applied sciences

MDPI

Article

Using an Artificial *Physarum polycephalum* Colony for Threshold Image Segmentation

Zhengying Cai *, Gengze Li, Jinming Zhang and Shasha Xiong

Hubei Province Engineering Technology Research Center for Construction Quality Testing Equipments, College of Computer and Information Technology, China Three Gorges University, Yichang 443002, China
* Correspondence: master_cai@163.com

Featured Application: Image segmentation can be applied to image recognition and computer vision as the most important preprocess, and theartificial *Physarum polycephalum* colony algorithm proposed in this article can also be used to solveall kinds of complex problems in image processing.

Abstract: Traditional artificial intelligence algorithms are prone to falling into local optima when solving threshold segmentation problems. Here, a novel artificial *Physarum polycephalum* colony algorithm is proposed to help us solve the difficult problem. First, the algorithm methodology of an artificial *Physarum polycephalum* colony algorithm is described to search for the optimal solutions by expansion and contraction of a lot of artificial hyphae. Different artificial *Physarum polycephalum* can learn from each other and produce more hyphae in expansion. In contraction, the artificial *Physarum polycephalum* colony can select the best hyphae with high fitness through a quick sort algorithm, but the other hyphae with low fitness will be absorbed and disappear. Second, a fitness function is modeled based on Kapur's entropy for the proposed artificial *Physarum polycephalum* colony algorithm to search for optimal threshold segmentation solutions. Third, a series of benchmark experiments are implemented to test the proposed artificial *Physarum polycephalum* colony algorithm, and some state-of-the-art approaches are employed for comparison. The experimental results verified that the proposed algorithm can obtain better accuracy and convergence speed, and is not easier to fall into the local optimal solution too early.

Keywords: threshold image segmentation; image processing; artificial intelligence; artificial *Physarum polycephalum* colony algorithm

check for updates

Citation: Cai, Z.; Li, G.; Zhang, J.; Xiong, S. Using an Artificial *Physarum polycephalum* Colony for Threshold Image Segmentation. *Appl. Sci.* **2023**, *13*, 11976. https://doi.org/10.3390/app132111976

Academic Editor: João M. F. Rodrigues

Received: 12 August 2023
Revised: 21 October 2023
Accepted: 30 October 2023
Published: 2 November 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Image segmentation is a key step from image processing to image analysis. Among them, threshold segmentation is the most commonly used parallel region technique and also the category with the highest number of applications in image segmentation. Threshold image segmentation employs the image gray-level histogram and one or more thresholds to segment an image [1]. This method compares the gray values between the target and the background on an image and selects appropriate thresholds to implement image segmentation where it judges each pixel belonging to the target region or the background region [2]. Now, the threshold image segmentation method is the most widely used and has become an essential link in image processing, i.e., COVID-19 diagnosis [2], satellite monitoring [3], remote sensing analysis [4], fault diagnosis [5], carotid artery surgery [6], prevention and control of maize diseases [7], CT scan [8], etc. Peak signal-to-noise ratio (PSNR) and feature similarity (FSIM) are often used indexes for performance evaluation of threshold image segmentation.

The selection of a threshold is important for threshold segmentation methods since only accurate threshold selection can ensure the accuracy of threshold image segmentation [9]. Recently, more and more scholars are getting involved in this area and trying to find

the optimal threshold. For a given image, the optimal threshold can often be determined by analyzing its histogram. For example, when the histogram clearly shows a bimodal pattern, the midpoint of the two peaks can be selected as the optimal threshold [10]. According to the dimension, the image histogram can be classified into one-dimensional (1-D) histogram, two-dimensional (2-D) histogram, three-dimensional (3-D) histogram [3,11], and so on. Because of the complexity of images and the uncertainty of noise interference, determining the appropriate threshold through histograms is still not an easy task [12].

Due to the difficulty of manually selecting thresholds, many scholars have begun to use artificial intelligence algorithms to help us determine appropriate thresholds and many algorithms are verified to have great potential in the field of threshold image segmentation [6,9]. Among them, swarm intelligence is a hotspot that simulates the behaviors of a swarm of animals in foraging and living. For example, the whale optimization algorithm (WOA) [1], Harris hawks optimization (HHO) [2], artificial neural networks (ANNs) [3,11], deep learning (DL) [4,12], gray wolf optimization (GWO) [5], particle swarm optimization (PSO) [7], differential evolution algorithm (DEA) [9], cuckoo search algorithm (CSA) [10], ant colony optimization (ACO) [13], genetic algorithm (GA) [14], artificial bee colony algorithm (ABC) [15], sparrow search algorithm (SSA) [16], and so on. Recent works verified that swarm intelligence algorithms can obtain good performance and are promising to solve image segmentation problems. However, the heuristic search used by artificial intelligence algorithms also easily to falls into the local optimal solution too early due to the probabilistic [6,9].

Inspired by the intelligent behavior of a natural *Physarum polycephalum*, which can produce a large number of hyphae to search for food [17,18], a novel artificial *Physarum polycephalum* colony algorithm (APPCA) is proposed to help us solve the threshold image segmentation. The main contributions of this article are as follows:

- To overcome the shortcomings of an algorithm search, an APPCA is designed, which can search for the optimal solutions by the expansion and contraction of a lot of artificial hyphae. Different *Physarum polycephalum* hyphae can learn from each other and produce more hyphae in expansion to enhance the global search capability. In contraction, the artificial *Physarum polycephalum* colony (APPC) can select the best hyphae with the highest fitness values through a quick sort algorithm to improve the convergence performance. Different from traditional artificial intelligence algorithms, the population sizes of APPCA in expansion and contraction are different, and the proposed algorithm uses a variable population size to search for optimal segmented images.
- To explore the average information content of different pixels in image segmentation, a fitness function is modeled based on Kapur's entropy for the proposed artificial *Physarum polycephalum* colony algorithm to search for the optimal threshold segmentation solutions. The fitness function can help the APPCA solve the image segmentation problem with high accuracy and short computational time.
- To test the proposed artificial *Physarum polycephalum* colony algorithm, a series of benchmark experiments are implemented, and some state-of-the-art approaches are compared. The experimental results verified the effectiveness of the proposed APPCA. The APPCA can obtain better accuracy and convergence speed, and is not easier to fall into the local optimal solution too early compared with traditional artificial intelligence algorithms.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the threshold image segmentation based on APPCA. Section 4 analyzes the fitness function and the convergence of the APPCA. Section 5 implements benchmark tests for the performance of the APPCA and compares some state-of-the-art algorithms. Finally, this paper is concluded in Section 6.

2. Related Work

Threshold image segmentation is a widely used segmentation technology, where an image is regarded as a combination of two types of areas, i.e., target areas and background areas with different grayscale levels [19]. Taking advantage of the difference between the target areas and the background areas in the grayscale characteristics, threshold image segmentation can generate a corresponding binary image by extracting the target areas [20]. In threshold image segmentation, a reasonable threshold is essential to determine whether each pixel in the image should belong to the target areas or the background areas. Hence, the key in threshold image segmentation is to search for the appropriate threshold accurately and quickly.

To find optimal thresholds in image segmentation, recent researchers selected different methods as follows.

First, the threshold in most cases can be determined by the separation between gray peaks in a histogram of the original image. The gray level of an original image usually presents normal distribution in target and background areas, so the image gray histogram tends to have two single peaks corresponding to the target and background areas, respectively [21]. The adjacent pixels between the target or background areas are correlated in the gray level, but the pixels at the junction of the target and background areas are different in the gray level [22]. In this case, the bottom of the gray level valley between the two peaks can be selected as a threshold.

Second, Ostu proposed the inter-class variance threshold segmentation method in 1979, which divides the image into foreground and background parts based on its grayscale distribution [1,7]. The segmentation values of the foreground and background are calculated using the inter-class variance method to obtain the threshold, so the foreground is the part we segmented. Ref. [23] described an Otsu multi-threshold image segmentation based on an improved particle swarm algorithm.

Third, multi-threshold segmentation is based on multiple valleys appearing in the grayscale histogram as segmentation points, using multiple thresholds to segment multiple peaks [2]. Therefore, there are as many target regions as there are peaks. Ref. [5] illustrated a fault diagnosis algorithm of gearboxes based on GWO-SCE adaptive multi-threshold segmentation and subdomain adaptation. The gray value of pixels may change by noise and produce an imbalance in the histogram of an original image, so the bottom of a histogram may be filled to generate new peaks [7]. Hence, the bottom of the gray level valley between the two peaks can be segmented by multi-threshold [9,10,13].

Fourth, multi-level segmentation can segment an image into multiple sets of segmentation results with different levels of detail at different scales and can determine the relative size between the targets and background [24]. Ref. [25] employed a multi-level image threshold segmentation using an improved Bloch quantum artificial bee colony algorithm. In many original images, there are a lot of small targets or large targets, so it is not easy to determine the valley bottom in the histogram. Traditional single-level image segmentation results in severe loss of final feature hotspots and resolution, leading to a lack of detail characterization ability in segmentation results [26,27]. Multi-level threshold segmentation forms a multi-level network structure representation of the same image at different scales [28,29].

Fifth, dynamic threshold segmentation does not require artificially setting thresholds, but rather segments based on the features present in the image [20]. Dynamic threshold segmentation generally compares the original image with the processed image to generate a difference image and then calculates the bright or dark regions in the difference image. The brightness of the target and background may vary with the illumination, such as maize diseases [7] and cucumber seedlings [14]. Dynamic threshold segmentation has the characteristics of strong anti-interference and stability and is not sensitive to changes in lighting. Ref. [30] explored an unsupervised segmentation method based on dynamic threshold neural P systems for color images.

Sixth, the entropy threshold can describe the degree to which certain material system states may occur, so some researchers employed entropy to depict the texture feature on an original image and select an entropy threshold for image segmentation [31]. Ref. [13] introduced a multi-threshold image segmentation for melanoma based on Kapur's entropy using enhanced ant colony optimization. Kapur's entropy focused on evaluating the compactness of all defined clusters since the intensity follows the cluster's distribution. An entropy threshold with the maximization of Kapur's entropy can search for the clusters around the cluster's centroid and central pixels without outliers.

Therefore, threshold selection and image segmentation involve too many factors, making such heavy tasks and calculations extremely challenging for the human brain [32,33]. Nowadays, with the help of artificial intelligence technology, it is entirely possible to entrust such complex tasks to computers, and they can be performed faster and more accurately than the human brain [6]. Artificial intelligence is the study of the laws of human intelligence activities, the construction of artificial systems with a certain level of intelligence, and the study of how to enable computers to perform tasks that previously required human intelligence, that is, the study of how to apply the basic theories, methods, and techniques of computer software and hardware to simulate certain intelligent behaviors of humans. For example, fuzzy set theory can be applied to a wide range of fields with incomplete or imprecise image segmentation, such as the fuzzy c-means method [34–36]. Wavelet transform is a local transformation of space and frequency, which can effectively extract information from the original images [36,37]. The application of artificial intelligence has made contemporary people no longer consider threshold selection and image segmentation as complex tasks that must be completed by human manual operation.'

Swarm intelligence is an important kind of artificial intelligence, and further, it is a subdomain of combinatorial optimization in intelligent computing. Its algorithm is affected by the natural selection mechanism of "survival of the fittest" in the process of biological evolution and the transmission law of genetic information. Through program iteration simulation of this process, the problem to be solved is regarded as the environment, and the optimal solution is found through natural evolution in the population composed of some possible solutions. In the process of searching for the optimal threshold, the fitness function plays an important role in swarm intelligence. If the fitness is high, the swarm will take the pixel value as a threshold. Hence, the higher the fitness is, the greater the swarm finds the threshold to segment the target from the background. Common swarm intelligence algorithms include the whale optimization algorithm [1,29], Harris hawks optimization [2], artificial neural networks [3,11,30], deep learning [4,12,21,38], gray wolf optimization [5,39], particle swarm optimization [7,23,40], differential evolution algorithm [9], cuckoo search algorithm [10], ant colony optimization [13,33], genetic algorithm [14,40], artificial bee colony algorithm [15,25], sparrow search algorithm [16], moth swarm algorithm (MSA) [24], emperor penguin optimization (EPO) [26], marine predators algorithm (MPA) [27], salp swarm algorithm (SSA) [27], firefly algorithm (FA) [28], Aptenodytes Forsteri optimization algorithm (AFOA) [32], artificial fish swarm algorithm (AFSA) [36], artificial plant community (APC) [41,42], krill swarm (KS) [43], immune system (IS) [44], naked mole-rat algorithm (NMRA) [45], attention mechanism [46], and so on.

