Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 805357, 25 pages
http://dx.doi.org/10.1155/2015/805357

Research Article

Multithreshold Segmentation by Using an Algorithm Based on the Behavior of Locust Swarms

Erik Cuevas,[1,2] Adrián González,[1] Fernando Fausto,[1] Daniel Zaldívar,[1,2] and Marco Pérez-Cisneros[3]

[1]*Departamento de Electrónica, Universidad de Guadalajara, CUCEI, Avenida Revolución 1500, 44430 Guadalajara, JAL, Mexico*
[2]*Centro Tapatío Educativo AC, Vicente Guerrero 232, Colonia Zapopan Centro, 45100 Zapopan, JAL, Mexico*
[3]*Centro Universitario de Tonalá, Avenida Nuevo Periférico No. 555 Ejido San José Tatepozco, 48525 Tonalá, JAL, Mexico*

Correspondence should be addressed to Erik Cuevas; erik.cuevas@cucei.udg.mx

Received 7 October 2014; Revised 25 May 2015; Accepted 10 June 2015

Academic Editor: Bogdan Dumitrescu

As an alternative to classical techniques, the problem of image segmentation has also been handled through evolutionary methods. Recently, several algorithms based on evolutionary principles have been successfully applied to image segmentation with interesting performances. However, most of them maintain two important limitations: (1) they frequently obtain suboptimal results (misclassifications) as a consequence of an inappropriate balance between exploration and exploitation in their search strategies; (2) the number of classes is fixed and known in advance. This paper presents an algorithm for the automatic selection of pixel classes for image segmentation. The proposed method combines a novel evolutionary method with the definition of a new objective function that appropriately evaluates the segmentation quality with respect to the number of classes. The new evolutionary algorithm, called Locust Search (LS), is based on the behavior of swarms of locusts. Different to the most of existent evolutionary algorithms, it explicitly avoids the concentration of individuals in the best positions, avoiding critical flaws such as the premature convergence to suboptimal solutions and the limited exploration-exploitation balance. Experimental tests over several benchmark functions and images validate the efficiency of the proposed technique with regard to accuracy and robustness.

1. Introduction

Image segmentation [1] consists in grouping image pixels based on some criteria such as intensity, color, and texture and still represents a challenging problem within the field of image processing. Edge detection [2], region-based segmentation [3], and thresholding methods [4] are the most popular solutions for image segmentation problems.

Among such algorithms, thresholding is the simplest method. It works by considering threshold (points) values to adequately separate distinct pixels regions within the image being processed. In general, thresholding methods are divided into two types depending on the number of threshold values, namely, bilevel and multilevel. In bilevel thresholding, only a threshold value is required to separate the two objects of an image (e.g., foreground and background). On the other hand, multilevel thresholding divides pixels into more than two homogeneous classes that require several threshold values.

The thresholding methods use a parametric or nonparametric approach [5]. In parametric approaches [6, 7], it is necessary to estimate the parameters of a probability density function that is capable of modelling each class. A nonparametric technique [8–11] employs a given criteria such as the between-class variance or the entropy and error rate, in order to determine optimal threshold values.

A common method to accomplish parametric thresholding is the modeling of the image histogram through a Gaussian mixture model [12] whose parameters define a set of pixel classes (threshold points). Therefore, each pixel that belongs to a determined class is labeled according to its corresponding threshold points with several pixel groups gathering those pixels that share a homogeneous grayscale level.

The problem of estimating the parameters of a Gaussian mixture that better model an image histogram has been commonly solved through the Expectation Maximization (EM) algorithm [13, 14] or gradient-based methods such as Levenberg-Marquardt, LM [15]. Unfortunately, EM algorithms are very sensitive to the choice of the initial values [16], meanwhile gradient-based methods are computationally expensive and may easily get stuck within local minima [17].

As an alternative to classical techniques, the problem of Gaussian mixture identification has also been handled through evolutionary methods. In general, they have demonstrated to deliver better results than those based on classical approaches in terms of accuracy and robustness [18]. Under these methods, an individual is represented by a candidate Gaussian mixture model. Just as the evolution process unfolds, a set of evolutionary operators are applied in order to produce better individuals. The quality of each candidate solution is evaluated through an objective function whose final result represents the similarity between the mixture model and the histogram. Some examples of these approaches involve optimization methods such as Artificial Bee Colony (ABC) [19], Artificial Immune Systems (AIS) [20], Differential Evolution (DE) [21], Electromagnetism Optimization (EO) [22], Harmony Search (HS) [23], and Learning Automata (LA) [24]. Although these algorithms own interesting results, they present two important limitations. (1) They frequently obtain suboptimal approximations as a consequence of a limited balance between exploration and exploitation in their search strategies. (2) They are based on the assumption that the number of Gaussians (classes) in the mixture is preknown and fixed; otherwise, they cannot work. The cause of the first limitation is associated with their evolutionary operators employed to modify the individual positions. In such algorithms, during their evolution, the position of each agent for the next iteration is updated yielding an attraction towards the position of the best particle seen so far or towards other promising individuals. Therefore, as the algorithm evolves, these behaviors cause that the entire population rapidly concentrates around the best particles, favoring the premature convergence and damaging the appropriate exploration of the search space [25, 26]. The second limitation is produced as a consequence of the objective function that evaluates the similarity between the mixture model and the histogram. Under such an objective function, the number of Gaussians functions in the mixture is fixed. Since the number of threshold values (Gaussian functions) used for image segmentation varies depending on the image, the best threshold number and values are obtained by an exhaustive trial and error procedure.

On the other hand, bioinspired algorithms represent a field of research that is concerned with the use of biology as a metaphor for producing optimization algorithms. Such approaches use our scientific understanding of biological systems as an inspiration that, at some level of abstraction, can be represented as optimization processes.

In the last decade, several optimization algorithms have been developed by a combination of deterministic rules and randomness, mimicking the behavior of natural phenomena. Such methods include the social behavior of bird flocking and fish schooling such as the Particle Swarm Optimization (PSO) algorithm [27] and the emulation of the differential evolution in species such as the Differential Evolution (DE) [28]. Although PSO and DE are the most popular algorithms for solving complex optimization problems, they present serious flaws such as premature convergence and difficulty to overcome local minima [29, 30]. The cause for such problems is associated with the operators that modify individual positions. In such algorithms, during the evolution, the position of each agent for the next iteration is updated yielding an attraction towards the position of the best particle seen so far (in case of PSO) or towards other promising individuals (in case of DE). As the algorithm evolves, these behaviors cause that the entire population rapidly concentrates around the best particles, favoring the premature convergence and damaging the appropriate exploration of the search space [31, 32].

Recently, the collective intelligent behavior of insect or animal groups in nature has attracted the attention of researchers. The intelligent behavior observed in these groups provides survival advantages, where insect aggregations of relatively simple and "unintelligent" individuals can accomplish very complex tasks using only limited local information and simple rules of behavior [33]. Locusts (*Schistocerca gregaria*) are a representative example of such collaborative insects [34]. Locust is a kind of grasshopper that can change reversibly between a solitary and a social phase, with clear behavioral differences among both phases [35]. The two phases show many differences regarding the overall level of activity and the degree to which locusts are attracted or repulsed among them [36]. In the solitary phase, locusts avoid contact to each other (locust concentrations). As consequence, they distribute throughout the space, exploring sufficiently over the plantation [36]. On the other hand, in the social phase, locusts frantically concentrate around those elements that have already found good food sources [37]. Under such a behavior, locusts attempt to efficiently find better nutrients by devastating promising areas within the plantation.

This paper presents an algorithm for the automatic selection of pixel classes for image segmentation. The proposed method combines a novel evolutionary method with the definition of a new objective function that appropriately evaluates the segmentation quality with regard to the number of classes. The new evolutionary algorithm, called Locust Search (LS), is based on the behavior presented in swarms of locusts. In the proposed algorithm, individuals emulate a group of locusts which interact to each other based on the biological laws of the cooperative swarm. The algorithm considers two different behaviors: solitary and social. Depending on the behavior, each individual is conducted by a set of evolutionary operators which mimics different cooperative conducts that are typically found in the swarm. Different to most of existent evolutionary algorithms, the behavioral model in the proposed approach explicitly avoids the concentration of individuals in the current best positions. Such fact allows avoiding critical flaws such as the premature convergence to suboptimal solutions and the incorrect exploration-exploitation balance. In order to automatically define the optimal number of pixel classes

(Gaussian functions in the mixture), a new objective function has been also incorporated. The new objective function is divided into two parts. The first part evaluates the quality of each candidate solution in terms of its similarity with regard to the image histogram. The second part penalizes the overlapped area among Gaussian functions (classes). Under these circumstances, Gaussian functions that do not "positively" participate in the histogram approximation could be easily eliminated in the final Gaussian mixture model.

