- *step1:* Randomly generate the initial populations.
- *step2:* For each individual, calculate the corresponding opposite individual and generate the opposite population;
- *step3:* Calculate the fitness values of the whole initial population and opposite population;
- *step4:* Replace the initial individual with its opposite individual if the opposite individual is better than initial individual;
- *step5:* Sort the population in descending order according to fitness value;
- *step6:* The svi value is considered as a threshold to divide the whole population into two groups and respectively execute different mutation strategies;
- *step7:* For superior individuals, execute threshold-value-based mutation strategy, and for inferior individuals, execute DE/rand/1 strategy;
- *step8:* Execute crossover and selection operation.

4.3. Experiment Results and Analysis

4.3.1. Evaluation Criterion

SSIM and PSNR are used in our experiment to evaluate the effectiveness of the segmentation algorithm. Both of those criteria follow the rule that the higher the value, the better the image quality. As it is easy to extract structural information from images with human vision, then the structural similarity is practicably computed to appraise the quality of the image after being segmented. The range of SSIM is 0 to 1, and if the same two images are measured, the SSIM value of those two images equals 1. The formula is as follows:

$$SSIM = \frac{(2\mu_x\mu_y + C_1)(\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{21}$$

where μ_x and μ_y is the mean value of x and y respectively, μ_x^2 and μ_y^2 are the variance of x and y respectively, and σ_{xy} represents the covariance between x and y, C_1 and C_2 are two constants for stabilization denoted as $C_1 = (k_1L)^2$, $C_2 = (k_2L)^2$, among which $k_1 = 0.01$, $k_2 = 0.03$, and L refers to the dynamic range of the pixel value and is usually equal to 255.

PSNR is Peak-Signal-to-Noise-Ratio, which is also the image objective evaluation index. It is the most commonly and widely used criteria and the unit of the measurement of *PSNR* is decibels (DB). The formula is as follows:

$$PSNR = 10log_{10}\frac{255^2}{MSE} \tag{22}$$

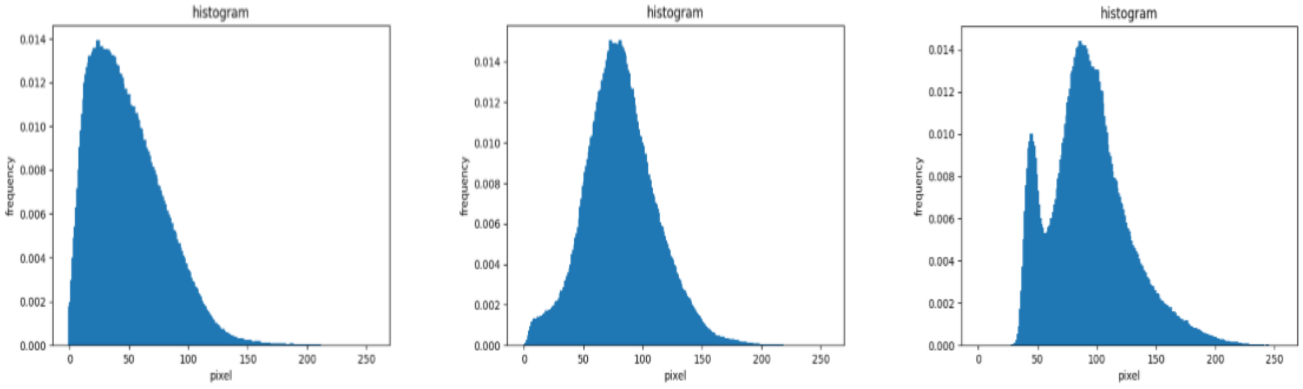
$$MSE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(f'(i,\ j) - f(i,\ j)\right)^2}{M \times N}, \tag{23}$$

where $f'(i,\ j)$ and $f(i,\ j)$ refer to the image to be evaluated and the original image respectively, M and N represent the length and width of the image. MSE in image segmentation indicates the mean value of the sum of squares of the difference of pixel values between the segmented image and the original one.