Although swarm intelligence algorithms have been proven effective, they have also been complained of being prone to premature local optima. The *Physarum polycephalum* in nature can stretch out tens of thousands of filamentous hyphae to search for surrounding food [17,18,47]. This inspires us as to whether the swarm intelligence of *Physarum polycephalum* can be utilized to provide us with a more efficient algorithm for solving image segmentation problems.

3. Methodology

Here, the mathematical model of threshold image segmentation is built as a multi-objective function with an entropy threshold. The symbol definition is provided in Section 3.1, the artificial *Physarum polycephalum* colony is designed in Section 3.2, the

multi-objective fitness function is built in Section 3.3, the performance measurement is provided in Section 3.4, and the algorithm flow is illustrated in Section 3.5.

3.1. Symbol Definitions

In our proposed model, each *Physarum polycephalum* hyphae is a feasible solution and is evaluated by a fitness function, which determines the searching and learning of hyphae and the optimization process of threshold image segmentation. The proposed model employs Kapur's entropy [13] to calculate the fitness function and variance for *Physarum polycephalum* colony to improve the segmentation performance. The symbol definition is shown in Table 1.

Table 1. Symbol definitions.

Symbol	Definition
O	An original image
A	A segmented image with additive noise
B	A benchmark image
H	The image size in the horizontal direction
V	The image size in the vertical direction
N	The total number of pixels on the original image
L	The grayscale level of an image
a_{ij}	A pixel with a pair of coordinates (i, j) on a gray image A
μ_A	The average gray level of a gray image A
σ_A	The standard deviation of a gray image A
A_{ij}	The intensity value on each pixel with a pair of coordinates (i, j)
K	Kapur's entropy
K_i	Kapur's entropy on a segmentation area
w_i	The possibility for each segmentation cluster
p_i	The possibility of the occurrence of pixels with a gray value i
t	A set of thresholds $t = \{t_1, t_2, t_3, \ldots, t_n\}$
M	The population size of *Physarum polycephalum*
m	The number of *Physarum polycephalum*
S	The population size of hyphae
k	The hyphae number
p_s	The possibility of social learning
p_f	The possibility of free learning
$x^k(Ite)$	A solution of the hyphae k at the iteration of Ite
$x^k_{self}(Ite)$	The self-learning hyphae k at the iteration of Ite
$x^k_{social}(Ite)$	The social-learning hyphae k at the iteration of Ite
$x^m_{free}(Ite)$	The free-learning hyphae k at the iteration of Ite
fit	A fitness function
Ite	An iterative computation counter
Ite_max	The maximization of the iterative computation counter
e_{th}	The error threshold of iterative computation
MSE	The mean squared error
$PSNR$	The peak signal-to-noise ratio
$CorCoe$	The correlation coefficient between two images
$Jaccard$	Jaccard's index, also called Intersection-Over-Union, IoU
$Dice$	The Sørensen Dice coefficient
$SSIM$	The structural similarity index

In threshold image segmentation, it processes an original image O into a segmented image A with a grayscale level L. Both images have the same image size H in the horizontal direction and the same image size V in the vertical direction. Hence, the total number of pixels on the original image is $N = HV$. The threshold segmentation can be implemented by an artificial *Physarum polycephalum* colony with a population size of S, and instructed by a fitness function fit based on Kapur's entropy of K. The segmentation of image A can be

compared with the benchmark image B, where a pixel a_{ij} has a pair of coordinates (i, j) on the gray image A and a pixel b_{ij} has a pair of coordinates (i, j) on the benchmark image B.

3.2. Artificial *Physarum polycephalum* Colony

In this subsection, to overcome the shortcomings of algorithm search, a novel artificial *Physarum polycephalum* colony is designed on the basis of natural Physarum in the literature [17,18,47]. Figure 1 shows an artificial *Physarum polycephalum* colony on a 2-D segmentation image. In our artificial *Physarum polycephalum* colony algorithm, there are three main components, i.e., an original image, an artificial *Physarum polycephalum* colony, and a fitness function.

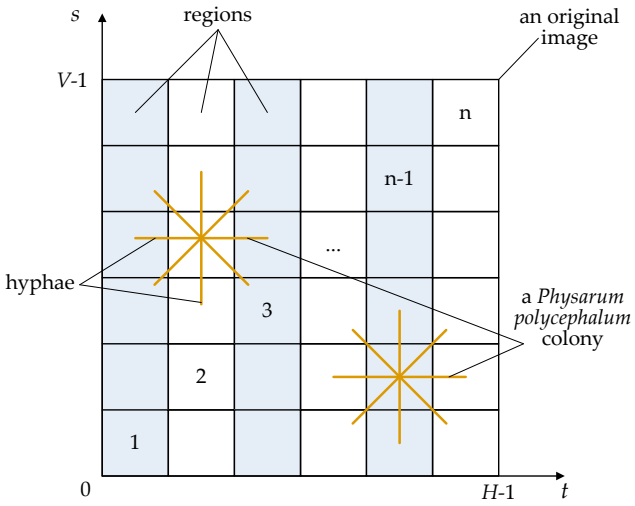


Figure 1. The main architecture of an artificial *Physarum polycephalum* colony.

First, the original image in Figure 1 is not only the living area of an artificial *Physarum polycephalum* colony but also the solution space of the threshold image segmentation or other complex problems. In an original image, there are a lot of pixels with different gray levels. An artificial *Physarum polycephalum* colony can search for feasible solutions on an original image or solution space. Each feasible solution is a segmentation model of the original image. The artificial *Physarum polycephalum* colony should search for feasible solutions as the food sources for its living and the feasible solutions are randomly distributed in the solution space. The food sources can be absorbed by the artificial *Physarum polycephalum* colony and changed into nutrients flowing in its body.

Second, several *Physarum polycephalum* constitute an artificial *Physarum polycephalum* colony in Figure 1, where an artificial *Physarum polycephalum* employs many filiform hyphae to randomly search for feasible solutions. The artificial *Physarum polycephalum* colony algorithm counts the number of hyphae as the population size S. The artificial *Physarum polycephalum* colony employs three learning mechanisms, i.e., self-learning, social learning, and free learning. Hence, different hyphae can learn from each other and many hyphae can cross to share the environmental information. The artificial *Physarum polycephalum* finding the food sources and obtaining more nutrients will survive; otherwise, it will be absorbed and disappear. The leaning mechanism and absorbing behavior can effectively simulate the swarm behavior of a real *Physarum polycephalum* colony.

Third, the artificial *Physarum polycephalum* colony utilizes a fitness function to evaluate its living state, which is not shown in Figure 1. The threshold t and region threshold s segment the original image into several parts, and the fitness depends on the nutrients from the food sources. The artificial *Physarum polycephalum* colony does not know the positions of the feasible solutions from the beginning, but it can judge the feasible solution by a fitness function. If an artificial *Physarum polycephalum* or a hyphae finds a solution, it will obtain a high fitness value. The fitness function can be designed according to the

problems to be solved, such as threshold image segmentation. Different artificial *Physarum polycephalum* can cross and learn the nutrient information to share the fitness information.

Based on the three basic components, the artificial *Physarum polycephalum* colony algorithm uses two main operations to search for the optimal solutions, i.e., expansion and contraction.

In the expansion operation, the artificial *Physarum polycephalum* colony employs three learning mechanisms to produce more hyphae, i.e., self-learning, social-learning, and free-learning. At the initialization, the artificial *Physarum polycephalum* colony produces a population size S of filiform hyphae, where each filiform hyphae is a randomly generated feasible solution. In expansion, the filiform hyphae with a population size of S remain unchanged by self-learning, and all these artificial *Physarum polycephalum* hyphae can learn from each other's experience to produce a new generation of hyphae with a population size of S by social-learning. At the same time, the artificial *Physarum polycephalum* colony will search the whole solution space by free learning to randomly generate several new hyphae completely unrelated to existing hyphae. Now, the population size of hyphae in expansion operation is greater than twice the original state $2S$.

In contraction operation, the artificial *Physarum polycephalum* colony calculates the fitness of all hyphae according to the fitness function and selects the top S best hyphae. Other hyphae with low fitness will be absorbed and disappear. Now, the population size of hyphae in contraction operation restores to the initial value S. The contraction operations can help the artificial *Physarum polycephalum* colony converge to the optimal solution and eliminate hyphae with low fitness. After multiple expansion and contraction iterations and fitness comparisons, the artificial *Physarum polycephalum* colony can find the optimal solution to the threshold image segmentation.

3.3. A Fitness Function for Entropy Threshold

This subsection provides the fitness function for the artificial *Physarum polycephalum* colony to implement threshold image segmentation. To explore the average information content of different pixels in image segmentation, the fitness of the proposed threshold image segmentation is based on the maximum entropy model proposed by Kapur [13,40] and maximizes the total entropy of the segmented target and background. The Kapur's entropy can help us obtain the largest amount of segmentation information on an original image.

In different threshold models, the 1-D segmentation method is the fastest in segmentation speed since it does not consider the gray values of different pixels on an original image. The 3-D segmentation is the lowest in segmentation speed since it considers 3-D gray information. The 2-D segmentation method is between 1-D and 3-D in image segmentation speed since it considers the gray information of each pixel and its neighborhood but not the 3-D gray information. Here, 2-D segmentation is employed since it can obtain balanced performance, as shown in Figure 1. A set of thresholds $t = \{t_1, t_2, t_3, \ldots, t_n\}$ segments the target and background into several regions, and the other regions are mainly image noise.

The fitness function should instruct the artificial *Physarum polycephalum* colony to be efficient and accurate in searching for the best threshold $(t_1, t_2, t_3, \ldots, t_n)$ to obtain the best segmentation quality. A good fitness function should make full use of the gray information of the original image and suppress the noise from target and background areas.

Here, the Kapur's entropy $K(t_1, t_2, t_3, \ldots, t_n)$ can be calculated for the set of thresholds $(t_1, t_2, t_3, \ldots, t_n)$ as follows [13,40]:

$$K(t_1, t_2, t_3, \ldots, t_n) = \sum_{i=0}^{n} K_i, \tag{1}$$

where $K(t_1, t_2, t_3, \ldots, t_n)$ indicates Kapur's entropy in all areas. Hence, Kapur's entropy in each area can be computed as

$$K_0 = -\sum_{i=0}^{t_1-1} \frac{p_i}{w_0} \ln\left(\frac{p_i}{w_0}\right); w_0 = \sum_{i=0}^{t_1-1} p_i \tag{2}$$

$$K_1 = -\sum_{i=t_1}^{t_2-1} \frac{p_i}{w_1} \ln\left(\frac{p_i}{w_1}\right); w_1 = \sum_{i=t_1}^{t_2-1} p_i \tag{3}$$

$$K_n = -\sum_{i=t_n}^{L-1} \frac{p_i}{w_n} \ln\left(\frac{p_i}{w_n}\right); w_n = \sum_{i=t_n}^{L-1} p_i, \tag{4}$$

where the parameters $w_0, w_1, \ldots, w_n$ indicate the possibility for each segmentation cluster, and p_i indicates the possibility of the occurrence of pixels with a gray value i. Now, Kapur's entropy can be obtained for each cluster and instruct the artificial *Physarum polycephalum* colony to evaluate the characteristics of clusters. There is

$$fit = argmax\{K(t_1, t_2, t_3, \ldots, t_n)\}. \tag{5}$$

The combination maximization function *fit* is the optimal threshold group. The artificial *Physarum polycephalum* colony uses the fitness function to select the optimal solution. The higher the fitness function, the better the quality is achieved.

3.4. Performance Measurement

This subsection describes the evaluation indicators for the performance measurement of threshold image segmentation. These indicators can measure the segmentation performance from different aspects based on the original image, where the original image is a reference and the segmentation models from the same original image are subjected to the image noise. In the benchmark tests of Section 5, the indicators can help us compare the performance difference between the proposed APPC algorithm and the traditional artificial intelligence ones.

First, the mean squared error (*MSE*) is a popular index to measure the difference between two images by computing the average sum of the squares between two images. The *MSE* can evaluate the total difference of all pixels on two whole images, where a lower *MSE* value means less difference between two images. The *MSE* can be written as follows:

$$MSE = \frac{1}{HV} \sum_{i=1}^{H} \sum_{j=1}^{V} (a_{ij} - b_{ij})^2, \tag{6}$$

where H represents the image size in the horizontal direction, V represents the image size in the vertical direction, a_{ij} is a pixel on the segmented image A with a pair of coordinates (i, j), and b_{ij} is a pixel on the benchmark image B with a pair of coordinates (i, j). The drawback of *MSE* is that the value often has the same weight in different grayscale. For example, a high *MSE* can be obtained from a noisy background on a well-recognizable image.