In order to illustrate the proficiency and robustness of the proposed approach, several numerical experiments have been conducted. Such experiments are divided into two sections. In the first part, the proposed LS method is compared to other well-known evolutionary techniques over a set of benchmark functions. In the second part, the performance of the proposed segmentation algorithm is compared to other segmentation methods which are also based on evolutionary principles. The results in both cases validate the efficiency of the proposed technique with regard to accuracy and robustness.

This paper is organized as follows: in Section 2 basic biological issues of the algorithm analogy are introduced and explained. Section 3 describes the novel LS algorithm and its characteristics. A numerical study on different benchmark function is presented in Section 4 while Section 5 presents the modelling of an image histogram through a Gaussian mixture. Section 6 exposes the LS segmentation algorithm and Section 7 the performance of the proposed segmentation algorithm. Finally, Section 8 draws some conclusions.

2. Biological Fundamentals and Mathematical Models

Social insect societies are complex cooperative systems that self-organize within a set of constraints. Cooperative groups are good at manipulating and exploiting their environment, defending resources and breeding, yet allowing task specialization among group members [38, 39]. A social insect colony functions as an integrated unit that not only possesses the ability to operate at a distributed manner but also undertakes a huge construction of global projects [40]. It is important to acknowledge that global order for insects can arise as a result of internal interactions among members.

Locusts are a kind of grasshoppers that exhibit two opposite behavioral phases: solitary and social (gregarious). Individuals in the solitary phase avoid contact to each other (locust concentrations). As consequence, they distribute throughout the space while sufficiently exploring the plantation [36]. In contrast, locusts in the gregarious phase gather into several concentrations. Such congregations may contain up to 10^{10} members, cover cross-sectional areas of up to 10 km^2, and a travelling capacity up to 10 km per day for a period of days or weeks as they feed causing a devastating crop loss [41]. The mechanism to switch from the solitary phase to the gregarious phase is complex and has been a subject of significant biological inquiry. Recently, a set of factors has been implicated to include geometry of the vegetation landscape and the olfactory stimulus [42].

Only few works [36, 37] that mathematically model the locust behavior have been published. Both approaches develop two different minimal models with the goal of reproducing the macroscopic structure and motion of a group of locusts. Considering that the method in [36] focuses on modelling the behavior for each locust in the group, its fundamentals have been employed to develop the algorithm that is proposed in this paper.

2.1. Solitary Phase. This section describes how each locust's position is modified as a result of its behavior under the solitary phase. Considering that $\mathbf{x}_i^k$ represents the current position of the ith locust in a group of N different elements, the new position $\mathbf{x}_i^{k+1}$ is calculated by using the following model:

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \Delta\mathbf{x}_i, \tag{1}$$

with $\Delta\mathbf{x}_i$ corresponding to the change of position that is experimented by $\mathbf{x}_i^k$ as a consequence of its social interaction with all other elements in the group.

Two locusts in the solitary phase exert forces on each other according to basic biological principles of attraction and repulsion (see, e.g., [36]). Repulsion operates quite strongly over a short length scale in order to avoid concentrations. Attraction is weaker and operates over a longer length scale, providing the social force that is required to maintain the group's cohesion. Therefore, the strength of such social forces can be modeled by the following function:

$$s(r) = F \cdot e^{-r/L} - e^{-r}. \tag{2}$$

Here, r is a distance, F describes the strength of attraction, and L is the typical attractive length scale. We have scaled the time and the space coordinates so that the repulsive strength and length scale are both represented by the unity. We assume that $F < 1$ and $L > 1$ so that repulsion is stronger and features in a shorter-scale, while attraction is applied in a weaker and longer-scale; both facts are typical for social organisms [21]. The social force exerted by locust j over locust i is

$$\mathbf{s}_{ij} = s\left(r_{ij}\right) \cdot \mathbf{d}_{ij}, \tag{3}$$

where $r_{ij} = \|\mathbf{x}_j - \mathbf{x}_i\|$ is the distance between the two locusts and $\mathbf{d}_{ij} = (\mathbf{x}_j - \mathbf{x}_i)/r_{ij}$ is the unit vector pointing from $\mathbf{x}_i$ to $\mathbf{x}_j$. The total social force on each locust can be modeled as the superposition of all of the pairwise interactions:

$$\mathbf{S}_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{s}_{ij}. \tag{4}$$

The change of position $\Delta\mathbf{x}_i$ is modeled as the total social force experimented by $\mathbf{x}_i^k$ as the superposition of all of the pairwise interactions. Therefore, $\Delta\mathbf{x}_i$ is defined as follows:

$$\Delta\mathbf{x}_i = \mathbf{S}_i. \tag{5}$$

In order to illustrate the behavioral model under the solitary phase, Figure 1 presents an example, assuming a population of

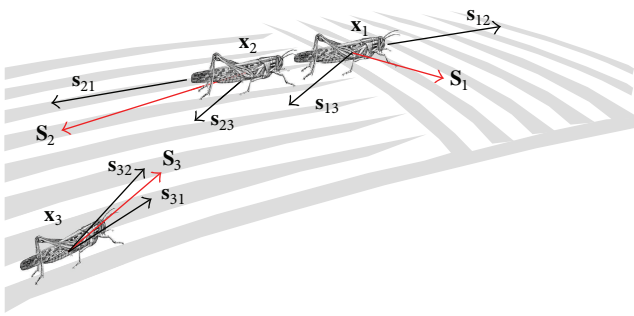


FIGURE 1: Behavioral model under the solitary phase.

three different members ($N = 3$) which adopt a determined configuration in the current iteration k. As a consequence of the social forces, each element suffers an attraction or repulsion to other elements depending on the distance among them. Such forces are represented by $\mathbf{s}_{12}$, $\mathbf{s}_{13}$, $\mathbf{s}_{21}$, $\mathbf{s}_{23}$, $\mathbf{s}_{31}$, and $\mathbf{s}_{32}$. Since $\mathbf{x}_1$ and $\mathbf{x}_2$ are too close, the social forces $\mathbf{s}_{12}$ and $\mathbf{s}_{13}$ present a repulsive nature. On the other hand, as the distances $\|\mathbf{x}_1 - \mathbf{x}_3\|$ and $\|\mathbf{x}_2 - \mathbf{x}_3\|$ are quite long, the social forces $\mathbf{s}_{13}$, $\mathbf{s}_{23}$, $\mathbf{s}_{31}$, and $\mathbf{s}_{32}$ between $\mathbf{x}_1 \leftrightarrow \mathbf{x}_3$ and $\mathbf{x}_2 \leftrightarrow \mathbf{x}_3$ all belong to the attractive nature. Therefore, the change of position $\Delta\mathbf{x}_1$ is computed as the vector resultant between $\mathbf{s}_{12}$ and $\mathbf{s}_{13}$ ($\Delta\mathbf{x}_1 = \mathbf{s}_{12} + \mathbf{s}_{13}$) is $\mathbf{S}_1$. The values $\Delta\mathbf{x}_2$ and $\Delta\mathbf{x}_3$ are also calculated accordingly.

In addition to the presented model [36], some studies [43–45] suggest that the social force $\mathbf{s}_{ij}$ is also affected by the dominance of the involved individuals $\mathbf{x}_i$ and $\mathbf{x}_j$ in the pairwise process. Dominance is a property that relatively qualifies the capacity of an individual to survive, in relation to other elements in a group. The locust's dominance is determined by several characteristics such as size, chemical emissions, and location with regard to food sources. Under such circumstances, the social force is magnified or weakened depending on the most dominant individual that is involved in the repulsion-attraction process.

2.2. Social Phase. In this phase, locusts frantically concentrate around the elements that have already found good food sources. They attempt to efficiently find better nutrients by devastating promising areas within the plantation. In order to simulate the social phase, the food quality index Fq_i is assigned to each locust $\mathbf{x}_i$ of the group as such index reflects the quality of the food source where $\mathbf{x}_i$ is located.

Under this behavioral model, each of the N elements of the group is ranked according to its corresponding food quality index. Afterwards, the b elements featuring the best food quality indexes are selected ($b \ll N$). Considering a concentration radius R_c that is created around each selected element, a set of c new locusts is randomly generated inside R_c. As a result, most of the locusts will be concentrated around the best b elements. Figure 2 shows a simple example of the behavioral model under the social phase. In the example, the configuration includes eight locusts ($N = 8$), just as it is illustrated by Figure 2(a) that also presents the food quality index for each locust. A food quality index near to one indicates a better food source. Therefore, Figure 2(b) presents the final configuration after the social phase, assuming $b = 2$.