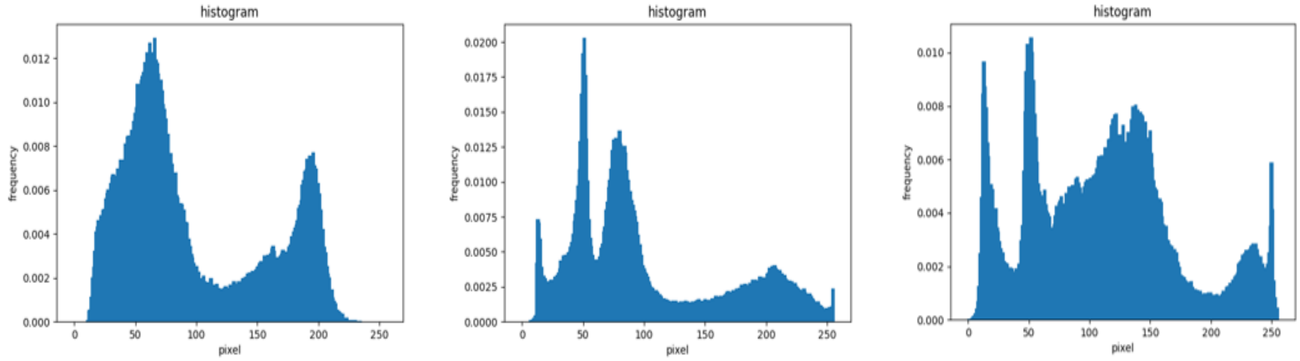
4.3.2. Contrast Experiments for High-Dimensional Segmentation

In the DE algorithm, the scaling factor can adjust the variation interval range of the population. As in image segmentation, the gray value of an image can only be integers and there are only 256 possibilities to vary, therefore, a larger F is not friendly for individuals to attain suitable mutation results as thresholds are prone to be out of bounds. The experiment shows that when F is set as 0.5, the optimal threshold can be obtained with relatively less iteration compared to other settings and CR is the crossover rate and is set as 0.9; empirical experience shows that high crossover can decrease the time.

In this experiment, the Berkeley segmentation dataset image was adopted as the experimental dataset, in which images 1–3 (forest, kangaroo, actinia) are a typical image with relatively concentrated image gray value distribution and small inter-class variance, and their corresponding histograms are shown in Figure 2, while images 4–6 (numbered 41033, 385039, 145086) have a more obvious difference in the image gray value distribution, that is, the inter-class variance is relatively large. For typical images, this paper segments the image through three higher dimensions when D = 8, 9, 10 to test the segmentation effect of the image. The time cost of the algorithm is concerned with the real-time problem, and it is an important standard to evaluate the performance of algorithms. As most practical problems are related to multi-threshold segmentation, which means that multiple parts need to be segmented, the multi-threshold image segmentation problem has high computational complexity and long time consumption. Thus, under the same condition, the time cost is not stable but fluctuated. In our experiment, with the same hardware configuration, the iteration time of DE is set as 1000 to sufficiently evolve, run the same image segmentation algorithm 50 times and take its average running time as an objective evaluation standard. The results of our proposed DE on three typical images with respect to the evaluation criterion including fitness value, PSNR, SSIM, iteration times and running time are given in Tables 1–6. Two of five algorithms that perform better on the evaluation criteria are marked in bold. It can be seen from the tables that every criterion of our method is marked in bold, which means that our method outperforms other competitive methods nearly in overall performance. The comparison of every single indicator is shown in Figures 2–4.



Histogram of Forest Histogram of Kangaroo Histogram of Actinia



Histogram of image 41033 Histogram of image 385039 Histogram of image 145086

Figure 2. Histogram distribution.

Table 1. Results comparison of five algorithms for segmentation of images 1–3 on different criteria when D = 8.