Second, the peak signal-to-noise ratio (*PSNR*) is an important indicator of the quality of image segmentation. It calculates the ratio of the effective signal to the noise signal on an image, that is, the ratio of the segmented image to the benchmark image. According to the mean square deviation in Equation (6), the *PSNR* can be defined for two images with the same size:

$$PSNR = 10 \times \log\left(\frac{L^2}{MSE}\right) \tag{7}$$

where *MSE* is the mean squared error in Equation (6), and L is the grayscale level of the image. A high *PSNR* often means high quality, and a low *PSNR* often means low quality. A high *PSNR* also indicates good image recognition.

Third, the correlation coefficient (*CorCoe*) is also a popular indicator to measure the linear correlation between two images. The *CorCoe* can compare the intensity difference of each pixel between two images and the average intensity difference between two whole images. The *CorCoe* can be calculated as follows:

$$CorCoe = \frac{\sum_i^H \sum_j^V (A_{ij} - \overline{A})(B_{ij} - \overline{B})}{\sqrt{\sum_i^H \sum_j^V (A_{ij} - \overline{A})^2} \sqrt{\sum_i^H \sum_j^V (B_{ij} - \overline{B})^2}} \tag{8}$$

where A_{ij} and B_{ij} indicate the intensity value on each pixel with a pair of coordinates (i, j) for two images A and B, respectively. $\overline{A}$ and $\overline{B}$ indicate the arithmetic mean of intensity value for two images A and B, respectively. H represents the image size in the horizontal direction, and V represents the image size in the vertical direction. The *CorCoe* metric often takes values in $[-1, 1]$, where 1 represents the highest match between two images, but -1 represents the lowest match between two images.

Fourth, Jaccard's index (*Jaccard*), also called Intersection-Over-Union, IoU, is another popular indicator to measure the segmentation quality. The *Jaccard* metric calculates the size of the intersection divided by the union size of each set of pixels to measure the similarity between finite sets of pixels. The calculation equation can be obtained as follows:

$$Jaccard = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{9}$$

where A is the segmented image with additive noise and B is the benchmark image. The numerator $|A \cap B|$ calculates the overlap area of images A and B, and the denominator $|A \cup B|$ describes the fusion area of images A and B. *Jaccard* index ranges from 0 to 1, where a value of 1 indicates a perfect overlap or the greatest similarity, but a value of 0 indicates no spatial overlap or an absolute mismatch.

Fifth, the Sørensen Dice coefficient (*Dice*) is similar to Jaccard's coefficient but is the double overlap region divided by the total number of pixels in two images. The *Dice* and *Jaccard* are positively correlated and are easy to reach a consensus assessment. For example, they both may determine model A is better than model B in segmentation performance, or vice versa. The calculation equation is as follows:

$$Dice = \frac{2|A \cap B|}{|A| + |B|} = \frac{2|A \cap B|}{|A \cup B| + |A \cap B|} \tag{10}$$

where A is the segmented image with additive noise and B is the benchmark image. The numerator $|A \cap B|$ calculates the overlap area of images A and B, and the denominator $|A \cup B|$ describes the fusion area of images A and B. *Dice* coefficient also ranges from 0 to 1, where a value of 1 indicates a perfect overlap or the greatest similarity, but a value of 0 indicates no spatial overlap or an absolute mismatch.

Sixth, the Structural Similarity Index (*SSIM*) calculates a multiplicative combination of three image components, i.e., brightness, contrast, and texture. The *SSIM* can describe the scene structure information on an image.

For a gray image A, the average gray level μ_A can be calculated to estimate the brightness, so the brightness can be written as follows:

$$\mu_A = \frac{1}{HV} \sum_{i=1}^{H} \sum_{j=1}^{V} a_{ij}, \tag{11}$$

where H represents the image size in the horizontal direction, V represents the image size in the vertical direction, and a_{ij} is a pixel on the segmented image A with a pair of coordinates (i, j).

The measurement system removes the average grayscale value μ_A from the image. For a digital image A, standard deviation σ_A can be used as the contrast estimation value.

$$\sigma_A = \sqrt{\frac{1}{HV - 1} \sum_{i=1}^{H} \sum_{j=1}^{V} (x_{ij} - \mu_A)^2} \tag{12}$$

According to the brightness μ_A in Equation (11) and the contrast σ_A in Equation (12), the texture can be expressed as their function. Therefore, the *SSIM* can be calculated as follows:

$$SSIM = \frac{(2\mu_A\mu_B + C_1)(2\sigma_A\sigma_B + C_2)}{(\mu_A^2 + \mu_B^2 + C_1)(\sigma_A^2 + \sigma_B^2 + C_2)} \tag{13}$$

where the average gray level μ_A and μ_B estimates the brightness, and the standard deviation σ_A and σ_B evaluates the contrast. The constant C_1 is used to avoid instability of the evaluation system when the brightness approaches 0, often $C_1 = (K_1 L)^2$. L is the grayscale level of the image, i.e., L = 255 for 8-bit grayscale images. K_1 is a positive constant far less than 1. The constant C_2 is used to avoid instability of the evaluation system when the contrast approaches 0, often $C_2 = (K_2 L)^2$. L is the grayscale level of the image, and K_2 is a positive constant far less than 1.

In the experimental results of Section 5, the performance measurement indicators above are employed for objective evaluation and algorithm comparison, which can objectively verify the proposed APPC algorithm and compare the segmentation performance.

3.5. Algorithm Flow of APPCA

This subsection provides the algorithm flow of APPCA. To solve the threshold image segmentation problem, all *Physarum polycephalum* will continuously search for the optimal solutions by iterative computation according to the algorithm flow, as shown in Figure 2. The whole algorithm flow includes four main stages, namely initialization, expansion, contraction, and end judgment.

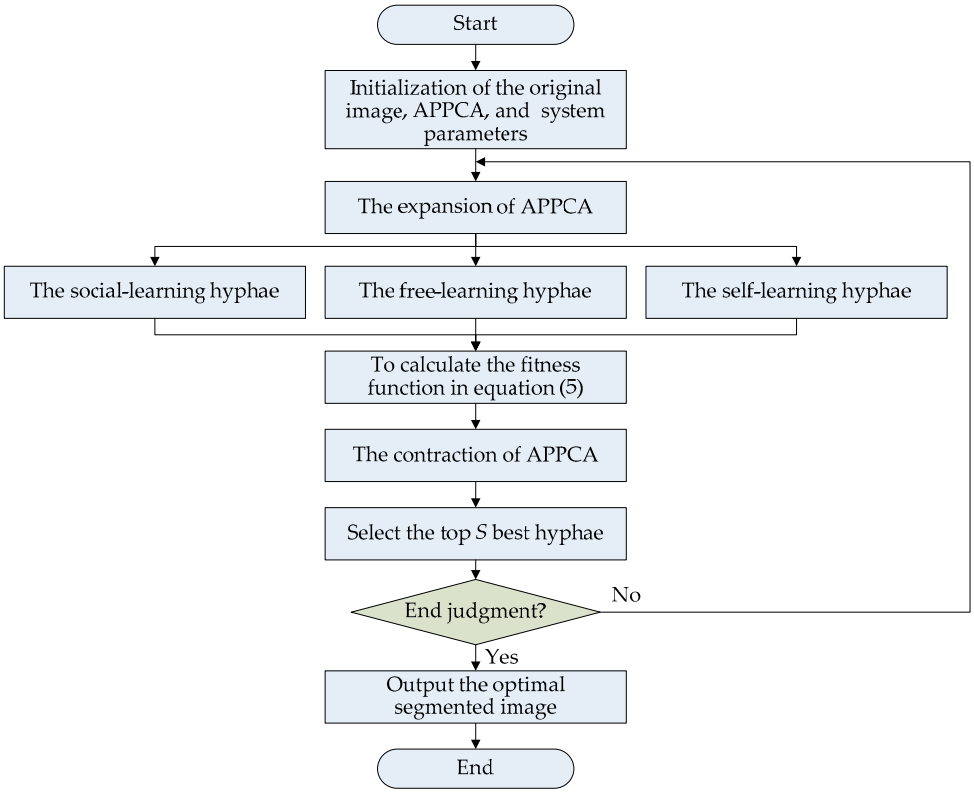


Figure 2. The algorithm flow of APPC for threshold image segmentation.

In the initialization stage, there are three main parts, i.e., the original image, the artificial *Physarum polycephalum* colony, and the system simulation parameters.

In the expansion stage, multiple *Physarum polycephalum* will search for the optimal solutions according to the fitness in Equation (5). The *Physarum polycephalum* colony selects the best solutions to the fitness function of Kapur's entropy. The *Physarum polycephalum* colony will learn the nutrient information to expand and contract with self-learning, social learning, and free learning to implement efficient image segmentation. Therefore, the population size increases.

In the contraction stage, the *Physarum polycephalum* colony will evaluate the fitness of hyphae and select the best ones for the next iterations. The optimal hyphae with high fitness will be left and the population size will decrease. After many iterations of expanding

and contracting, it is possible to obtain an optimal segmented image. For all *Physarum polycephalum*, it is not compulsory to search for all solutions. It is sufficient to select the optimal hyphae through a sufficient number of iterative calculations.

At the end of iteration learning, the error varying becomes less and less, and then the solution can be output. A predefined error threshold e_{th}, the number of iterations *Ite_max*, or a maximum calculation time limit can be used as an end judgment before the optimal solutions are generated.

4. Algorithm Steps of APPCA

This section illustrates the algorithm of the proposed APPCA. The threshold image segmentation is NP-hard and cannot be easily solved by traditional algorithms. Here, an artificial *Physarum polycephalum* colony algorithm uses a kind of novel swarm learning mechanism and variable population size to search for the optimal solution to threshold image segmentation. The proposed artificial *Physarum polycephalum* colony can collaboratively deform according to the fitness in Equation (5), that is, the artificial hyphae with the highest fitness are where the optimal solution is.

In our artificial *Physarum polycephalum* colony algorithm, the whole solving process consists of four main stages, namely initialization, expansion, contraction, and end judgment. The first initialization stage will set all important parameters and define the objective function and constraints. In the second expansion stage after the initialization, multiple *Physarum polycephalum* will continuously use hyphae to expand and search for optimal solutions. In the third contraction stage, multiple *Physarum polycephalum* will evaluate the fitness of hyphae to contract to form an optimal solution to threshold image segmentation. Similar to a real *Physarum polycephalum* colony, the artificial *Physarum polycephalum* colony can also collaboratively search for the optimal solutions in parallel. At the end judgment stage, the artificial *Physarum polycephalum* colony will judge whether the iterative computation is completed, and the optimal segmented solution will be output in the end.

4.1. Step 1: Initialization

The threshold image segmentation at the start of the solving process is initialized from three main aspects, i.e., the original image, the artificial *Physarum polycephalum* colony algorithm, and the system simulation parameters.

First, the original image needs to be formatted before segmentation, where the original color image should be transformed into a grayscale image with a gray level of L.

Second, the initialization of the artificial *Physarum polycephalum* colony algorithm sets the parameters of the APPC. The maximum number of hyphae S, the possibility p_s of social learning, and the possibility p_r of free learning should be set first. The artificial *Physarum polycephalum* colony algorithm can search for the optimal solutions in the whole solution space and the artificial hyphae can move from one pixel to another to connect them into a segmented image. The segmented image for hyphae k can be seen as a feasible solution. The fitness in Equation (5) can be set to zero at the start.

In the process of initialization, *Ite* = 1, the artificial *Physarum polycephalum* colony should be randomly encoded as the feasible solutions to threshold image segmentation and the algorithm can easily adjust the initialization parameters on a personal computer to fit the requirements of threshold image segmentation, and obtain good performance. Assuming there are S hyphae in the APPC, each hyphae indicates a feasible solution. For a hyphae $k(k = 1, 2, 3, \ldots, S)$, it can produce a random segmented vector x^k of the original image:

$$x^k(1) = \left\{ x_i^k(1) \middle| i = 1, 2, 3, \ldots, HV \right\}, \tag{14}$$

where $x_i^k = \{0, 1\}$ is the selecting bit of pixel i calculated by hyphae k. For a hyphae k, if pixel i is selected as a segmented region, then $x_i^k = 1$; otherwise, $x_i^k = 0$. Therefore, in subsequent iterative calculations, the optimal solution x^* can be calculated in Equation (14) and compared by S hyphae according to the objective function in Equation (5).