3. The Locust Search (LS) Algorithm

In this paper, some behavioral principles drawn from a swarm of locusts have been used as guidelines for developing a new swarm optimization algorithm. The LS assumes that entire search space is a plantation, where all the locusts interact to each other. In the proposed approach, each solution within the search space represents a locust position inside the plantation. Every locust receives a food quality index according to the fitness value of the solution that is symbolized by the locust's position. As it has been previously discussed, the algorithm implements two different behavioral schemes: solitary and social. Depending on each behavioral phase, each individual is conducted by a set of evolutionary operators which mimics the different cooperative operations that are typically found in the swarm. The proposed method formulates the optimization problem in the following form:

$$\begin{aligned} &\text{maximize/minimize} \quad f(\mathbf{l}), \quad \mathbf{l} = (l_1, \ldots, l_n) \in \mathbb{R}^n \\ &\text{subject to} \quad \mathbf{l} \in \mathbf{S}, \end{aligned} \tag{6}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear function whereas $\mathbf{S} = \{\mathbf{l} \in \mathbb{R}^n \mid lb_d \le l_d \le ub_d,\ d = 1, \ldots, n\}$ is a bounded feasible search space, which is constrained by the lower (lb_d) and upper (ub_d) limits.

In order to solve the problem formulated in (6), a population $\mathbf{L}^k$ ($\{\mathbf{l}_1^k, \mathbf{l}_2^k, \ldots, \mathbf{l}_N^k\}$) of N locusts (individuals) is evolved inside the LS operation from the initial point ($k = 0$) to a total gen number of iterations ($k = gen$). Each locust $\mathbf{l}_i^k$ ($i \in [1, \ldots, N]$) represents an n-dimensional vector $\{l_{i,1}^k, l_{i,2}^k, \ldots, l_{i,n}^k\}$ where each dimension corresponds to a decision variable of the optimization problem to be solved. The set of decision variables constitutes the feasible search space $\mathbf{S} = \{\mathbf{l}_i^k \in \mathbb{R}^n \mid lb_d \le l_{i,d}^k \le ub_d\}$, where lb_d and ub_d correspond to the lower and upper bounds for the dimension d, respectively. The food quality index that is associated with each locust $\mathbf{l}_i^k$ (candidate solution) is evaluated through an objective function $f(\mathbf{l}_i^k)$ whose final result represents the fitness value of $\mathbf{l}_i^k$. In the LS algorithm, each iteration of the evolution process consists of two operators: (A) solitary and (B) social. Beginning by the solitary stage, the set of locusts is operated in order to sufficiently explore the search space. On the other hand, the social operation refines existent solutions within a determined neighborhood (exploitation) subspace.

3.1. Solitary Operation (A). One of the most interesting features of the proposed method is the use of the solitary operator to modify the current locust positions. Under this approach, locusts are displaced as a consequence of the social forces produced by the positional relations among the elements of the swarm. Therefore, near individuals tend to repel each other, avoiding the concentration of elements in regions. On the other hand, distant individuals tend to attract to each other, maintaining the cohesion of the swarm. A clear difference to the original model in [20] considers that social forces are also magnified or weakened depending on the most dominant (best fitness values) individuals that are involved in the repulsion-attraction process.

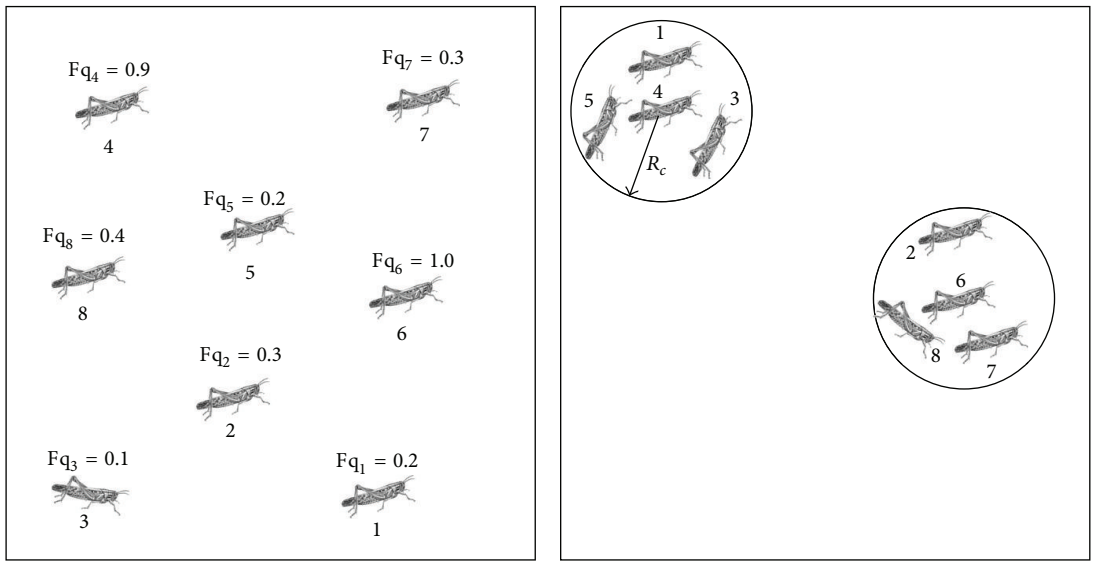


Figure 2: Behavioral model under the social phase. (a) Initial configuration and food quality indexes, (b) final configuration after the operation of the social phase.

In the solitary phase, a new position $\mathbf{p}_i (i \in [1, \ldots, N])$ is produced by perturbing the current locust position $\mathbf{l}_i^k$ with a change of position $\Delta\mathbf{l}_i (\mathbf{p}_i = \mathbf{l}_i^k + \Delta\mathbf{l}_i)$. The change of position $\Delta\mathbf{l}_i$ is the result of the social interactions experimented by $\mathbf{l}_i^k$ as a consequence of its repulsion-attraction behavioral model. Such social interactions are pairwise computed among $\mathbf{l}_i^k$ and the other $N - 1$ individuals in the swarm. In the original model, social forces are calculated by using (3). However, in the proposed method, it is modified to include the best fitness value (the most dominant) of the individuals involved in the repulsion-attraction process. Therefore, the social force, that is exerted between $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$, is calculated by using the following new model:

$$\mathbf{s}_{ij}^m = \rho\left(\mathbf{l}_i^k, \mathbf{l}_j^k\right) \cdot s\left(r_{ij}\right) \cdot \mathbf{d}_{ij} + \text{rand}\,(1, -1)\,, \tag{7}$$

where $s(r_{ij})$ is the social force strength defined in (2) and $\mathbf{d}_{ij} = (\mathbf{l}_j^k - \mathbf{l}_i^k)/r_{ij}$ is the unit vector pointing from $\mathbf{l}_i^k$ to $\mathbf{l}_j^k$. Besides, rand$(1, -1)$ is a randomly generated number between 1 and -1. Likewise, $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ is the dominance function that calculates the dominance value of the most dominant individual from $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$. In order to operate $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$, all the individuals from $\mathbf{L}^k$ $(\{\mathbf{l}_1^k, \mathbf{l}_2^k, \ldots, \mathbf{l}_N^k\})$ are ranked according to their fitness values. The ranks are assigned so that the best individual receives the rank 0 (zero) whereas the worst individual obtains the rank $N - 1$. Therefore, the function $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ is defined as follows:

$$\rho\left(\mathbf{l}_i^k, \mathbf{l}_j^k\right) = \begin{cases} e^{-(5 \cdot \text{rank}(\mathbf{l}_i^k)/N)} & \text{if rank}\left(\mathbf{l}_i^k\right) < \text{rank}\left(\mathbf{l}_j^k\right) \\ e^{-(5 \cdot \text{rank}(\mathbf{l}_j^k)/N)} & \text{if rank}\left(\mathbf{l}_i^k\right) > \text{rank}\left(\mathbf{l}_j^k\right), \end{cases} \tag{8}$$

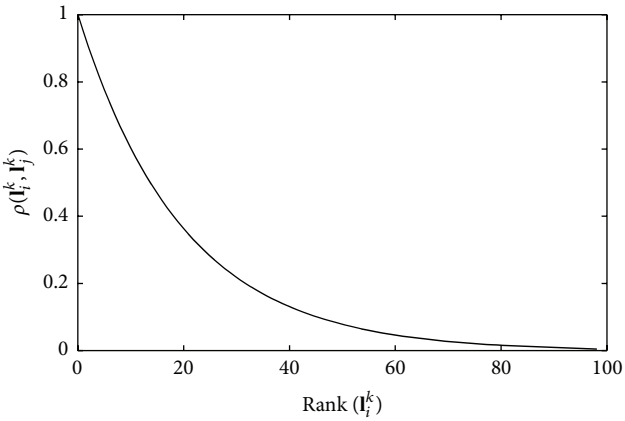


Figure 3: Behavior of $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ considering 100 individuals.

where the function rank(α) delivers the rank of the α-individual. According to (8), $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ yields a value within the interval $(1, 0)$. Its maximum value of one in $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ is reached when either individual $\mathbf{l}_j^k$ or $\mathbf{l}_i^k$ is the best element of the population $\mathbf{L}^k$ regarding their fitness values. On the other hand, a value close to zero is obtained when both individuals $\mathbf{l}_j^k$ and $\mathbf{l}_i^k$ possess quite bad fitness values. Figure 3 shows the behavior of $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ considering 100 individuals. In the figure, it is assumed that $\mathbf{l}_i^k$ represents one of the 99 individuals with ranks between 0 and 98 whereas $\mathbf{l}_j^k$ is fixed to the element with the worst fitness value (rank 99).