D = 8	Algorithm	Evaluation Criterion				
		Fitness Value	PSNR	SSIM	Iterations	Running Time
Image1(Forest)	DE(DE/rand/1)	924.2405	30.0992	0.8936	114.06	2.9308
	DE(DE/best/1)	922.4241	29.8563	0.8845	**30.96**	**1.0445**
	CSDE	**924.3083**	**30.1454**	**0.8956**	327.36	7.5639
	VNCDE	924.1787	30.1439	0.89501	130.32	45.8256
	Our Method	**924.3082**	**30.1521**	**0.8954**	**99.32**	**2.6921**
Image2(Kangaroo)	DE(DE/rand/1)	889.8179	29.8395	0.8769	107.48	2.7631
	DE(DE/best/1)	887.5622	29.6249	0.867	**31.26**	**0.9713**
	CSDE	**889.9062**	**29.8681**	**0.8785**	328.68	8.6718
	VNCDE	889.7553	29.8422	0.8778	152.12	53.0893
	Our Method	**889.9043**	**29.8668**	**0.8783**	**95.3**	**2.5922**
Image3(Actinia)	DE(DE/rand/1)	1116.6892	29.4944	0.7632	105.12	3.5271
	DE(DE/best/1)	1113.7184	29.2255	0.7598	**31.06**	**0.9773**
	CSDE	**1116.703**	**29.4996**	**0.7638**	304.02	7.1308
	VNCDE	1116.5651	29.4879	0.764	235.86	81.6603
	Our Method	**1116.7014**	**29.4959**	**0.7636**	**97.24**	**2.8167**

Table 2. Results comparison of five algorithms for segmentation of images 1–3 on different criteria when D = 9.

D = 9	**Algorithm**	**Evaluation Criterion**				
		Fitness Value	**PSNR**	**SSIM**	**Iterations**	**Running Time**
Image1(Forest)	DE(DE/rand/1)	928.3622	30.5839	0.9102	135.28	4.2656
	DE(DE/best/1)	925.5307	30.1935	0.8938	**33.22**	**1.0158**
	CSDE	**928.4816**	**30.6411**	**0.9132**	369.8	8.9023
	VNCDE	928.1165	30.5994	0.9129	134.14	54.7727
	Our Method	**928.477**	**30.6273**	**0.9139**	**119.2**	**3.18**
Image2(Kangaroo)	DE(DE/rand/1)	894.7103	30.4694	0.8953	132.66	4.14
	DE(DE/best/1)	891.3213	29.9178	0.8813	**33.66**	**1.1719**
	CSDE	**894.8415**	**30.5598**	**0.8972**	384.76	8.3398
	VNCDE	894.7194	30.4893	0.8969	161.24	59.9547
	Our Method	**894.8188**	**30.5214**	**0.8967**	**115.98**	**3.2098**
Image3(Actinia)	DE(DE/rand/1)	1121.1749	29.8381	0.7793	124.36	3.7242
	DE(DE/best/1)	1118.2762	29.5798	**0.782**	**33.36**	**1.0423**
	CSDE	**1121.2952**	**29.8734**	0.7808	378.92	8.1109
	VNCDE	1120.91	29.8418	0.7809	290.74	107.641
	Our Method	**1121.2713**	**29.8735**	**0.7835**	**116.14**	**3.6367**

Table 3. Results comparison of five algorithms for segmentation of images 1–3 on different criteria when D = 10.

D = 10	**Algorithm**	**Evaluation Criterion**				
		Fitness Value	**PSNR**	**SSIM**	**Iterations**	**Running Time**
Image1(Forest)	DE(DE/rand/1)	931.1964	30.9917	0.9187	165.72	5.0252
	DE(DE/best/1)	927.9442	30.466	0.9057	**34.46**	**1.0486**
	CSDE	**931.541**	**31.2247**	**0.9242**	414.88	9.6212
	VNCDE	931.3505	31.2011	0.9245	149.26	60.1333
	Our Method	**931.5102**	**31.2395**	**0.9246**	**132.4**	**3.776**
Image2(Kangaroo)	DE(DE/rand/1)	898.2523	31.0023	0.9084	154.12	4.7712
	DE(DE/best/1)	895.4196	30.3822	0.8963	**36.68**	**1.1319**
	CSDE	**898.481**	**31.1343**	**0.9115**	430.58	9.5338
	VNCDE	897.977	31.0215	0.9101	201.24	79.1909
	Our Method	**898.4808**	**31.1409**	**0.9119**	**125.68**	**3.4951**
Image3(Actinia)	DE(DE/rand/1)	1124.6503	30.4593	0.799	148.24	3.9829
	DE(DE/best/1)	1121.1936	29.8745	0.8017	**34.26**	**0.8487**
	CSDE	**1124.7492**	**30.5089**	**0.8008**	459.64	8.4798
	VNCDE	1124.2897	30.4549	0.8092	338.84	125.4491
	Our Method	**1124.7372**	**30.5241**	**0.8006**	**140.82**	**3.6164**