Third, the system simulation parameters should be set during initialization. The simulation time is set as $t = 0$. The simulation end conditions should be predefined here, including the maximum simulation time T_{max}, maximum iteration steps *Ite_max*, and minimum iteration error threshold e_{th}. If any of these conditions are met, the iterative calculation can be ended.

4.2. Step 2: Expansion

The expansion is an important step of an artificial *Physarum polycephalum* colony algorithm to search for feasible solutions before an optimal segmented image is output. In expansion, the most important thing is that the population size increases. The expanded population size includes three parts, i.e., the self-learning hyphae, the social-learning hyphae, and the free-learning hyphae.

First, the original hyphae vary between the first iteration ($Ite = 1$) and subsequent iterations ($Ite > 1$). In the first iteration ($Ite = 1$), the original population is the randomly generated hyphae with a population size S in the initialization stage, that is $Ite = 1$, a random segmented image $[x^k]_S$, where $x^k = \left\{ x_i^k \middle| i = 1, 2, 3, \ldots, HV \right\}$. Since they are randomly distributed in the whole solution space, their fitness may vary greatly and the optimal solution cannot be determined. However, in the subsequent iterations ($Ite > 1$), the original population is the hyphae with a population size S generated through contraction operations in the previous iteration calculation $(Ite - 1)$. Different from the random hyphae population generated in the first iteration, the hyphae in the subsequent iterations are the optimal solution with the highest fitness in the previous iteration, that is, the optimal segmented images $x^*(Ite - 1)$ of APPC.

$$\begin{cases} x_{self}^k(Ite) = x^k(1) = random(k), if\ Ite = 1 \\ x_{self}^k(Ite) = x^*(Ite - 1), if\ Ite > 1 \end{cases} \tag{15}$$

In expansion, multiple *Physarum polycephalum* will collaboratively search for the optimal solutions to identify a set of neighbor pixels. When multiple *Physarum polycephalum* hyphae move through several pixels, each *Physarum polycephalum* hyphae will first identify the rest pixels, which are closer to it. In each iterative computing, the segmented image will change with time-varying. There are two expansions in this stage: the one with those solutions with high fitness in Equation (5) will cross for social learning and produce a new generation of solutions; the other is that the *Physarum polycephalum* will randomly search for the feasible solutions in the whole solution space.

Second, the social-learning hyphae allow different hyphae to learn environmental knowledge from each other and generate a new generation of hyphae with a population size S. This simulates the social-learning behavior of natural *Physarum polycephalum* colony and helps to search in the solution space near the optimal solution in each iteration, improving search efficiency and convergence speed. The social learning of an artificial *Physarum polycephalum* colony employs the strategy of division and cooperation and provides the algorithm with good swarm learning and searching capabilities. For a pair of parent-segmented images $x^k(Ite - 1)$ and $x^l(Ite - 1)$, a pair of children-segmented images $x_{social}^k(Ite)$ and $x_{social}^l(Ite)$ can be obtained.

$$\begin{cases} x_{social}^k(Ite) = x^k(Ite - 1) \times p_s + x^l(Ite - 1) \times \left(1 - p_s\right) \\ x_{social}^l(Ite) = x^k(Ite - 1) \times \left(1 - p_s\right) + x^l(Ite - 1) \times p_s \end{cases} \tag{16}$$

Due to the completely random nature of social learning, each hyphae has at least one opportunity to learn from any other hyphae, resulting in a new hyphae that is different from themselves. A new generation of hyphae includes two parts, where the one is the same part of the parent by a social-learning probability p_s, while the other part different from the parent comes from other hyphae with a probability of $1 - p_s$. The larger the p_s, the

more experience is retained for self-learning, the stronger the local search ability, and the faster the convergence speed; on the contrary, the smaller the p_s, the more social-learning experience it has, and the stronger its global search ability, but the slower its convergence speed. In Equation (16), only one child is randomly selected for the next iteration.

Third, the free-learning hyphae allow the artificial *Physarum polycephalum* colony to randomly produce a new generation of hyphae completely unrelated to the existing hyphae in each iteration. Because the search is completely random, free learning helps to maintain a certain level of global search ability in each iteration, avoiding premature convergence to the local optimal solution. The population size is decided by the possibility p_f of free learning, that is, a rand-segmented image of the free-learning population $x^m_{free}(Ite)$.

$$x^m_{free}(Ite) = Random(Ite)(m = 1, 2, 3, \ldots, S \times p_f) \tag{17}$$

Considering the original population size S, the social-learning population size S, and the free-learning population size Sp_f, the total population size is $(2 + p_f)S$ after the expansion operation. There is

$$x^k(Ite) = \left\{ x^k_{self}(Ite), x^k_{social}(Ite) \text{ or } x^l_{social}(Ite), x^m_{free}(Ite) \right\} \tag{18}$$

$$x(Ite) = \left\{ x^k(Ite) \middle| k = 1, 2, 3, \ldots, S \right\}. \tag{19}$$

By the aiding of more hyphae population, all pixels will be arranged into the feasible solutions by the artificial *Physarum polycephalum* colony algorithm, where the optimal scheduling solutions may be generated by the fitness in the contraction stage and different hyphae can obtain different fitness.

Now, the pseudo-code of expansion is provided and the algorithm performance is analyzed. Assuming the scale of the threshold image segmentation includes a total of HV pixels, and the population size of the *Physarum polycephalum* colony algorithm includes the maximum number M of *Physarum polycephalum* and the maximum number S of hyphae in each *Physarum polycephalum*, there is the expansion algorithm in Algorithm 1.

Algorithm 1 Expansion with a triple-loop

```
Input: the original image, H, V, grayscale L
Define: the fitness function in Equation (5)
Initialization: the APPC, M, S, p_s, p_f
Calculate: randomly generate hyphae {x^k(1)}
Initialization: T_max, Ite_max, e_th
For Ite = 1:Ite_max
    For Physarum polycephalum m = 1 : M
        For hyphae k = 1 : S
            Random select a pixel
            Self-learning
            Social learning by p_s
            Free learning by p_f
            To integrate the three parts of the expansion population
        End for
        To update the Physarum polycephalum colony by Equation (18)
    End for
    To update the Physarum polycephalum colony by Equation (19)
    Return the solutions to threshold image segmentation
//contraction algorithm
End for
//Output the optimal segmented image
```

As we can see from Algorithm 1, there is a triple-loop, so the time complexity of the expansion algorithm is $O(Ite_\max MSHV)$, and the space complexity is $O(MSHV)$. In the triple-loop of expansion, the APPC has M of *Physarum polycephalum* and each has S hyphae. The APPC will search for a total of HV pixels and select the best segmented image with the highest fitness. The APPC expansion algorithm seems to have good search capabilities, but the computation seems too time-consuming.

Now, it is necessary to improve the expansion algorithm with a triple-loop in Algorithm 1. Since the *Physarum polycephalum* colony has a strong ability to learn from each other, it is unnecessary to investigate which each *Physarum polycephalum* hypha comes from. In this way, one loop can be directly reduced, greatly improving the algorithm's computational performance and search efficiency. The improved expansion of APPC is shown in Algorithm 2. The problem scale and the *Physarum polycephalum* colony are the same as in Algorithm 1, where there are a total of HV pixels and the maximum number S of hyphae. The maximum number M of *Physarum polycephalum* is neglected.

Algorithm 2 Expansion with a dual-loop

```
Input: the original image, H, V, grayscale L
Define: the fitness function in Equation (5)
Initialization: the APPC, M, S, p_s, p_f
Calculate: randomly generate hyphae {x^k(1)}
Initialization: T_max, Ite_max, e_th
For Ite = 1:Ite_max
    For hyphae k = 1 : S
        Random select a pixel
        Self-learning
        Social learning by p_s
        Free learning by p_f
        To integrate the three parts of the expansion population
        To update the Physarum polycephalum colony by Equation (18)
    End for
    To update the Physarum polycephalum colony by Equation (19)
    Return the solutions to threshold image segmentation
//contraction algorithm
End for
//Output the optimal segmented image
```

Compared with the expansion algorithm with a triple-loop in Algorithm 1, the improved expansion algorithm in Algorithm 2 has a dual-loop, so the time complexity of the expansion algorithm is improved as $O(Ite_\max SHV)$, and the space complexity is enhanced to $O(SHV)$. In the dual-loop of expansion, the APPC uses a total of S hyphae to search for a total of available HV pixels and selects the optimal solutions to the same threshold image segmentation. After improvement, the new algorithm no longer distinguishes the source of hyphae, and the time and spatial performance have been greatly improved. The improved APPC expansion algorithm both has good search capabilities and higher computational efficiency.

4.3. Step 3: Contraction

In this step, the artificial *Physarum polycephalum* colony will contract and the optimal solutions to the threshold image segmentation will be selected. The population size of the Physarum hyphae will decrease after comparing the fitness of the feasible solutions. As we all know, the population size increases from S during initialization to $(2 + p_f)S$ after expansion, but not all hyphae can survive. Only the hyphae with the highest fitness of

population size S can survive after contraction, the other population $(1 + p_f)S$ with low fitness will disappear. There is

$$x^*(Ite) = \left\{ x^{k*}(Ite) \middle| fitness\left(x^{k*}(Ite)\right) \geq fitness\left(x^{l}(Ite)\right), k < S, l > S \right\}. \quad (20)$$

The *Physarum polycephalum* colony will compare the fitness obtained by Equation (5), and the optimal hyphae $x^{1*}(Ite)$ with the highest fitness will be selected as the optimal scheduling solution to be output.

$$fitness\left(x^{1*}(Ite)\right) \geq fitness\left(x^{k*}(Ite) \middle| k = 1, 2, 3, \ldots, S\right) \quad (21)$$

Similarly, here also consider two cases, i.e., contraction with a triple-loop and contraction with a dual-loop.

Assuming the scale of the threshold image segmentation includes a total of available HV pixels, and the population size of the *Physarum polycephalum* colony algorithm includes the maximum number M of *Physarum polycephalum* and the maximum number S of hyphae. Bubble sort is employed to rank the fitness of hyphae with a population size $(2+p_f)MS$ in total. The pseudo-code of contraction is shown in Algorithm 3.

Algorithm 3 Contraction algorithm with a triple-loop

```
For Ite = 1:Ite_max
//expansion algorithm
    For hyphae k = 1:(2 + p_f)MS
        For hyphae s = 1:(2 + p_f)MS-1
            If fitness (s) > fitness(s + 1)
                temp = fitness (s)
                fitness (s) = fitness(s + 1)
                fitness(s + 1) = temp
            End if
        End for
    End for
    To store the optimal solution Temp_s = 1
    To store the optimal fitness Temp_fitness = fitness(1)
    To calculate the iterative error e= | fitness(1)(Ite)- fitness(1)(Ite-1) |
    To judge the error with end condition: T_max, Ite_max, e_th
    If the end condition does not fit
        Select the top S best hyphae
        Return to the expansion algorithm
    Else
        Exit
End for
Exit: output the optimal segmented image x^1*(Ite)
Exit: output the optimal fitness fitness(x^1*(Ite))
```

As we can see from Algorithm 3, there is a triple-loop and a bubble sort algorithm, so the time complexity of the contraction algorithm is $O((2 + p_f)^2 M^2 S^2)$, the best time complexity is $O((2 + p_f)MS)$, and the space complexity is $O(MS)$. In the triple-loop of contraction, the APPC has M of *Physarum polycephalum* and each has S hyphae. The APPC will rank the fitness of all hyphae using the bubble sort algorithm and select the top S best hyphae. If the end condition is fit, such as the error threshold e_{th}, the maximum iterations Ite_max, and the maximum simulation time T_{max}, the optimal solution(rank No.1) and the corresponding fitness (fitness of No.1) will be output. Otherwise, the top S best solutions will be selected from all solutions and returned to the expansion algorithm. The APPC contraction algorithm seems to have good sorting capabilities, but the computation is still too time-consuming.

Similarly, the contraction algorithm with a triple-loop in Algorithm 3 can also be improved. Then, the quick sort is introduced here to replace the bubble sort in Algorithm 3, improving the algorithm's time performance and search efficiency. The improved contraction of APPC is shown in Algorithm 4. The problem scale and the *Physarum polycephalum* colony are the same as the Algorithm 3, where there are a total of HV pixels and the maximum number S of hyphae. The maximum number M of *Physarum polycephalum* is ignored.