Under the incorporation of $\rho(\mathbf{l}_i^k, \mathbf{l}_j^k)$ in (7), social forces are magnified or weakened depending on the best fitness value (the most dominant) of the individuals involved in the repulsion-attraction process.

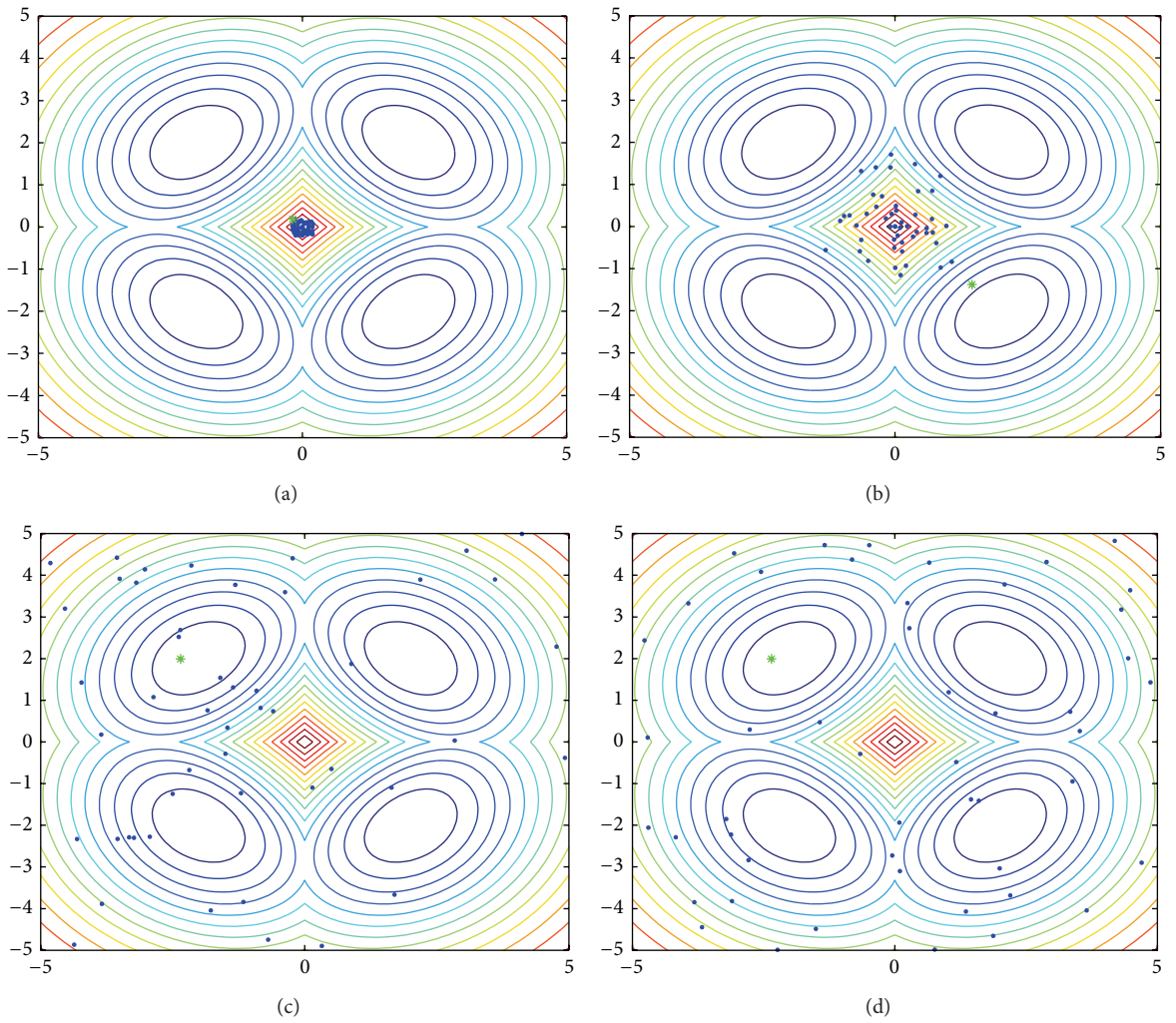


FIGURE 4: Examples of different distributions. (a) Initial condition, (b) distribution after applying 25, (c) 50 and (d) 100 operations. The green asterisk represents the minimum value so far.

Finally, the total social force on each individual $\mathbf{l}_i^k$ is modeled as the superposition of all of the pairwise interactions exerted over it:

$$\mathbf{S}_i^m = \sum_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{s}_{ij}^m. \tag{9}$$

Therefore, the change of position $\Delta\mathbf{l}_i$ is considered as the total social force experimented by $\mathbf{l}_i^k$ as the superposition of all of the pairwise interactions. Thus, $\Delta\mathbf{l}_i$ is defined as follows:

$$\Delta\mathbf{l}_i = \mathbf{S}_i^m. \tag{10}$$

After calculating the new positions $\mathbf{P}$ $(\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\})$ of the population $\mathbf{L}^k$ $(\{\mathbf{l}_1^k, \mathbf{l}_2^k, \ldots, \mathbf{l}_N^k\})$, the final positions $\mathbf{F}$ $(\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N\})$ must be calculated. The idea is to admit only the changes that guarantee an improvement in the search strategy. If the fitness value of $\mathbf{p}_i$ $(f(\mathbf{p}_i))$ is better than $\mathbf{l}_i^k(f(\mathbf{l}_i^k))$, then $\mathbf{p}_i$ is accepted as the final solution. Otherwise, $\mathbf{l}_i^k$ is retained. This procedure can be resumed by the following statement (considering a minimization problem):

$$\mathbf{f}_i = \begin{cases} \mathbf{p}_i & \text{if } f(\mathbf{p}_i) < f\left(\mathbf{l}_i^k\right) \\ \mathbf{l}_i^k & \text{otherwise.} \end{cases} \tag{11}$$

In order to illustrate the performance of the solitary operator, Figure 4 presents a simple example with the solitary operator being iteratively applied. A population of 50 different members ($N = 50$) is assumed which adopt a concentrated configuration as initial condition (Figure 4(a)). As a consequence of the social forces, the set of elements tends to distribute throughout the search space. Examples of different distributions are shown in Figures 4(b), 4(c), and 4(d) after applying 25, 50, and 100 different solitary operations, respectively.

3.2. Social Operation (B). The social procedure represents the exploitation phase of the LS algorithm. Exploitation is

the process of refining existent individuals within a small neighborhood in order to improve their solution quality.

The social procedure is a selective operation which is applied only to a subset $\mathbf{E}$ of the final positions $\mathbf{F}$ (where $\mathbf{E} \subseteq \mathbf{F}$). The operation starts by sorting $\mathbf{F}$ with respect to fitness values, storing the elements in a temporary population $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N\}$. The elements in $\mathbf{B}$ are sorted so that the best individual receives the position $\mathbf{b}_1$ whereas the worst individual obtains the location $\mathbf{b}_N$. Therefore, the subset $\mathbf{E}$ is integrated by only the first g locations of $\mathbf{B}$ (promising solutions). Under this operation, a subspace C_j is created around each selected particle $\mathbf{f}_j \in \mathbf{E}$. The size of C_j depends on the distance e_d which is defined as follows:

$$e_d = \frac{\sum_{q=1}^{n} \left(ub_q - lb_q\right)}{n} \cdot \beta, \tag{12}$$

where ub_q and lb_q are the upper and lower bounds in the qth dimension and n is the number of dimensions of the optimization problem, whereas $\beta \in [0, 1]$ is a tuning factor. Therefore, the limits of C_j can be modeled as follows:

$$\begin{aligned} uss_j^q &= b_{j,q} + e_d, \\ lss_j^q &= b_{j,q} - e_d, \end{aligned} \tag{13}$$

where uss_j^q and lss_j^q are the upper and lower bounds of the qth dimension for the subspace C_j, respectively.

Considering the subspace C_j around each element $\mathbf{f}_j \in \mathbf{E}$, a set of h new particles ($\mathbf{M}_j^h = \{\mathbf{m}_j^1, \mathbf{m}_j^2, \ldots, \mathbf{m}_j^h\}$) are randomly generated inside bounds fixed by (13). Once the h samples are generated, the individual $\mathbf{l}_j^{k+1}$ of the next population $\mathbf{L}^{k+1}$ must be created. In order to calculate $\mathbf{l}_j^{k+1}$, the best particle $\mathbf{m}_j^{best}$, in terms of its fitness value from the h samples (where $\mathbf{m}_j^{best} \in [\mathbf{m}_j^1, \mathbf{m}_j^2, \ldots, \mathbf{m}_j^h]$), is compared to $\mathbf{f}_j$. If $\mathbf{m}_j^{best}$ is better than $\mathbf{f}_j$ according to their fitness values, $\mathbf{l}_j^{k+1}$ is updated with $\mathbf{m}_j^{best}$; otherwise, $\mathbf{f}_j$ is selected. The elements of $\mathbf{F}$ that have not been processed by the procedure ($\mathbf{f}_w \notin \mathbf{E}$) transfer their corresponding values to $\mathbf{L}^{k+1}$ with no change.