Table 4. Results comparison of five algorithms for segmentation of images 4–6 on different criteria when D = 8.

D = 8	**Algorithm**	**Evaluation Criterion**				
		Fitness Value	**PSNR**	**SSIM**	**Iterations**	**Running Time**
Image1(41033)	DE(DE/rand/1)	3434.6711	29.0786	0.7675	100.8667	**2.6089**
	DE(DE/best/1)	3433.6338	**29.1543**	**0.7683**	**31.2333**	**0.8624**
	CSDE	**3434.6965**	29.0487	0.7587	366.7333	7.6867
	VNCDE	3434.6609	29.0267	0.7634	120.432	65.3453
	Our Method	**3434.6894**	**29.1625**	**0.7686**	**98.8**	2.7308
Image2(385039)	DE(DE/rand/1)	**4030.9421**	**28.5971**	**0.8249**	**84.4333**	4.8844
	DE(DE/best/1)	4030.9409	28.5958	0.8242	88.3333	**3.3523**
	CSDE	4030.9397	28.594	0.8233	88.1333	7.5729
	VNCDE	4030.9132	28.4387	0.8078	167.9087	70.4665
	Our Method	**4030.9418**	**28.5996**	**0.8254**	**84.7667**	**2.3907**
Image3(145086)	DE(DE/rand/1)	3725.232	28.5494	0.7921	94.76	4.5831
	DE(DE/best/1)	3724.4946	28.5059	0.7916	**28.54**	**1.5662**
	CSDE	**3725.2315**	**28.5514**	**0.7922**	275.32	10.5008
	VNCDE	3724.9765	28.5439	0.7786	242.579	86.7562
	Our Method	**3725.2336**	**28.5533**	**0.7922**	**89.24**	**4.3692**

Table 5. Results comparison of five algorithms for segmentation of images 4–6 on different criteria when D = 9.

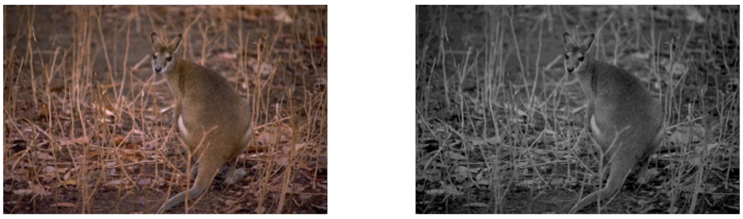
D = 9	**Algorithm**	**Evaluation Criterion**				
		Fitness Value	**PSNR**	**SSIM**	**Iterations**	**Running Time**
Image4(41033)	DE(DE/rand/1)	3441.5956	29.3233	0.7951	107.9333	**2.9114**
	DE(DE/best/1)	3439.5199	29.2966	0.7889	**31.1333**	**0.8228**
	CSDE	**3441.6141**	**29.3404**	**0.7969**	333.4	6.212
	VNCDE	3441.543	29.2994	0.7129	141.25	62.6528
	Our Method	**3441.5978**	**29.3406**	**0.7959**	**107.6333**	3.033
Image5(385039)	DE(DE/rand/1)	4038.7195	28.7989	0.8417	**106.2**	3.0852
	DE(DE/best/1)	4038.3371	**28.9104**	**0.8461**	28.7333	**0.7629**
	CSDE	**4038.7681**	28.7269	0.8376	500.7	11.3685
	VNCDE	4038.5453	28.6893	0.8369	153.5	69.7843
	Our Method	**4038.7209**	**28.8138**	**0.8424**	**104.2667**	**2.88**
Image6(145086)	DE(DE/rand/1)	3734.052	28.6686	0.8021	109.38	**4.9802**
	DE(DE/best/1)	3733.1225	28.6688	0.8035	**106.2**	**4.8643**
	CSDE	**3734.1269**	**28.6798**	**0.8041**	357.32	11.8959
	VNCDE	3734.1235	28.6701	0.7809	290.74	107.641
	Our Method	**3734.1244**	**28.6704**	**0.8039**	**108**	5.9049

Table 6. Results comparison of five algorithms for segmentation of images 4–6 on different criteria when D = 10.