Algorithm 4 Contraction algorithm with a dual-loop

```
For Ite = 1:Ite_max
//expansion algorithm
    For hyphae k = 1:(2 + p_f) S
        QuickSort(s, 1, (2 + p_f) S)
        To store the optimal solution Temp_s = 1
        To store the optimal fitness Temp_fitness = fitness(1)
    End for
    To calculate the iterative error e= | fitness(1)(Ite)- fitness(1)(Ite-1) |
    To judge the error with end condition: T_max, Ite_max, e_th
    If the end condition is not fit
        Select the top S best hyphae
        Return to the expansion algorithm
    Else
        Exit
End for
Exit: output the optimal segmented image x^{1*}(Ite)
Exit: output the optimal fitness fitness(x^{1*}(Ite))
```

Compared with the contraction algorithm with a triple-loop in Algorithm 3, the improved contraction algorithm in Algorithm 4 employs the quick sort with better performance than the bubble sort, so the time complexity of the contraction algorithm is improved as $O((2+p_f)S\log_2[(2+p_f)S])$, the best time performance is also $O(\log_2[(2+p_f)S)]$, and the space complexity is $O((2+p_f)S)$. In the dual-loop of contraction, the improved algorithm no longer distinguishes the source of hyphae, the time performance has been greatly improved, and the spatial performance is stable. The improved APPC contraction algorithm has higher computational efficiency.

4.4. Step 4: End Judgment

The end conditions of the proposed APPC approach can be judged by a predefined error threshold e_{th} or the number of iterations *Ite_max*. If the end conditions are not satisfied, the top S best solutions in the contraction algorithm will be returned to the expansion algorithm, and the process of expansion and contraction will be repeated until the end conditions are fit. Finally, an optimal segmented image $x^{1*}(Ite)$ will be output as a solution to the threshold image segmentation.

5. Benchmark Tests and Comparison

In this section, to test the proposed artificial *Physarum polycephalum* colony algorithm, we design a series of benchmark experiment tests on threshold image segmentation. Some mainstream artificial intelligence algorithms are also employed for comparison to verify the proposed APPC algorithm.

5.1. Experiment Design

For a fair comparison, some assumptions are made as follows:

- It is assumed that the ethics, privacy, politics, race, topic, history, ecology, and complexity factors of the original images are not taken into account.

- It is assumed that the differences between the binarization methods, the filters, and the color differences of the pixels are not considered.
- It is assumed that the performance difference of different algorithms running on different software and hardware of personal computers is ignored.
- It is assumed that the optimal parameter adjustment on determined datasets is not discussed. Although each algorithm has a lot of improved editions to obtain better performance, this is not considered here.

There are many datasets for benchmark tests of image segmentation, but they cannot all be selected for our experiments. Here, the famous Pascal VOC dataset and Stanford background dataset were selected for our benchmark test, and they have always been the benchmark test dataset in the field of image segmentation. The links to the two supporting datasets are provided in the Data Availability Statement at the end of this article. The experimental platform includes an AMD Ryzen 3 4300U with Radeon Graphics 2.70 GHz CPU, 8.00 GB RAM, a 64-bit Windows 10 operating system, and Matlab R2018a simulation software.

Alongside our APPCA, some mainstream swarm intelligence algorithms are employed for performance comparison, including deep learning (DL) [4,12,21,38], gray wolf optimization (GWO) [5,39], particle swarm optimization (PSO) [7,23,40], differential evolution algorithm (DEA) [9], ant colony optimization (ACO) [13,33], genetic algorithm (GA) [14,40], artificial bee colony algorithm (ABC) [15,25], artificial fish swarm algorithm (AFSA) [36], and immune system (IS) [44]. For a fair comparison, only twenty agents (S = 20) were used in the benchmark tests, the fitness function selected Equation (5), and the simulation parameters of different algorithms were kept unchanged in the whole tests. For the simulation system, the maximum number of iterations is preset to 200, and the iterative error threshold is less than 0.1%.

The main simulation parameters used in our proposed algorithm are as follows.

The population size of the artificial *Physarum polycephalum* colony is S = 20. For the two learning factors, the social-learning probability is p_s = 0.9, and the free-learning probability is set as p_s= 0.2.

For DL [4,12,21,38], the parameters include a convolutional neural network with eight convolution cores, eight input channels (c_{in} = 8), and eight output channels (c_{out} = 8). The learning rate of the offset item was twice that of the weight. The extension edge was set to 0, the weight was initialized to Gaussian, and the value of the constant offset item was 0.

For GWO [5,39], the population size of the gray wolves was set to S = 20, the upper bound of variables was set as ub = [10,10], the lower bound of variables was set as lb = [−10, −10], and the dimension of the problem was set as dim = 2.

For PSO [7,23,40], the population size of the particles was set to S = 20, the location limitation was 0.5, the speed limitation was [−0.5, 0.5], the self-learning factor was $c1$ = 1.5, and the social-learning factor was $c2$ = 1.5.

For DEA [9], the population size was set to S = 20, the dimension of the variable to be solved was 20, the crossover factor was cr = 0.1, and the initial mutation factor was $f0$ = 0.4.

For ACO [13,33], the population size of ants was set to S = 20, the parameters contain a pheromone importance of 1.0, a heuristic factor importance of 5.0, and a pheromone volatilization factor of 0.1.

For GA [14,40], the population size of the chromosome was set to S = 20, the chromosome length was $Lind$ = 20, the crossover probability was px = 0.7, and the mutation probability was pm = 0.01.

For ABC [15,25], the parameters include the population size of bees S = 20, the searching capability limit = 100, and the neighborhood size NI = 10.

For AFSA [36], the population size of fish was S = 20, the maximum number of attempts was 100, the perceptual distance was 1.5, the congestion factor was 0.5, and the fish step was 0.2.

For IS [44], the population size of the chromosome was set to $S = 20$, the incentive coefficient included *alpha* = 3 and *beta* = 1, the dimension of the variable to be solved was 10, the concentration threshold *delta* = 0.3, and the mutation probability was *pm* = 0.6.

5.2. Benchmark Test on Pascal VOC 2012 Dataset

The Pascal VOC 2012 segmentation dataset is employed here for the benchmark tests, which is the ancestor of visual recognition competitions. It provides benchmark tests for image segmentation, target detection, object classification, and other image-processing tasks. It contains 20 categories of objects for image segmentation, including people, animals, ships, vehicles, planes, and indoor scenes. Figure 3 shows the experimental results on six original images from the Pascal VOC 2012 dataset. The image IDs of the six images are 2012_001155, 2012_003403, 2012_003438, 2012_004019, 2012_004134, and 2012_004248, and they have 500 × 500 pixels, 314 × 186 pixels, 500 × 332 pixels, 500 × 375 pixels, 500 × 493 pixels, and 500 × 333 pixels, respectively.

Figure 3 presents benchmark test results on the Pascal VOC 2012 dataset, including the image ID on the dataset in the first line, the original images in the second line, the 2-D histogram in the third line, and the segmented images of different algorithms in the following lines, respectively. For the fairness of the benchmark test, all the algorithms were tested under the same conditions for comparison. Furthermore, to accurately calculate each algorithm's performance on different benchmark images, all algorithms ran at the same time. At the end of the test, all algorithms can obtain similar experimental results in Figure 3.



Figure 3. *Cont.*

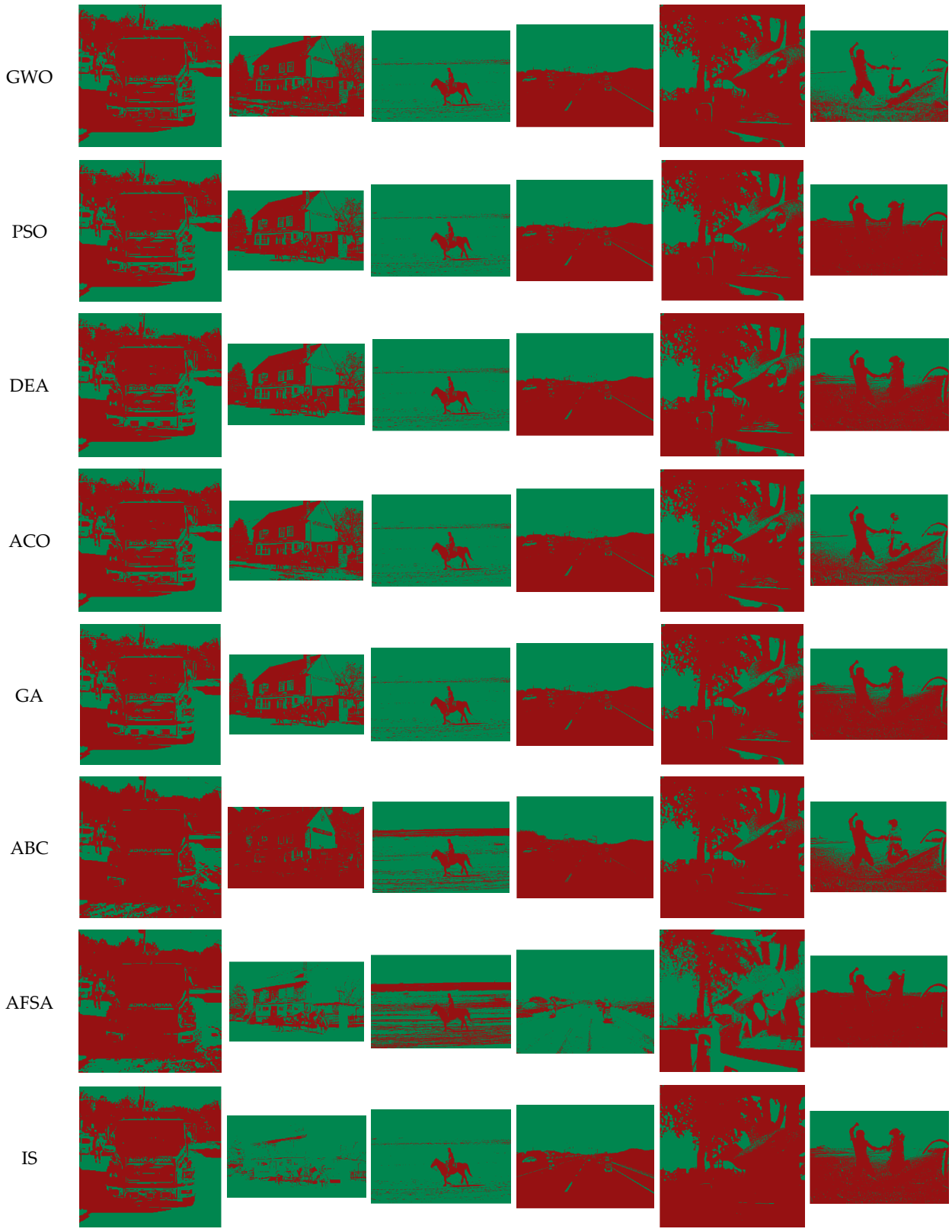


Figure 3. Benchmark test on Pascal VOC 2012 dataset.

After that, Figure 4 showed the fitness convergence curves of APPCA and other artificial intelligence algorithms on the same test images from the Pascal VOC 2012 dataset, where the maximum number of iterations was preset to 200. The convergence curves

further verified that APPCA obtains lower fitness errors than the other artificial intelligence algorithms in most cases. As we can see, APPCA is less likely to fall into local optima than other algorithms. Other artificial intelligence algorithms can also search for the optimal solution with a certain probability but, usually, they are more likely to fall into local optima too early.