The social operation is used to exploit only prominent solutions. According to the proposed method, inside each subspace C_j, h random samples are selected. Since the number of selected samples at each subspace is very small (typically $h < 4$), the use of this operator substantially reduces the number of fitness function evaluations.

In order to demonstrate the social operation, a numerical example has been set by applying the proposed process to a simple function. Such function considers the interval of $-3 \leq d_1, d_2 \leq 3$ whereas the function possesses one global maxima of value 8.1 at (0, 1.6). Notice that d_1 and d_2 correspond to the axis coordinates (commonly x and y). For this example, a final position population $\mathbf{F}$ of six 2-dimensional members ($N = 6$) is assumed. Figure 5 shows the initial configuration of the proposed example, with the black points representing half of the particles with the best fitness values (the first three elements of $\mathbf{B}$, $g = 3$) whereas the grey points ($\mathbf{f}_2, \mathbf{f}_4, \mathbf{f}_6 \notin \mathbf{E}$) correspond to the remaining individuals. From Figure 5, it

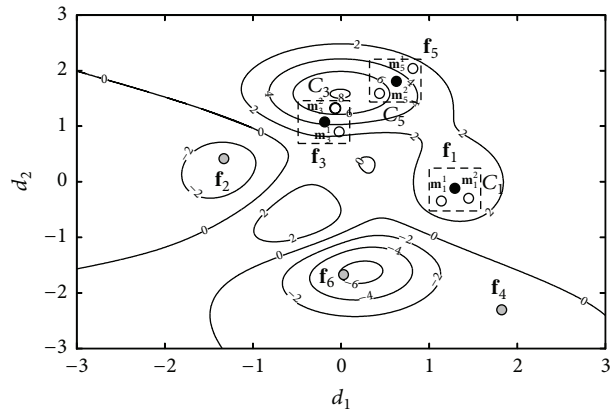


Figure 5: Operation of the social procedure.

can be seen that the social procedure is applied to all black particles ($\mathbf{f}_5 = \mathbf{b}_1$, $\mathbf{f}_3 = \mathbf{b}_2$, and $\mathbf{f}_1 = \mathbf{b}_3$, $\mathbf{f}_5, \mathbf{f}_3, \mathbf{f}_1 \in \mathbf{E}$) yielding two new random particles ($h = 2$), which are characterized by white points $\mathbf{m}_1^1$, $\mathbf{m}_1^2$, $\mathbf{m}_3^1$, $\mathbf{m}_3^2$, $\mathbf{m}_5^1$, and $\mathbf{m}_5^2$, for each black point inside of their corresponding subspaces C_1, C_3, and C_5. Considering the particle $\mathbf{f}_3$ in Figure 7, the particle $\mathbf{m}_3^2$ corresponds to the best particle ($\mathbf{m}_3^{best}$) from the two randomly generated particles (according to their fitness values) within C_3. Therefore, the particle $\mathbf{m}_3^{best}$ will substitute $\mathbf{f}_3$ in the individual $\mathbf{l}_3^{k+1}$ for the next generation, since it holds a better fitness value than $\mathbf{f}_3 (f(\mathbf{f}_3) < f(\mathbf{m}_3^{best}))$.

The LS optimization procedure is defined over a bounded search space $\mathbf{S}$. Search points that do not belong to such area are considered to be infeasible. However, during the evolution process, some candidate solutions could fall outside the search space. In the proposed approach, such infeasible solutions are arbitrarily placed with a random position inside the search space $\mathbf{S}$.

3.3. Complete LS Algorithm. LS is a simple algorithm with only seven adjustable parameters: the strength of attraction F, the attractive length L, number of promising solutions g, the population size N, the tuning factor β, the number of random samples h, and the number of generations gen. The operation of LS is divided into three parts: initialization of the solitary and social operations. In the initialization ($k = 0$), the first population $\mathbf{L}^0$ ($\{\mathbf{l}_1^0, \mathbf{l}_2^0, \ldots, \mathbf{l}_N^0\}$) is produced. The values $\{l_{i,1}^0, l_{i,2}^0, \ldots, l_{i,n}^0\}$ of each individual $\mathbf{l}_i^k$ and each dimension d are randomly and uniformly distributed between the prespecified lower initial parameter bound lb_d and the upper initial parameter bound ub_d:

$$\begin{aligned} l_{i,j}^0 &= lb_d + \text{rand} \cdot \left(ub_d - lb_d\right); \\ & i = 1, 2, \ldots, N; \; d = 1, 2, \ldots, n. \end{aligned} \tag{14}$$

In the evolution process, the solitary (A) and social (B) operations are iteratively applied until the number of iterations $k = gen$ has been reached. The complete LS procedure is illustrated in Algorithm 1.

(1) **Input**: F, L, g, N, gen, h and β.
(2) Initialize $\mathbf{L}^0$ ($k = 0$)
(3) **until** ($k = gen$)
(4) $\mathbf{F} \leftarrow$ SolitaryOperation($\mathbf{L}^k$) Solitary operator (Section 3.1)
(5) $\mathbf{L}^{k+1} \leftarrow$ SocialOperation($\mathbf{L}^k$, $\mathbf{F}$) Social operator (Section 3.2)
(6) $k = k + 1$
(7) **end until**

Algorithm 1: Locust Search (LS) algorithm.

3.4. Discussion about the LS Algorithm. Evolutionary algorithms (EA) have been widely employed for solving complex optimization problems. These methods are found to be more powerful than conventional methods that are based on formal logics or mathematical programming [46]. In the EA algorithm, search agents have to decide whether to explore unknown search positions or to exploit already tested positions in order to improve their solution quality. Pure exploration degrades the precision of the evolutionary process but increases its capacity to find new potentially solutions. On the other hand, pure exploitation allows refining existent solutions but adversely drives the process to local optimal solutions. Therefore, the ability of an EA to find a global optimal solution depends on its capacity to find a good balance between the exploitation of found-so-far elements and the exploration of the search space [47].

Most of swarm algorithms and other evolutionary algorithms tend to exclusively concentrate the individuals in the current best positions. Under such circumstances, such algorithms seriously limit their exploration-exploitation capacities.

Different to most of existent evolutionary algorithms, in the proposed approach, the modeled behavior explicitly avoids the concentration of individuals in the current best positions. Such fact allows not only to emulate the cooperative behavior of the locust colony in a good realistic way but also to incorporate a computational mechanism to avoid critical flaws that are commonly present in the popular evolutionary algorithms, such as the premature convergence and the incorrect exploration-exploitation balance.

It is important to emphasize that the proposed approach conducts two operators (solitary and social) within a single iteration. Such operators are similar to those that are used by other evolutionary methods such as ABC (employed bees, onlooker bees, and scout bees), AIS (clonal proliferation operator, affinity maturation operator, and clonal selection operator), and DE (mutation, crossover, and selection), which are all executed in a single iteration.

4. Numerical Experiments over Benchmark Functions

A comprehensive set of 13 functions, all collected from [48–50], has been used to test the performance of the LS approach as an optimization method. Tables 11 and 12 present the benchmark functions used in our experimental study. Such functions are classified into two different categories: unimodal test functions (Table 11) and multimodal test functions (Table 12). In these tables, n is the function dimension; f_{opt} is the minimum value of the function, with $\mathbf{S}$ being a subset of R^n. The optimum locations ($\mathbf{x}_{\text{opt}}$) for functions in Tables 11 and 12 are in $[0]^n$, except for f_5, f_{12}, and f_{13} with $\mathbf{x}_{\text{opt}}$ in $[1]^n$ and f_8 in $[420.96]^n$. A detailed description of optimum locations is given in Tables 11 and 12.

We have applied the LS algorithm to 13 functions whose results have been compared to those produced by the Particle Swarm Optimization (PSO) method [27] and the Differential Evolution (DE) algorithm [28], both considered as the most popular algorithms for many optimization applications. In all comparisons, the population has been set to 40 ($N = 40$) individuals. The maximum iteration number for all functions has been set to 1000. Such stop criterion has been selected to maintain compatibility to similar works reported in the literature [48, 49].

The parameter setting for each of the algorithms in the comparison is described as follows:

(1) PSO: in the algorithm, $c_1 = c_2 = 2$ while the inertia factor (ω) is decreased linearly from 0.9 to 0.2.

(2) DE: the DE/Rand/1 scheme is employed. The parameter settings follow the instructions in [28, 51]. The crossover probability is CR $= 0.9$ and the weighting factor is $F = 0.8$.