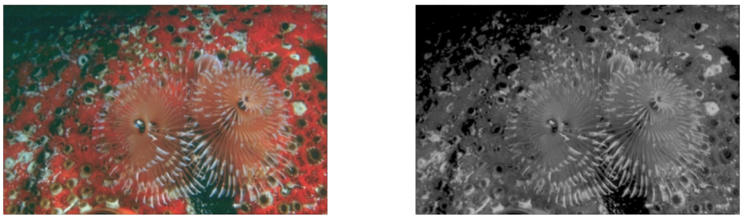
D = 10	**Algorithm**	**Evaluation Criterion**				
		Fitness Value	**PSNR**	**SSIM**	**Iterations**	**Running Time**
Image1(Forest)	DE(DE/rand/1)	3446.1561	29.7778	0.8202	**135.0667**	**3.8724**
	DE(DE/best/1)	3444.6259	29.5565	0.8127	**34.2333**	**0.883**
	CSDE	**3446.1878**	**29.7858**	**0.8226**	489.8	10.0595
	VNCDE	3446.1678	29.7658	0.8206	207.0667	80.7313
	Our Method	**3446.1771**	**29.8002**	**0.8213**	136.9333	4.2332
Image2(Kangaroo)	DE(DE/rand/1)	4046.1617	29.2401	0.8507	119.6	**3.231**
	DE(DE/best/1)	**4045.2943**	**29.271**	0.8628	**31.2667**	**0.9824**
	CSDE	4046.2345	29.2679	0.86	360.9333	7.7758
	VNCDE	4046.2064	29.2415	**0.9101**	201.24	76.1909
	Our Method	**4046.2481**	**29.2975**	**0.8677**	**114.3333**	3.248
Image3(Actinia)	DE(DE/rand/1)	3740.8882	28.9319	0.8066	125.12	**5.4697**
	DE(DE/best/1)	3739.8085	28.9277	**0.8217**	**31.46**	**1.3898**
	CSDE	**3741.0143**	**28.9404**	0.8155	387.22	14.4902
	VNCDE	3740.9982	28.9334	0.8092	338.84	125.4491
	Our Method	**3741.0105**	**28.9376**	**0.8156**	**122.98**	5.576

To further evaluate the performance of the segmentation of our proposed algorithm, we compared the proposed algorithm with the original DE, DE with DE/best/1 operator, CSDE [20], and VNCDE [21] on several evaluation criteria. The results of these algorithms are obtained under different dimensions. It can be seen from Table 1 that our proposed method presents a relatively better performance in terms of computational speed and segmentation effect. A forest image, kangaroo image and actinia image were selected as experimental images to testify the segmentation effect and the computational efficiency of the OTSU algorithm. Taking "Forest picture" as an example, the PSNR value of the image is 30.1521, which obviously precedes other comparative algorithms. The SSIM value of the image is 0.8947, which is basically the same as other algorithms, indicating the low image distortion, and the quality of the image after segmentation is guaranteed. On the premise of maintaining the segmentation effect, the running time of our proposed method is 2.8 times shorter than that of the CSDE algorithm, reduced from 7.5639 to 2.6921, which decreased by more than 60% and greatly improved the running speed. Iteration times were 108.32, although this cannot compare with DE using the DE/best/1 strategy but is better than the original DE and CSDE, which reflects the fast convergence of the algorithm to a certain extent. Analysis of the other two images also reached a similar conclusion. Regarding the experiment of the second group of pictures, under the same experimental settings as the first group of pictures, it can also be seen that although the DE algorithm using DE/best/1 has a better performance in terms of time, the accuracy of the algorithm is relatively low. In the case of ensuring high precision, our proposed algorithm has obvious advantages in comprehensive performance. The image after segmentation is shown in Figures 3 and 4.