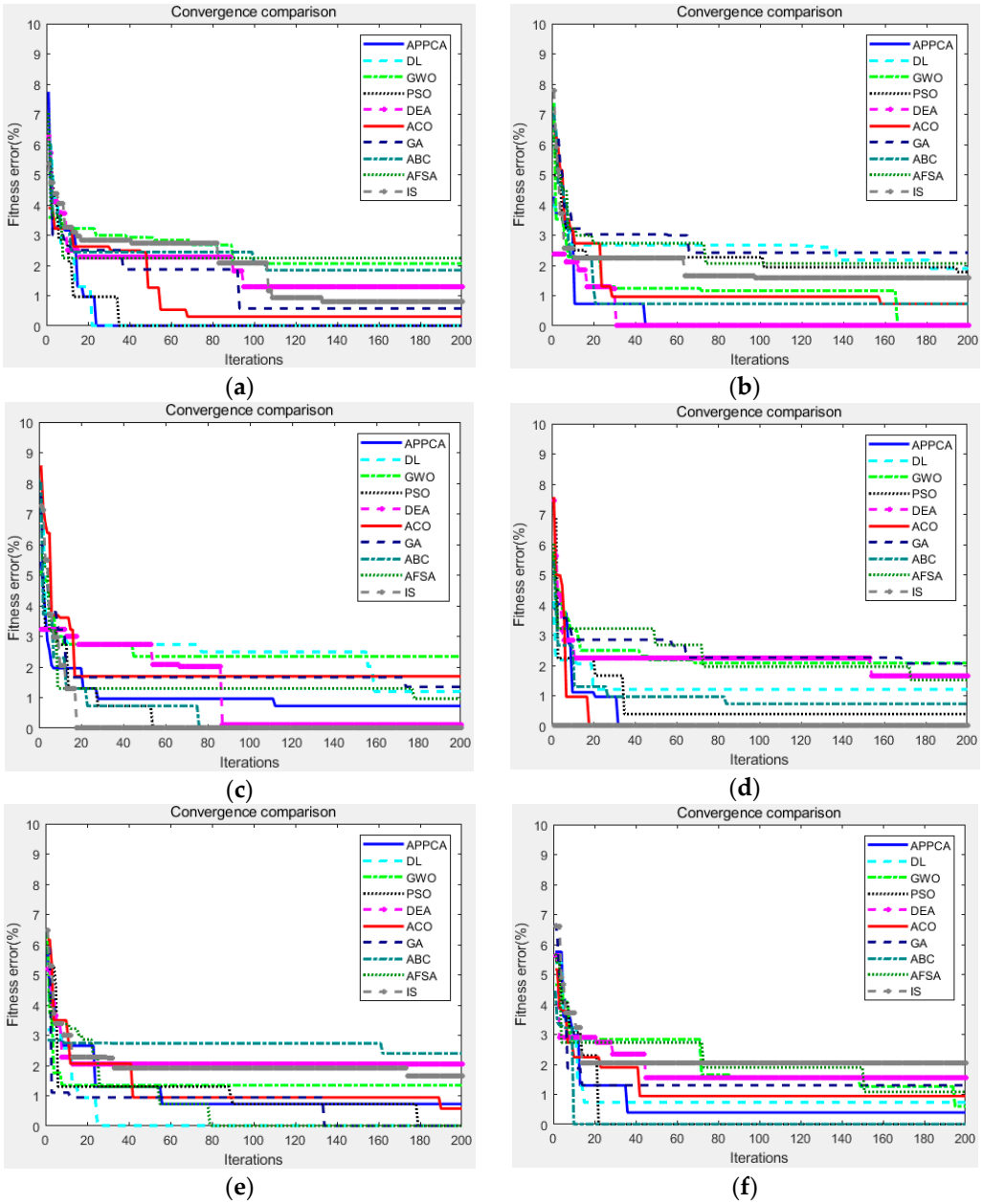


Figure 4. Convergence curves on Pascal VOC 2012 dataset. (**a**) image ID 2012_001155; (**b**) image ID 2012_003403; (**c**) image ID 2012_003438; (**d**) image ID 2012_004019; (**e**) image ID 2012_004134; (**f**) image ID 2012_004248.

Finally, the statistical comparison of the Pascal VOC 2012 dataset is shown in Table 2. Some metrics were explored for performance comparison, including *MSE* in Equation (6), *PSNR* in Equation (7), and *SSIM* in Equation (13). The optimal solution for each benchmark image is highlighted in bold. From Table 2, it is obvious that the proposed APPCA can obtain better performance in most performance metrics than other artificial intelligence algorithms, including three optimal solutions on image ID 2012_003403, 2012_004019, and 2012_004248. Other artificial intelligence algorithms have also obtained the optimal solution

with a certain probability; for example, DL [4,12,21,38] obtained two optimal solutions on image ID 2012_001155 and 2012_004134, and IS [44] achieved an optimal solution on image ID 2012_003438. At the same time, the APPCA did not achieve the highest *MSE* or the lowest *PSNR* or *SSIM*. Although other artificial intelligence algorithms have a small chance of achieving high-performance metrics, they are more likely to achieve the highest *MSE* or lowest *PSNR* or *SSIM*. The DL [4,12,21,38] algorithm performs well in computational accuracy, but it often requires more computational time. The increase in population size in the expansion stage helps the artificial *Physarum polycephalum* colony search for more solutions in a shorter time, and then the contraction operation helps the artificial *Physarum polycephalum* colony select the best solution with low *MSE*, high *PSNR*, and *SSIM*.

Table 2. Statistical comparison on Pascal VOC 2012 dataset.

Algorithms	Metrics	2012_001155	2012_003403	2012_003438	2012_004019	2012_004134	2012_004248
APPCA	*MSE*	395	**356**	379	**387**	443	**398**
	PSNR	22.16	**22.62**	22.34	**22.25**	21.67	**22.13**
	SSIM	0.63	**0.65**	0.65	**0.63**	0.59	**0.62**
DL	*MSE*	**376**	1214	1629	391	**405**	461
	PSNR	**22.38**	17.29	16.01	22.21	**22.06**	21.49
	SSIM	**0.64**	0.46	0.41	0.61	**0.61**	0.58
GWO	*MSE*	1602	2213	1007	2166	1211	856
	PSNR	16.08	14.68	18.10	14.77	17.30	18.81
	SSIM	0.42	0.36	0.48	0.36	0.48	0.48
PSO	*MSE*	739	383	476	502	722	2353
	PSNR	19.44	22.30	21.35	21.12	19.55	14.41
	SSIM	0.52	0.63	0.59	0.58	0.52	0.35
DEA	*MSE*	584	391	726	718	507	1829
	PSNR	20.47	22.21	19.52	19.57	21.08	15.51
	SSIM	0.56	0.61	0.53	0.54	0.57	0.41
ACO	*MSE*	827	745	636	2029	744	661
	PSNR	18.96	19.41	20.10	15.06	19.42	19.93
	SSIM	0.51	0.50	0.55	0.38	0.51	0.53
GA	*MSE*	549	574	373	712	588	2012
	PSNR	20.74	20.54	22.41	19.61	20.44	15.09
	SSIM	0.57	0.55	0.65	0.55	0.54	0.39
ABC	*MSE*	3216	4637	1405	2683	2556	409
	PSNR	13.06	11.47	16.65	13.84	14.06	22.01
	SSIM	0.32	0.28	0.45	0.33	0.41	0.61
AFSA	*MSE*	2134	3936	1982	2351	425	2439
	PSNR	14.84	12.18	15.16	14.42	21.85	14.26
	SSIM	0.35	0.33	0.39	0.34	0.60	0.35
IS	*MSE*	1482	4335	**347**	417	3875	746
	PSNR	16.42	11.76	**22.73**	21.93	12.25	19.40
	SSIM	0.43	0.30	**0.67**	0.60	0.31	0.51

5.3. Benchmark Test on Stanford Background Dataset

In this subsection, the Stanford background dataset was employed for the benchmark tests. It was selected from many famous databases, including Pascal VOC, LabelMe, MSRC, and Geometric Context. The Stanford background dataset includes eight categories and 715 outdoor images, including people, sky, forest, roads, trees, grass, water, mountains, and buildings. Figure 5 presents the experimental results on six original images from the Stanford background dataset. The image IDs of the six images are 0105146, 5000226, 6000141, 6000321, 9000002, and 9005294, and they contain 320 × 194 pixels, 320 × 213 pixels, 320 × 240 pixels, 320 × 240 pixels, 320 × 240 pixels, and 320 × 240 pixels, respectively.

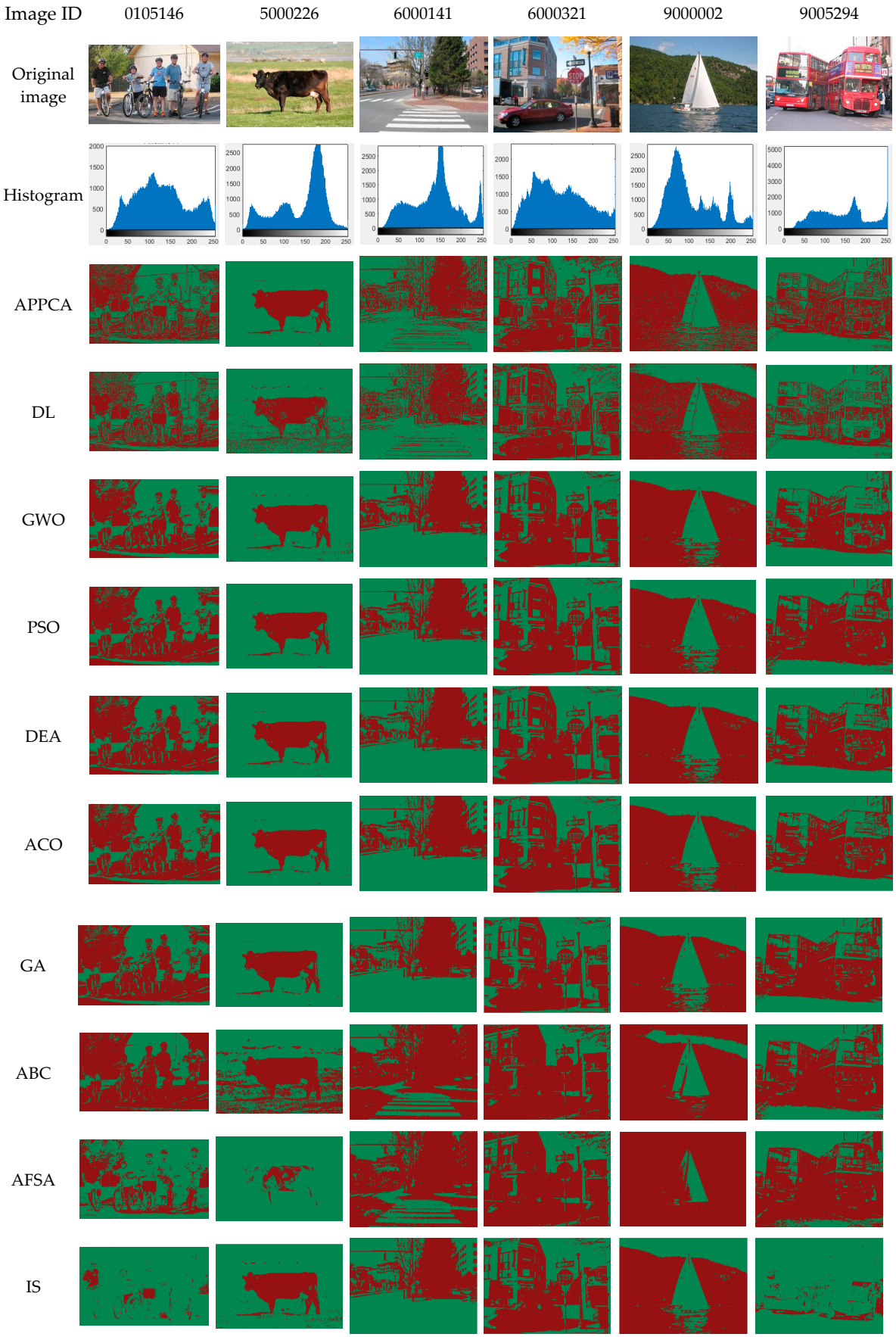
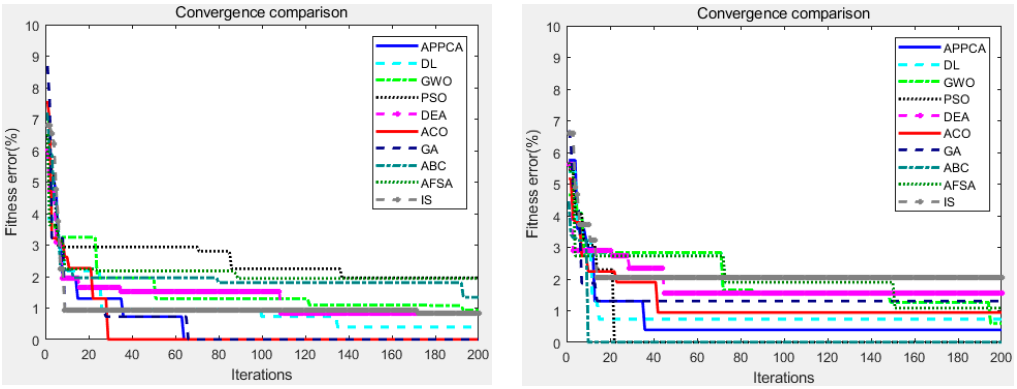


Figure 5. Benchmark test on Stanford background dataset.