(3) In LS, F and L are set to 0.6 and L, respectively. Besides, g is fixed to 20 ($N/2$), $h = 2$, $\beta = 0.6$ whereas gen and N are configured to 1000 and 40, respectively. Once such parameters have been experimentally determined, they are kept for all experiments in this section.

In the comparison, three indexes are considered: the average best-so-far solution (ABS), the standard deviation (SD), and the number of function evaluations (NFE). The first two indexes assess the accuracy of the solution whereas the last one measures the computational cost. The average best-so-far solution (ABS) expresses the average value of the best function evaluations that have been obtained from 30 independent executions. The standard deviation (SD) indicates the dispersion of the ABS values. Evolutionary methods are, in general, complex pieces of software with several operators and stochastic branches. Therefore, it is difficult to conduct a complexity analysis from a deterministic perspective. Under such circumstances, it is more appropriate

Table 1: Minimization results from the benchmark functions test in Table 11 with $n = 30$. Maximum number of iterations = 1000.

		PSO	DE	LS
f_1	ABS	1.66×10^{-1}	6.27×10^{-3}	4.55×10^{-4}
	SD	3.79×10^{-1}	1.68×10^{-1}	6.98×10^{-4}
	NFE	28,610	20,534	16,780
f_2	ABS	4.83×10^{-1}	2.02×10^{-1}	5.41×10^{-3}
	SD	1.59×10^{-1}	0.66	1.45×10^{-2}
	NFE	28,745	21,112	16,324
f_3	ABS	2.75	5.72×10^{-1}	1.61×10^{-3}
	SD	1.01	0.15	1.32×10^{-3}
	NFE	38,320	36,894	20,462
f_4	ABS	1.84	0.11	1.05×10^{-2}
	SD	0.87	0.05	6.63×10^{-3}
	NFE	37,028	36,450	21,158
f_5	ABS	3.07	2.39	4.11×10^{-2}
	SD	0.42	0.36	2.74×10^{-3}
	NFE	39,432	37,264	21,678
f_6	ABS	6.36	6.51	5.88×10^{-2}
	SD	0.74	0.87	1.67×10^{-2}
	NFE	38,490	36,564	22,238
f_7	ABS	6.14	0.12	2.71×10^{-2}
	SD	0.73	0.02	1.18×10^{-2}
	NFE	37,274	35,486	21,842

to use the number of function evaluations (NFE), just as it is used in the literature [52, 53], to evaluate and assess the computational effort (time) and the complexity among optimizers. It represents how many times an algorithm uses the objective function to evaluate the objective (fitness) function until the best solution of a determined execution has been found. Since the experiments require 30 different executions, the NFE index corresponds to the averaged value obtained from these executions. A small NFE value indicates that less time is needed to reach the global optimum.

4.1. Unimodal Test Functions. Functions f_1 to f_7 are unimodal functions. The results for unimodal functions over 30 runs are reported in Table 1 considering the average best-so-far solution (ABS), the standard deviation (SD), and the number of function evaluations (NFE). According to this table, LS provides better results than PSO and DE for all functions in terms of ABS and SD. In particular, the test yields the largest performance difference in functions f_4–f_7. Such functions maintain a narrow curving valley that is hard to optimize, in case the search space cannot be explored properly and the direction changes cannot be kept up with [54]. For this reason, the performance differences are directly related to a better trade-off between exploration and exploitation that is produced by LS operators. In the practice, a main goal of an optimization algorithm is to find a solution as good as possible within a small time window. The computational cost for the optimizer is represented by its NFE values. According to Table 1, the NFE values that are obtained by the proposed method are smaller than its counterparts. Lower NFE values are more desirable since they correspond to less computational overload and, therefore, faster results. In the results perspective, it is clear that PSO and DE need more than 1000 generations in order to produce better solutions. However, this number of generations is considered in the experiments aiming for producing a visible contrast among the approaches. If the number of generations has been set to an exaggerated value, then all methods would converge to the best solution with no significant troubles.

Table 2: p values produced by Wilcoxon's test that compares LS versus PSO and DE over the "average best-so-far" values from Table 1.

LS versus	PSO	DE
f_1	1.83×10^{-4}	1.73×10^{-2}
f_2	3.85×10^{-3}	1.83×10^{-4}
f_3	1.73×10^{-4}	6.23×10^{-3}
f_4	2.57×10^{-4}	5.21×10^{-3}
f_5	4.73×10^{-4}	1.83×10^{-3}
f_6	6.39×10^{-5}	2.15×10^{-3}
f_7	1.83×10^{-4}	2.21×10^{-3}

A nonparametric statistical significance proof known as Wilcoxon's rank sum test for independent samples [55, 56] has been conducted with an 5% significance level, over the "average best-so-far" data of Table 1. Table 2 reports the p values produced by Wilcoxon's test for the pairwise comparison of the "average best-so-far" of two groups. Such groups are formed by LS versus PSO and LS versus DE. As a null hypothesis, it is assumed that there is no significant difference between mean values of the two algorithms. The alternative hypothesis considers a significant difference between the "average best-so-far" values of both approaches. All p values reported in the table are less than 0.05 (5% significance level) which is a strong evidence against the null hypothesis, indicating that the LS results are statistically significant and that it has not occurred by coincidence (i.e., due to the normal noise contained in the process).

4.2. Multimodal Test Functions. Multimodal functions possess many local minima which make the optimization a difficult task to be accomplished. In multimodal functions, the results reflect the algorithm's ability to escape from local optima. We have applied the algorithms over functions f_8 to f_{13} where the number of local minima increases exponentially as the dimension of the function increases. The dimension of such functions is set to 30. The results are averaged over 30 runs, with performance indexes being reported in Table 3 as follows: the average best-so-far solution (ABS), the standard deviation (SD), and the number of function evaluations (NFE). Likewise, p values of the Wilcoxon signed-rank test of 30 independent runs are listed in Table 4. From the results, it is clear that LS yields better solutions than others algorithms for functions f_9, f_{10}, f_{11}, and f_{12}, in terms of the indexes ABS and SD. However, for functions f_8 and f_{13}, LS produces similar results to DE. The Wilcoxon rank test results, that are presented in Table 6, confirm that

Table 3: Minimization results from the benchmark functions test in Table 12 with $n = 30$. Maximum number of iterations = 1000.

		PSO	DE	LS
f_8	ABS	-6.7×10^3	-1.26×10^4	-1.26×10^4
	SD	6.3×10^2	3.7×10^2	1.1×10^2
	NFE	38,452	35,240	21,846
f_9	ABS	14.8	4.01×10^{-1}	2.49×10^{-3}
	SD	1.39	5.1×10^{-2}	4.8×10^{-4}
	NFE	37,672	34,576	20,784
f_{10}	ABS	14.7	4.66×10^{-2}	2.15×10^{-3}
	SD	1.44	1.27×10^{-2}	3.18×10^{-4}
	NFE	39,475	37,080	21,235
f_{11}	ABS	12.01	1.15	1.47×10^{-4}
	SD	3.12	0.06	1.48×10^{-5}
	NFE	38,542	34,875	22,126
f_{12}	ABS	6.87×10^{-1}	3.74×10^{-1}	5.58×10^{-3}
	SD	7.07×10^{-1}	1.55×10^{-1}	4.18×10^{-4}
	NFE	35,248	30,540	16,984
f_{13}	ABS	1.87×10^{-1}	1.81×10^{-2}	1.78×10^{-2}
	SD	5.74×10^{-1}	1.66×10^{-2}	1.64×10^{-3}
	NFE	36,022	31,968	18,802

Table 4: p values produced by Wilcoxon's test comparing LS versus PSO and DE over the "average best-so-far" values from Table 3.

LS versus	PSO	DE
f_8	1.83×10^{-4}	0.061
f_9	1.17×10^{-4}	2.41×10^{-4}
f_{10}	1.43×10^{-4}	3.12×10^{-3}
f_{11}	6.25×10^{-4}	1.14×10^{-3}
f_{12}	2.34×10^{-5}	7.15×10^{-4}
f_{13}	4.73×10^{-4}	0.071

LS performed better than PSO and DE considering four problems f_9–f_{12}, whereas, from a statistical viewpoint, there is no difference between results from LS and DE for f_8 and f_{13}. According to Table 3, the NFE values obtained by the proposed method are smaller than those produced by other optimizers. The reason of this remarkable performance is associated with its two operators: (i) the solitary operator allows a better particle distribution in the search space, increasing the algorithm's ability to find the global optima and (ii) the use of the social operation provides a simple exploitation operator that intensifies the capacity of finding better solutions during the evolution process.