a. forest a. segmented forest

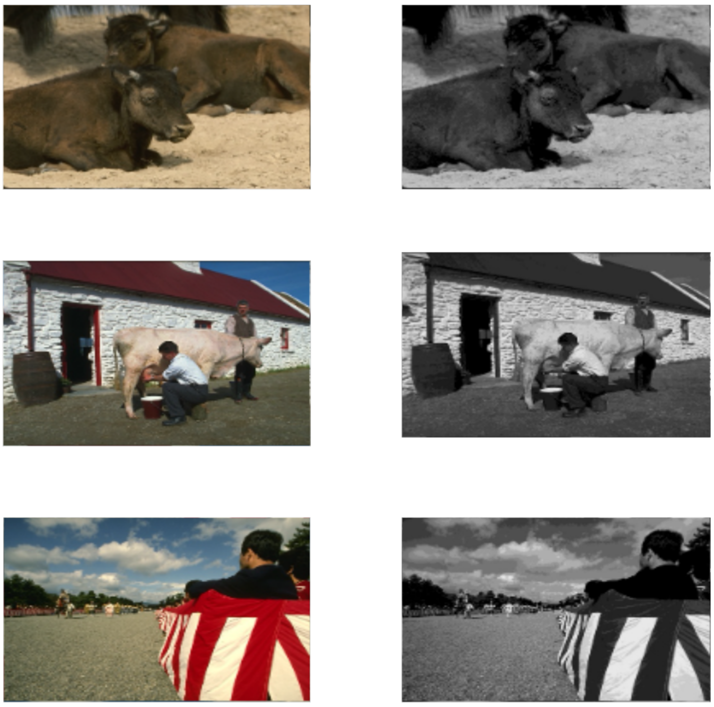


b. kangaroo b. segmented kangaroo



c. actinia c. segmented actinia

Figure 3. Group1: Original Images and Segmented Images on D = 10.



original image segmented image

Figure 4. Group2: Original Images and Segmented Images on D = 10.

5. Conclusions

This paper aims to solve problems in traditional OTSU when tackling high dimensional image segmentation, and a probability-based bi-mutation strategy is proposed in DE, which achieves a better performance in refining the objective function and thus improves the quality of the image being segmented. According to maximal variance between classes, the proposed strategy can effectively improve the calculation speed and segmentation efficiency. The experiment results indicate that the DE with the proposed strategy can not only outperform other contrast algorithms in terms of time cost, but also excels in segmentation quality.

Author Contributions: Conceptualization, Y.S. and Y.Y.; methodology, Y.Y.; validation, Y.S., Y.Y.; formal analysis, Y.Y.; investigation, Y.Y.; resources, Y.Y.; data curation, Y.Y.; writing—original draft preparation, Y.Y.; writing—review and editing, Y.Y. and Y.S.; visualization, Y.Y.; project administration, Y.S.; funding acquisition, Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China (Grant Nos. 61763002 and 62072124), Guangxi Major projects of science and technology (Grants No. 2020AA21077007).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DE	Differential Evolution
EA	Evolution Algorithm
PSNR	Peak-Signal-to-Noise-Ratio
SSIM	Structural Similarity
CSDE	Combined Strategy Differential Evolution
VNCDE	Parameter-free Voronoi Neighborhood for Evolutionary Multimodal Optimization