Figure 5 shows benchmark test results on the Stanford background dataset, including the image ID on the dataset in the first line, the original images in the second line, the 2-D histogram in the third line, and the segmented images of different algorithms in the following lines, respectively. The same as the benchmark test in Section 5.2, all the algorithms were tested under the same conditions for comparison, and all algorithms ran at the same time. The benchmark images in the Stanford background dataset have smaller image sizes than those in the Pascal VOC 2012 segmentation dataset, so all the algorithms tended to need less time to obtain the optimal segmented solutions. At the end of the test, all algorithms can obtain similar experimental results in Figure 5.

After that, Figure 6 showed the fitness convergence curves of APPCA and other artificial intelligence algorithms on the same test images from the Stanford background dataset, where the maximum number of iterations was preset to 200. As we can see, APPCA obtains lower fitness errors than the other artificial intelligence algorithms in most cases. Due to the smaller image size, all algorithms can search for the optimal solution in a shorter time. With the aid of increased population size and more individuals for searching, APPCA is less likely to fall into local optima than other algorithms. According to the convergence curves in Figures 4 and 6, the proposed algorithm to some extent improves the disadvantage of traditional swarm intelligence algorithms that are prone to converge to local optima.

Finally, the statistical comparison of the Stanford background dataset is shown in Table 3. Some metrics were explored for performance comparison, including *MSE* in Equation (6), *PSNR* in Equation (7), and *SSIM* in Equation (13). The optimal solution for each benchmark image is highlighted in bold. Table 3 again verified that the proposed APPCA can obtain better performance in most performance metrics than other artificial intelligence algorithms, including three optimal solutions on image ID 0105146, 6000141, and 9000002. Other artificial intelligence algorithms have also obtained the optimal solution with a certain probability; for example, DL [4,12,21,38] obtained two optimal solutions on image ID 6000321 and 9005294, and PSO [7,23,40] achieved an optimal solution on image ID 5000226. At the same time, the APPCA did not achieve the highest *MSE* or the lowest *PSNR* or *SSIM*. Although other artificial intelligence algorithms have a small chance of achieving high-performance metrics, they are more likely to achieve the highest *MSE* or lowest *PSNR* or *SSIM*. The DL [4,12,21,38] algorithm performs well in computational accuracy, but it often requires more computational time. The increase in population size in the expansion stage helps the artificial *Physarum polycephalum* colony search for more solutions in a shorter time and prevents it from falling into local optima too early.



(**a**) (**b**)

Figure 6. *Cont.*

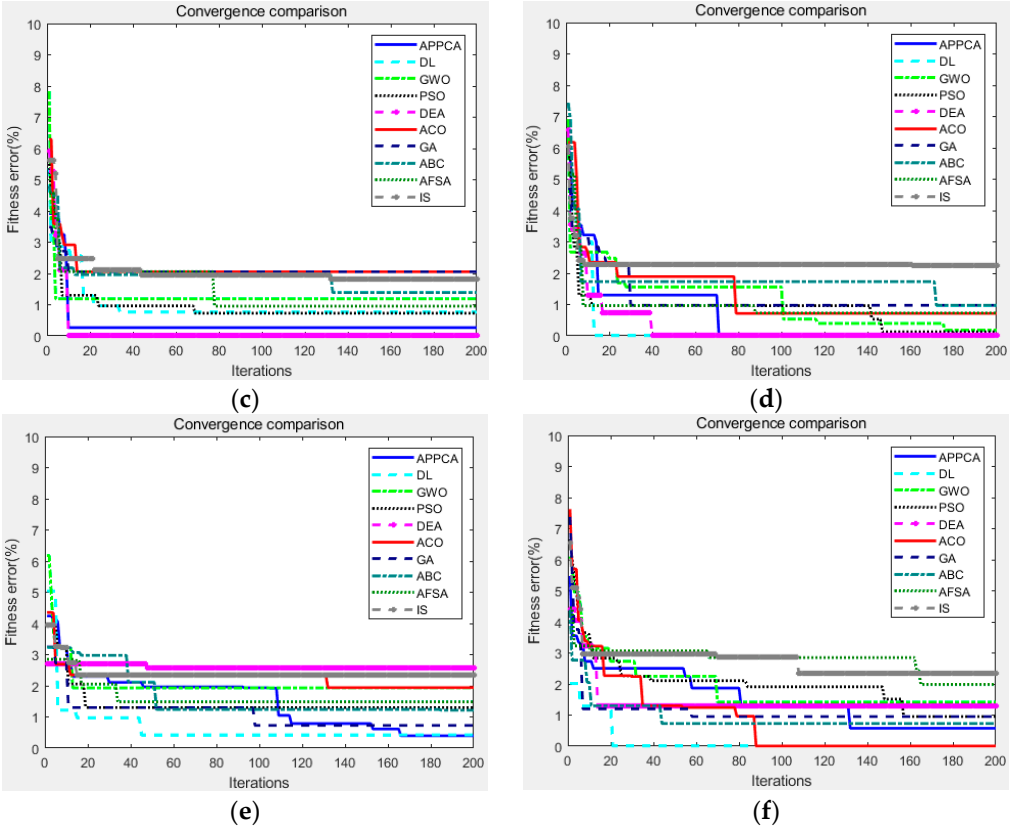


Figure 6. Convergence curves on Stanford background dataset. (**a**) image ID 0105146; (**b**) image ID 5,000,226; (**c**) image ID 6,000,141; (**d**) image ID 6,000,321; (**e**) image ID 9,000,002; (**f**) image ID 9,005,294.

5.4. Experiment Summary

According to the benchmark experimental results in Sections 5.2 and 5.3, it is verified that the proposed APPCA has some advantages in solving the threshold image segmentation problem.

First, the proposed APPC algorithm can use Kapur's entropy in Equation (5) as a fitness function to search for the optimal solutions to the threshold image segmentation problem. Kapur's entropy is useful to maximize the total entropy of the segmented target and background. The artificial *Physarum polycephalum* colony can employ the fitness function to instruct its expansion and contraction to search for the optimal segmented images.

Second, the variable population size of the proposed APPC algorithm can effectively improve its global search performance. In the expansion operation, the population size increases and more hyphae are produced to search for more feasible solutions by self-learning, social learning, and free learning. Three parts of hyphae can enhance global searching and fast convergence performance and obtain better segmentation performance than other artificial intelligence algorithms in most cases.

Third, the optimal solutions are easy to preserve through a quick sort algorithm in the contraction of the proposed APPC so as to improve its convergence performance. In each iteration, the APPC uses a quick sort algorithm to select the best hyphae with high fitness and the population size recovers. The contraction operation protects the optimal solutions after the expansion operation, but the other swarm intelligence algorithms are more likely to lose the optimal solution in each iteration. The expansion and contraction operations effectively solved the problem that the traditional swarm intelligence algorithm easily falls into the local optimal solution prematurely.

Fourth, the proposed algorithm has good robustness but traditional swarm intelligence algorithms tend to deteriorate rapidly with the expansion of the problem scale. Although

other artificial intelligence algorithms can obtain good segmentation performance, i.e., DL can obtain accurate segmented images in many cases, they will consume more time and more iterative computation. The APPC algorithm can obtain a good balance between the solving accuracy and solving time. The time performance and space performance of APPC are linearly correlated with the problem scale and computational parameters.

Hence, the APPC algorithm does not easily deteriorate the solution performance as the scale of the problem increases, and it has great potential for application to other NP-hard problems.

Table 3. Statistical comparison on Stanford background dataset.

Algorithms	Metrics	0105146	5000226	6000141	6000321	9000002	9005294
APPCA	*MSE*	**374**	377	**351**	396	**349**	382
	PSNR	**22.40**	22.37	**22.68**	22.15	**22.70**	22.31
	SSIM	**0.61**	0.64	**0.65**	0.61	**0.63**	0.62
DL	*MSE*	411	1616	370	**354**	368	**345**
	PSNR	21.99	16.05	22.45	**22.64**	22.47	**22.75**
	SSIM	0.59	0.46	0.62	**0.62**	0.61	**0.64**
GWO	*MSE*	477	615	1116	1252	720	768
	PSNR	21.35	20.24	17.65	17.15	19.56	19.28
	SSIM	0.57	0.51	0.46	0.47	0.52	0.51
PSO	*MSE*	1954	**342**	751	716	731	1462
	PSNR	15.22	**22.79**	19.37	19.58	19.49	16.48
	SSIM	0.39	**0.67**	0.53	0.53	0.53	0.42
DEA	*MSE*	1095	568	637	459	694	726
	PSNR	17.74	20.59	20.09	21.51	19.72	19.52
	SSIM	0.43	0.54	0.55	0.57	0.54	0.53
ACO	*MSE*	614	539	990	836	555	657
	PSNR	20.25	20.81	18.17	18.91	20.69	19.96
	SSIM	0.54	0.52	0.49	0.51	0.56	0.54
GA	*MSE*	927	425	429	702	572	1121
	PSNR	18.46	21.85	21.81	19.67	20.56	17.63
	SSIM	0.47	0.57	0.58	0.54	0.55	0.45
ABC	*MSE*	2628	2664	1711	1692	3272	1635
	PSNR	13.93	13.88	15.80	15.85	12.98	16.00
	SSIM	0.38	0.37	0.39	0.40	0.35	0.41
AFSA	*MSE*	3063	2941	1624	1570	4647	3343
	PSNR	13.27	13.45	16.02	16.17	11.46	12.89
	SSIM	0.36	0.33	0.44	0.43	0.28	0.34
IS	*MSE*	3590	946	865	955	673	3619
	PSNR	12.58	18.37	18.76	18.33	19.85	12.54
	SSIM	0.31	0.49	0.50	0.49	0.54	0.29

6. Conclusions

This paper proposed an artificial *Physarum polycephalum* colony algorithm to solve threshold image segmentation based on Kapur's entropy fitness function. From an engineering application perspective, the proposed APPCA has a variable population size and can obtain a faster convergence speed, better segmented results, and stronger robustness. In application, the APPCA and Kapur's entropy can be developed and implemented with just a regular personal computer and a general software platform. Kapur's entropy can instruct the artificial *Physarum polycephalum* colony to expand and contract to obtain good segmentation with high *PSNR* and low *MSE*. Our benchmark experiment results indicate that the expansion and contraction operations can help APPC improve search performance, and it is less likely to fall into local optima prematurely compared to other algorithms. The

quality of segmented images on benchmark datasets verifies that the APPCA has great potential for application in various image scenarios, such as medical image processing, autonomous navigation of robots, object recognition, plant detection in agricultural planting, pest and disease recognition, moving object detection, etc.

There are also some shortcomings in our work. First, the natural Physarum can optimize the foraging networks with millions of hyphae in a real parallel, while it is very difficult to implement on a classic computer. Second, our benchmark experiments are not complete, such as algorithm performance affected by computer software and hardware, the segmentation effect affected by the adjustment of algorithm parameters, and better quality for specific images through determined algorithms. Third, the entropy threshold method also has certain drawbacks, such as not considering the relationship between pixels, and the calculation results are affected by data quality.

In the future direction, the proposed algorithm should be further improved and more benchmark datasets should be tested. Some important factors should be further considered for analysis, i.e., improvement on Kapur's entropy, and the impact of the computer hardware and the algorithm parameters. In addition, the proposed APPC algorithm can also be combined with other algorithms to improve performance, such as deep learning algorithms.

Author Contributions: Conceptualization, Z.C. and G.L.; methodology, Z.C., G.L. and J.Z.; validation, Z.C., J.Z. and S.X.; writing—original draft preparation, G.L.; writing—review and editing, Z.C., G.L., J.Z. and S.X.; supervision, Z.C. and G.L.; project administration, Z.C. and G.L.; funding acquisition, Z.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 71471102), Major Science and Technology Projects in Hubei Province of China (Grant No. 2020AEA012), and Yichang University Applied Basic Research Project in China (Grant No. A17-302-a13).

Institutional Review Board Statement: This study did not require ethical approval.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data associated with this manuscript are contained in a data repository. Links to the supporting data are available: (1) Pascal VOC 2012 dataset https://pjreddie.com/projects/pascal-voc-dataset-mirror/ (accessed on 18 May 2012). (2) Stanford background dataset http://dags.stanford.edu/projects/scenedataset.html (accessed on 8 July 2009).