5. Gaussian Mixture Modelling

In this section, the modeling of image histograms through Gaussian mixture models is presented. Let one consider an image holding L gray levels $[0, \ldots, L-1]$ whose distribution is defined by a histogram $h(g)$ represented by the following formulation:

$$h(g) = \frac{n_g}{Np}, \quad h(g) > 0,$$
$$Np = \sum_{g=0}^{L-1} n_g, \tag{15}$$
$$\sum_{g=0}^{L-1} h(g) = 1,$$

where n_g denotes the number of pixels with gray level g and Np the total number of pixels in the image. Under such circumstances, $h(g)$ can be modeled by using a mixture $p(x)$ of Gaussian functions of the form:

$$p(x) = \sum_{i=1}^{K} \frac{P_i}{\sqrt{2\pi}\sigma_i} \exp\left[\frac{-(x-\mu_i)^2}{2\sigma_i^2}\right], \tag{16}$$

where K symbolizes the number of Gaussian functions of the model whereas P_i is the a priori probability of function i. μ_i and σ_i represent the mean and standard deviation of the ith Gaussian function, respectively. Furthermore, the constraint $\sum_{i=1}^{K} P_i = 1$ must be satisfied by the model. In order to evaluate the similarity between the image histogram and a candidate mixture model, the mean square error can be used as follows:

$$J = \frac{1}{n}\sum_{j=1}^{L}\left[p(x_j) - h(x_j)\right]^2 + \omega \cdot \left|\left(\sum_{i=1}^{K} P_i\right) - 1\right|, \tag{17}$$

where ω represents the penalty associated with the constrain $\sum_{i=1}^{K} P_i = 1$. Therefore, J is considered as the objective function which must be minimized in the estimation problem. In order to illustrate the histogram modeling through a Gaussian mixture, Figure 6 presents an example, assuming three classes, that is, $K = 3$. Considering Figure 6(a) as the image histogram $h(x)$, the Gaussian mixture $p(x)$, that is shown in Figure 6(c), is produced by adding the Gaussian functions $p_1(x)$, $p_2(x)$, and $p_3(x)$ in the configuration presented in Figure 6(b). Once the model parameters that better model the image histogram have been determined, the final step is to define the threshold values T_i ($i \in [1, \ldots, K]$) which can be calculated by simple standard methods, just as it is presented in [19–21].

6. Segmentation Algorithm Based on LS

In the proposed method, the segmentation process is approached as an optimization problem. Computational optimization generally involves two distinct elements: (1) a search strategy to produce candidate solutions (individuals, particles, insects, locust, etc.) and (2) an objective function that evaluates the quality of each selected candidate solution. Several computational algorithms are available to perform the first element. The second element, where the objective function is designed, is unquestionably the most critical. In the objective function, it is expected to embody the full

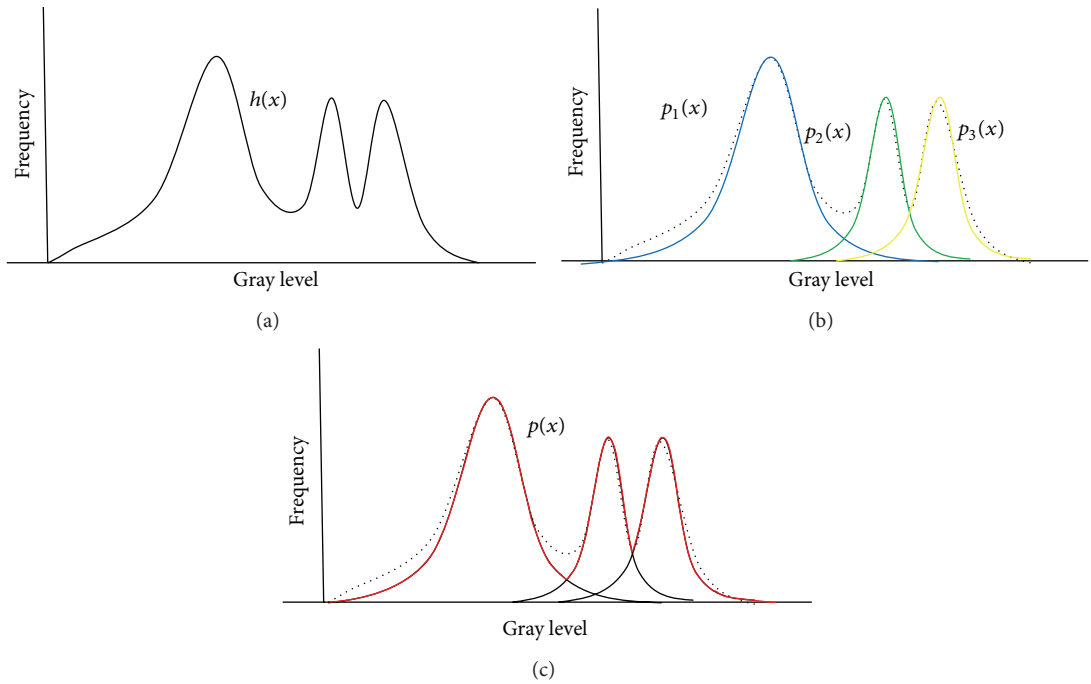


FIGURE 6: Histogram approximation through a Gaussian mixture. (a) Original histogram, (b) configuration of the Gaussian components $p_1(x)$, $p_2(x)$, and $p_3(x)$, and (c) final Gaussian mixture $p(x)$.

complexity of the performance, biases, and restrictions of the problem to be solved. In the segmentation problem, candidate solutions represent Gaussian mixtures. The objective function J (17) is used as a fitness value to evaluate the similarity presented between the Gaussian mixture and the image histogram. Therefore, guided by the fitness values (J values), a set of encoded candidate solutions are evolved using the evolutionary operators until the best possible resemblance can be found.

Over the last decade, several algorithms based on evolutionary and swarm principles [19–22] have been proposed to solve the problem of segmentation through a Gaussian mixture model. Although these algorithms own certain good performance indexes, they present two important limitations. (1) They frequently obtain suboptimal approximations as a consequence of an inappropriate balance between exploration and exploitation in their search strategies. (2) They are based on the assumption that the number of Gaussians (classes) in the mixture is preknown and fixed; otherwise, they cannot work.

In order to eliminate such flaws, the proposed approach includes (A) a new search strategy and (B) the definition of a new objective function. For the search strategy, the LS method (Section 4) is adopted. Under LS, the concentration of individuals in the current best positions is explicitly avoided. Such fact allows reducing critical problems such as the premature convergence to suboptimal solutions and the incorrect exploration-exploitation balance.

6.1. New Objective Function J^{new}. Previous segmentation algorithms based on evolutionary and swarm methods use (17) as objective function. Under these circumstances, each candidate solution (Gaussian mixture) is only evaluated in terms of its approximation with the image histogram.

Since the proposed approach aims to automatically select the number of Gaussian functions K in the final mixture $p(x)$, the objective function must be modified. The new objective function J^{new} is defined as follows:

$$J^{\text{new}} = J + \lambda \cdot Q, \tag{18}$$

where λ is a scaling constant. The new objective function is divided into two parts. The first part J evaluates the quality of each candidate solution in terms of its similarity with regard to the image histogram (17). The second part Q penalizes the overlapped area among Gaussian functions (classes), with Q being defined as follows:

$$Q = \sum_{i=1}^{K} \sum_{\substack{j=1 \\ j \neq i}}^{K} \sum_{l=1}^{L} \min\left(P_i \cdot p_i(l), P_j \cdot p_j(l)\right), \tag{19}$$

where K and L represent the number of classes and the gray levels, respectively. The parameters $p_i(l)$ and $p_j(l)$ symbolize the Gaussian functions i and j, respectively, that are to be evaluated on the point l (gray level) whereas the elements P_i and P_j represent the a priori probabilities of the Gaussian functions i and j, respectively. Under such circumstances, mixtures with Gaussian functions that do not "positively" participate in the histogram approximation are severely penalized.

Figure 7 illustrates the effect of the new objective function J^{new} in the evaluation of Gaussian mixtures (candidate solutions). From the image histogram (Figure 7(a)), it is evident that two Gaussian functions are enough to accurately