References

1. Kang, W.X.; Yang, Q.Q.; Liang, R.P. The Comparative Research on Image Segmentation Algorithms. In Proceedings of the International Workshop on Education Technology & Computer Science, Wuhan, China, 7–8 March 2009.
2. Cheng, Y.; Li, B. Image Segmentation Technology and Its Application in Digital Image Processing. In Proceedings of the 2021 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC), Dalian, China, 14–16 April 2021; pp. 1174–1177. [CrossRef]
3. Ye, Z.; Zhang, A.; Cao, Y.; Ma, L.; Hu, J. An Image Thresholding Method Based on Differential Evolution Algorithm and Genetic Algorithm. In Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Metz, France, 18–21 September 2019.
4. Mao, X.Y. Application of Genetic Algorithm in Image Segmentation. *J. Proj. Guid.* **2004**, *3*, 186–188. [CrossRef]
5. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
6. Liu, Y.; Mu, C.; Kou, W.; Liu, J. Modified particle swarm optimization-based multilevel thresholding for image segmentation. *Soft Comput.* **2015**, *19*, 1311–1327. [CrossRef]
7. Gao, H.; Fu, Z.; Pun, C.M.; Hu, H.; Lan, R. A Multi-Level Thresholding Image Segmentation Based on an Improved Artificial Bee Colony Algorithm. *Comput. Electr. Eng.* **2018**, *70*, 931–938. [CrossRef]
8. Kurban, T.; Civicioglu, P.; Kurban, R.; Besdok, E. Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding. *Appl. Soft Comput.* **2014**, *23*, 128–143. [CrossRef]
9. Charansiriphaisan, K.; Chiewchanwattana, S.; Sunat, K. A Global Multilevel Thresholding Using Differential Evolution Approach. *Math. Probl. Eng.* **2014**, *2014*, 1–23. [CrossRef]
10. Qianhui, Q.I.; Tian, Y.; Han, L.; Zhang, T. Research on Image Segmentation Algorithms. *J. Beijing Inst. Graph. Commun.* **2019**, *27*, 102–106.
11. Forouzanfar, M.; Forghani, N.; Teshnehlab, M. Parameter optimization of improved fuzzy c-means clustering algorithm for brain MR image segmentation. *Eng. Appl. Artif. Intell.* **2010**, *23*, 160–168. [CrossRef]
12. Wang, Y. Improved OTSU and Adaptive Genetic Algorithm for Infrared Image Segmentation. In Proceedings of the 30th China Conference on Control and Decision Making, Shenyang, China, 9–11 June 2018.

13. Ostu, N.; Nobuyuki, O.; Otsu, N. A thresholding selection method from gray level histogram. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66.
14. Luo, J.; Yang, Y.; Shi, B. Multi-threshold Image Segmentation of 2D Otsu Based on Improved Adaptive Differential Evolution Algorithm. *J. Electron. Inf. Technol.* **2019**, *41*, 2017–2024.
15. Mousavirad, S.J.; Rahnamayan, S.; Schaefer, G. Many-level Image Thresholding using a Center-Based Differential Evolution Algorithm. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020.
16. Tizhoosh, H.R. Opposition-Based Learning: A New Scheme for Machine Intelligence. In Proceedings of the International Conference on International Conference on Computational Intelligence for Modelling, Control & Automation, Vienna, Austria, 28–30 November 2005; pp. 695–701.
17. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M.A. Opposition-Based Differential Evolution. *IEEE Trans. Evol. Comput.* **2008**, *12*, 64–79. [CrossRef]
18. Zhou, X.Y.; Zhi-Jian, W.U.; Wang, H. A Differential Evolution Algorithm Using Elite Opposition-based Learning. *J. Chin. Comput. Syst.* **2013**, *34*, 2129–2134.
19. Wang, Z.J.; Zhan, Z.H.; Lin, Y.; Yu, W.J.; Yuan, H.Q.; Gu, T.L.; Kwong, S.; Zhang, J. Dual-Strategy Differential Evolution with Affinity Propagation Clustering for Multimodal Optimization Problems. *IEEE Trans. Evol. Comput.* **2017**, *22*, 894–908. [CrossRef]
20. Huang, C.; Li, X.; Wen, Y. AN OTSU image segmentation based on fruitfly optimization algorithm. *AEJ Alex. Eng. J.* **2020**, *60*, 183–188. [CrossRef]
21. Zhang, Y.H.; Gong, Y.J.; Gao, Y.; Wang, H.; Zhang, J. Parameter-Free Voronoi Neighborhood for Evolutionary Multimodal Optimization. *IEEE Trans. Evol. Comput.* **2020**, *24*, 335–349. [CrossRef]