Acknowledgments: The authors thank the hard work of all anonymous reviewers to improve the quality of this submission.

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

References

1. Ning, G.Y. Two-dimensional Otsu multi-threshold image segmentation based on hybrid whale optimization algorithm. *Multimed. Tools Appl.* **2023**, *82*, 15007–15026. [CrossRef]
2. Ryalat, M.H.; Dorgham, O.; Tedmori, S.; Al-Rahamneh, Z.; Al-Najdawi, N.; Mirjalili, S. Harris hawks optimization for COVID-19 diagnosis based on multi-threshold image segmentation. *Neural Comput. Appl.* **2023**, *35*, 6855–6873. [CrossRef] [PubMed]
3. Palermo, G.; Raparelli, E.; Tuccella, P.; Orlandi, M.; Marzano, F.S. Using artificial neural networks to couple satellite c-band synthetic aperture radar interferometry and alpine3d numerical model for the estimation of snow cover extent, height, and density. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2023**, *16*, 2868–2888. [CrossRef]
4. He, C.; Liu, Y.L.; Wang, D.C.; Liu, S.F.; Yu, L.J.; Ren, Y.H. Automatic extraction of bare soil land from high-resolution remote sensing images based on semantic segmentation with deep learning. *Remote Sens.* **2023**, *15*, 1646. [CrossRef]
5. Liu, Y.S.; Kang, J.S.; Wen, L.; Bai, Y.J.; Guo, C.M.; Yu, W.B. Fault diagnosis algorithm of gearboxes based on GWO-SCE adaptive multi-threshold segmentation and subdomain adaptation. *Processes* **2023**, *11*, 556. [CrossRef]
6. Wang, Y.; Yao, Y.D. Application of artificial intelligence methods in carotid artery segmentation: A review. *IEEE Access* **2023**, *11*, 13846–13858. [CrossRef]
7. Chen, C.C.; Wang, X.C.; Heidari, A.A.; Yu, H.L.; Chen, H.L. Multi-threshold image segmentation of maize diseases based on elite comprehensive particle swarm optimization and Otsu. *Front. Plant Sci.* **2021**, *12*, 789911. [CrossRef]
8. Siri, S.K.; Kumar, S.P.; Latte, M.V. Threshold-based new segmentation model to separate the liver from CT scan images. *IETE J. Res.* **2022**, *68*, 4468–4475. [CrossRef]

9. Sun, Y.; Yang, Y.Y. An adaptive bi-mutation-based differential evolution algorithm for multi-threshold image segmentation. *Appl. Sci.* **2022**, *12*, 5759. [CrossRef]
10. Jiao, W.T.; Chen, W.Q.; Zhang, J. An improved cuckoo search algorithm for multithreshold image segmentation. *Secur. Commun. Netw.* **2021**, *2021*, 6036410. [CrossRef]
11. Zheng, Y.Y.; Chen, B.J.; Shen, Y.F.; Shen, K.D. TeethGNN: Semantic 3D teeth segmentation with graph neural networks. *IEEE Trans. Vis. Comput. Graph.* **2023**, *29*, 3158–3168. [CrossRef]
12. Zhang, H.; Zhu, P.M.; Liao, Z.Y. SaltISNet3D: Interactive salt segmentation from 3D seismic images using deep learning. *Remote Sens.* **2023**, *15*, 2319. [CrossRef]
13. Yang, X.; Ye, X.J.; Zhao, D.; Heidari, A.A.; Xu, Z.; Chen, H.; Li, Y. Multi-threshold image segmentation for melanoma based on Kapur's entropy using enhanced ant colony optimization. *Front. Neuroinform.* **2022**, *16*, 1041799. [CrossRef] [PubMed]
14. Xu, T.T.; Yao, L.J.; Xu, L.J.; Chen, Q.H.; Yang, Z.D. Image segmentation of cucumber seedlings based on genetic algorithm. *Sustainability* **2023**, *15*, 3089. [CrossRef]
15. Huo, F.C.; Liu, Y.; Wang, D.; Sun, B.X. Bloch quantum artificial bee colony algorithm and its application in image threshold segmentation. *Signal Image Video Process.* **2017**, *11*, 1585–1592. [CrossRef]
16. Wu, D.M.; Yuan, C.Z. Threshold image segmentation based on improved sparrow search algorithm. *Multimed. Tools Appl.* **2022**, *81*, 33513–33546. [CrossRef] [PubMed]
17. Cai, Z.; Xiong, Z.; Wan, K.; Xu, Y.; Xu, F. A node selecting approach for traffic network based on artificial slime mold. *IEEE Access* **2020**, *8*, 8436–8448. [CrossRef]
18. Cai, Z.; Yang, Y.; Zhang, X.; Zhou, Y. Design a robust logistics network with an artificial Physarum swarm algorithm. *Sustainability* **2022**, *14*, 14930. [CrossRef]
19. Long, W.; Zhang, Y.J.; Cui, Z.W.; Xu, Y.J.; Zhang, X.X. Threshold attention network for semantic segmentation of remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 4600312. [CrossRef]
20. Zhou, J.M.; Zhu, Q.; Wang, Y.A.; Zhou, X.E.; Feng, M.; Liu, X.; Mo, Y. Toward TR-PCB bubble detection via an efficient attention segmentation network and dynamic threshold. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 2506712. [CrossRef]
21. Li, X.; Wang, Z.Q.; Zhang, B.; Sun, F.C.; Hu, X.L. Recognizing object by components with human prior knowledge enhances adversarial robustness of deep neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 8861–8873. [CrossRef] [PubMed]
22. Tian, X.Y.; Lu, W.K.; Li, Y.D.; Liu, J.P.; Zhong, M.; Pan, H.; Jiang, B. Improved seismic residual diffracted multiple suppression method based on object detection and image segmentation. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 5901213. [CrossRef]
23. Zheng, J.F.; Gao, Y.C.; Zhang, H.; Lei, Y.; Zhang, J. Otsu multi-threshold image segmentation based on improved particle swarm algorithm. *Appl. Sci.* **2022**, *12*, 11514. [CrossRef]
24. Zhou, Y.Q.; Yang, X.; Ling, Y.; Zhang, J.Z. Meta-heuristic moth swarm algorithm for multilevel thresholding image segmentation. *Multimed. Tools Appl.* **2018**, *77*, 23699–23727. [CrossRef]
25. Huo, F.C.; Sun, X.T.; Ren, W.J. Multilevel image threshold segmentation using an improved Bloch quantum artificial bee colony algorithm. *Multimed. Tools Appl.* **2020**, *79*, 2447–2471. [CrossRef]
26. Subasree, S.; Sakthivel, N.K.; Balasaraswathi, V.R.; Tyagi, A.K. Selection of optimal thresholds in multi-level thresholding using multi-objective emperor penguin optimization for precise segmentation of mammogram images. *J. Circuits Syst. Comput.* **2022**, *31*, 2250131. [CrossRef]
27. Abualigah, L.; Al-Okbi, N.K.; Abd Elaziz, M.; Houssein, E.H. Boosting marine predators algorithm by salp swarm algorithm for multilevel thresholding image segmentation. *Multimed. Tools Appl.* **2022**, *81*, 16707–16742. [CrossRef] [PubMed]
28. Sharma, A.; Chaturvedi, R.; Bhargava, A. A novel opposition based improved firefly algorithm for multilevel image segmentation. *Multimed. Tools Appl.* **2022**, *81*, 15521–15544. [CrossRef]
29. Wang, C.Z.; Tu, C.K.; Wei, F.F. MSWOA: A mixed-strategy-based improved whale optimization algorithm for multilevel thresholding image segmentation. *Electronics* **2023**, *12*, 2698. [CrossRef]
30. Cai, Y.L.; Mi, S.H.; Yan, J.H.; Peng, H.; Luo, X.; Yang, Q.; Wang, J. An unsupervised segmentation method based on dynamic threshold neural P systems for color images. *Inf. Sci.* **2022**, *587*, 473–484. [CrossRef]
31. Cai, Z.; Zhang, Y.; Wu, M.; Cai, D. An entropy-robust optimization of mobile commerce system based on multi-agent system. *Arab. J. Sci. Eng.* **2016**, *41*, 3703–3715. [CrossRef]
32. Zhang, P.L.; Yang, J.N.; Lou, F.F.; Wang, J.Q.; Sun, X.B. Aptenodytes Forsteri optimization algorithm based on adaptive perturbation of oscillation and mutation operation for image multi-threshold segmentation. *Expert Syst. Appl.* **2023**, *224*, 120058. [CrossRef]
33. Zhao, X.Z.; Liu, L.; Heidari, A.A.; Chen, Y.; Ma, B.J.; Chen, H.; Quan, S. An enhanced ant colony optimizer with Cauchy-Gaussian fusion and novel movement strategy for multi-threshold COVID-19 X-ray image segmentation. *Front. Neuroinform.* **2023**, *17*, 1126783. [CrossRef] [PubMed]
34. Couto, P.; Bento, T.; Melo-Pinto, P. Positron emission tomography image segmentation based on Atanassov's intuitionistic fuzzy sets. *Appl. Sci.* **2022**, *12*, 4865. [CrossRef]
35. Lin, J.; Han, M.H.; Wu, Y.M. A novel defect segmentation for pulse infrared images based on improved fuzzy c-means algorithm with weighted distance. *IEEE Trans. Magn.* **2023**, *59*, 6200113. [CrossRef]
36. Yang, R.; Li, D.H. Adaptive wavelet transform based on artificial fish swarm optimization and fuzzy c-means method for noisy image segmentation. *Comput. Sci. Inf. Syst.* **2022**, *19*, 1389–1408. [CrossRef]

37. Gao, J.Q.; Wang, B.B.; Wang, Z.Y.; Wang, Y.F.; Kong, F.Z. A wavelet transform-based image segmentation method. *Optik* **2020**, *208*, 164123. [CrossRef]
38. Shi, G.H.; Zuo, B.H. CloudRCNN: A Framework based on deep neural networks for semantic segmentation of satellite cloud images. *Appl. Sci.* **2022**, *12*, 5370. [CrossRef]
39. Yu, X.B.; Wu, X.J. Ensemble grey wolf optimizer and its application for image segmentation. *Expert Syst. Appl.* **2022**, *209*, 118267. [CrossRef]
40. Kubicek, J.; Varysova, A.; Cerny, M.; Skandera, J.; Oczka, D.; Augustynek, M.; Penhaker, M. Novel hybrid optimized clustering schemes with genetic algorithm and PSO for segmentation and classification of articular cartilage loss from MR images. *Mathematics* **2023**, *11*, 1027. [CrossRef]
41. Cai, Z.; Ma, Z.; Zuo, Z.; Xiang, Y.; Wang, M. An image edge detection algorithm based on an artificial plant community. *Appl. Sci.* **2023**, *13*, 4159. [CrossRef]
42. Cai, Z.; Jiang, S.; Dong, J.; Tang, S. An artificial plant community algorithm for the accurate range-free positioning of wireless sensor networks. *Sensors* **2023**, *23*, 2804. [CrossRef] [PubMed]
43. Geng, Q.; Yan, H.F. Image segmentation under the optimization algorithm of krill swarm and machine learning. *Comput. Intell. Neurosci.* **2022**, *2022*, 8771650. [CrossRef] [PubMed]
44. Mabrouk, E.; Ayman, A.; Raslan, Y.; Hedar, A.R. Immune system programming for medical image segmentation. *J. Comput. Sci.* **2019**, *31*, 111–125. [CrossRef]
45. Singh, S.; Mittal, N.; Singh, U.; Salgotra, R.; Zaguia, A.; Singh, D. A novel hybrid tunicate swarm naked mole-rat algorithm for image segmentation and numerical optimization. *CMC-Comput. Mater. Contin.* **2022**, *71*, 3445–3462. [CrossRef]
46. Rao, Y.B.; Lv, Q.S.; Zeng, S.N.; Yi, Y.L.; Huang, C.; Gao, Y.; Cheng, Z.; Sun, J. COVID-19 CT ground-glass opacity segmentation based on attention mechanism threshold. *Biomed. Signal Process. Control* **2023**, *81*, 104486. [CrossRef] [PubMed]
47. Cai, Z.; Feng, Y.; Yang, S.; Yang, J. A state transition diagram and an artificial physarum polycephalum colony algorithm for the flexible job shop scheduling problem with transportation constraints. *Processes* **2023**, *11*, 2646. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.