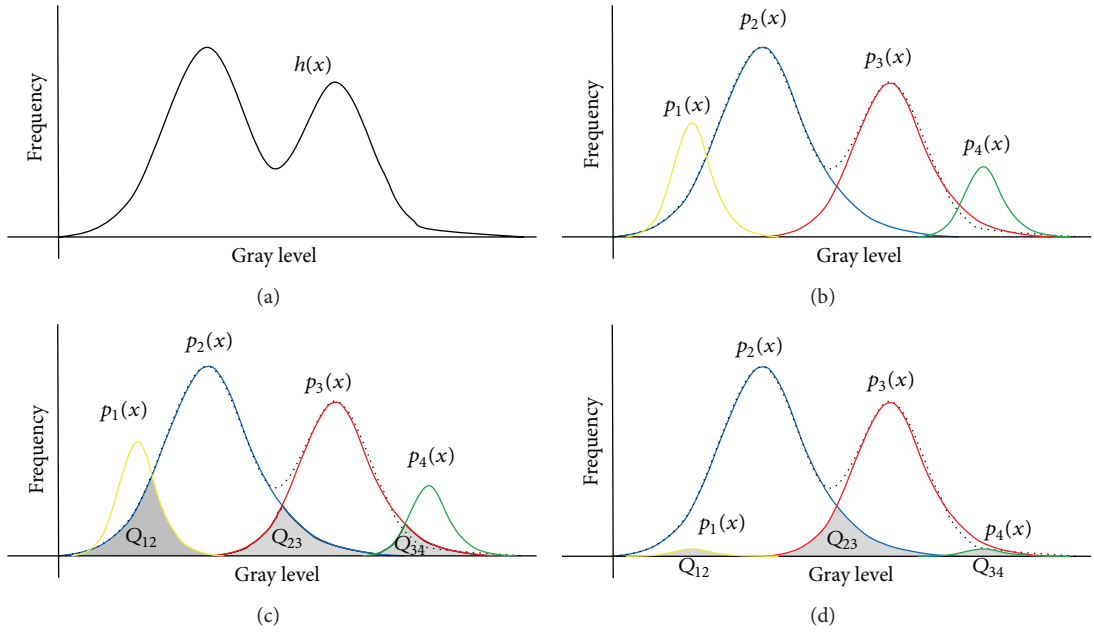


FIGURE 7: Effect of the new objective function J^{new} in the evaluation of Gaussian mixtures (candidate solutions). (a) Original histogram, (b) Gaussian mixture considering four classes, (c) penalization areas, and (d) Gaussian mixture of better quality solution.

approximate the original histogram. However, if the Gaussian mixture is modeled by using a greater number of functions (e.g., four as it is shown in Figure 7(b)), the original objective function J is unable to obtain a reasonable approximation. Under the new objective function J^{new}, the overlapped area among Gaussian functions (classes) is penalized. Such areas, in Figure 7(c), correspond to Q_{12}, Q_{23}, and Q_{34}, where Q_{12} represents the penalization value produced between the Gaussian function $p_1(x)$ and $p_2(x)$. Therefore, due to the penalization, the Gaussian mixture shown in Figures 7(b) and 7(c) provides a solution of low quality. On the other hand, the Gaussian mixture presented in Figure 7(d) maintains a low penalty; thus, it represents a solution of high quality. From Figure 7(d), it is easy to see that functions $p_1(x)$ and $p_4(x)$ can be removed from the final mixture. This elimination could be performed by a simple comparison with a threshold value θ, since $p_1(x)$ and $p_4(x)$ have a reduced amplitude ($p_1(x) \approx p_2(x) \approx 0$). Therefore, under J^{new}, it is possible to find the reduced Gaussian mixture model starting from a considerable number of functions.

Since the proposed segmentation method is conceived as an optimization problem, the overall operation can be reduced to solve the formulation of (20) by using the LS algorithm:

$$\begin{aligned} \text{minimize} \quad & J^{\text{new}}(\mathbf{x}) = J(\mathbf{x}) + \lambda \cdot Q(\mathbf{x}), \\ & \mathbf{x} = (P_1, \mu_1, \sigma_1, \ldots, P_K, \mu_K, \sigma_K) \in \mathbb{R}^{3 \cdot K} \\ \text{subject to} \quad & 0 \leq P_d \leq 1, \quad d \in (1, \ldots, K) \\ & 0 \leq \mu_d \leq 255 \\ & 0 \leq \sigma_d \leq 25, \end{aligned} \tag{20}$$

where P_d, μ_d, and σ_d represent the probability, mean, and standard deviation of the class d. It is important to remark that the new objective function J^{new} allows the evaluation of a candidate solution in terms of the necessary number of Gaussian functions and its approximation quality. Under such circumstances, it can be used in combination with any other evolutionary method and not only with the proposed LS algorithm.

6.2. Complete Segmentation Algorithm. Once the new search strategy (LS) and objective function (J^{new}) have been defined, the proposed segmentation algorithm can be summarized by Algorithm 2. The new algorithm combines operators defined by LS and operations for calculating the threshold values.

(Line 1) The algorithm sets the operative parameters F, L, g, gen, N, K, λ, and θ. They rule the behavior of the segmentation algorithm. (Line 2) Afterwards, the population $\mathbf{L}^0$ is initialized considering N different random Gaussian mixtures of K functions. The idea is to generate an N-random Gaussian mixture subjected to the constraints formulated in (20). The parameter K must hold a high value in order to correctly segment all images (recall that the algorithm is able to reduce the Gaussian mixture to its minimum expression). (Line 3) Then, the Gaussian mixtures are evolved by using the LS operators and the new objective function J^{new}. This process is repeated during gen cycles. (Line 8) After this procedure, the best Gaussian mixture $\mathbf{l}_{best}^{gen}$ is selected according to its objective function J^{new}. (Line 9) Then, the Gaussian mixture $\mathbf{l}_{best}^{gen}$ is reduced by eliminating those functions whose amplitudes are lower than θ ($p_i(x) \leq \theta$). (Line 10) Then, the threshold values T_i from the reduced model are calculated. (Line 11) Finally, the calculated T_i values are employed to segment the image. Figure 8 shows a flowchart of the complete segmentation procedure.

(1) **Input**: F, L, g, gen, N, K, λ and θ.
(2) Initialize $\mathbf{L}^0$ ($k = 0$)
(3) **until** ($k = gen$)
(4) $\mathbf{F} \leftarrow$ SolitaryOperation($\mathbf{L}^k$) Solitary operator (Section 3.1)
(5) $\mathbf{L}^{k+1} \leftarrow$ SocialOperation($\mathbf{L}^k$, $\mathbf{F}$) Social operator (Section 3.2)
(6) $k = k + 1$
(7) **end until**
(8) Obtain $\mathbf{I}_{best}^{gen}$
(9) Reduce $\mathbf{I}_{best}^{gen}$
(10) Calculate the threshold values T_i from the reduced model
(11) Use T_i to segment the image

Algorithm 2: Segmentation LS algorithm.

The proposed segmentation algorithm permits to automatically detect the number of segmentation partitions (classes). Furthermore, due to its remarkable search capacities, the LS method maintains a better accuracy than previous algorithms based on evolutionary principles. However, the proposed method presents two disadvantages: first, it is related to its implementation which in general is more complex than most of the other segmentators based on evolutionary basics. The second refers to the segmentation procedure of the proposed approach which does not consider any spatial pixel characteristics. As a consequence, pixels that may belong to a determined region due to its position are labeled as a part of another region due to its gray level intensity. Such a fact adversely affects the segmentation performance of the method.

7. Segmentation Results

This section analyses the performance of the proposed segmentation algorithm. The discussion is divided into three parts: the first one shows the performance of the proposed LS segmentation algorithm while the second presents a comparison between the proposed approach and others segmentation algorithms that are based on evolutionary and swam methods. The comparison mainly considers the capacities of each algorithm to accurately and robustly approximate the image histogram. Finally, the third part presents an objective evaluation of segmentation results produced by all algorithms that have been employed in the comparisons.

7.1. Performance of LS Algorithm in Image Segmentation. This section presents two experiments that analyze the performance of the proposed approach. Table 5 presents the algorithm's parameters that have been experimentally determined and kept for all the test images through all experiments.

First Image. The first test considers the histogram shown by Figure 9(b) while Figure 9(a) presents the original image. After applying the proposed algorithm, just as it has been configured according to parameters in Table 5, a minimum model of four classes emerges after the start from Gaussian mixtures of 10 classes. Considering 30 independent executions, the averaged parameters of the resultant Gaussian mixture are presented in Table 6. One final Gaussian mixture (ten classes), which has been obtained by LS, is presented in Figure 10. Furthermore, the approximation of the reduced Gaussian mixture is also visually compared with the original histogram in Figure 10. On the other hand, Figure 11 presents the segmented image after calculating the threshold points.

Table 5: Parameter setup for the LS segmentation algorithm.

F	L	g	gen	N	K	λ	θ
0.6	0.2	20	1000	40	10	0.01	0.0001

Table 6: Results of the reduced Gaussian mixture for the first and the second image.

Parameters	First image	Second image
$\overline{P_1}$	0.004	0.032
$\overline{\mu_1}$	18.1	12.1
$\overline{\sigma_1}$	8.2	2.9
$\overline{P_2}$	0.0035	0.0015
$\overline{\mu_2}$	69.9	45.1
$\overline{\sigma_2}$	18.4	24.9
$\overline{P_3}$	0.01	0.006
$\overline{\mu_3}$	94.9	130.1
$\overline{\sigma_3}$	8.9	17.9
$\overline{P_4}$	0.007	0.02
$\overline{\mu_4}$	163.1	167.2
$\overline{\sigma_4}$	29.9	10.1

Second Image. For the second experiment, the image in Figure 12 is tested. The method aims to segment the image by using a reduced Gaussian mixture model obtained by the LS approach. After executing the algorithm according to parameters from Table 5, the resulting averaged parameters of the resultant Gaussian mixture are presented in Table 6. In order to assure consistency, the results are averaged considering 30 independent executions. Figure 13 shows the approximation quality that is obtained by the reduced Gaussian mixture model in (a) and the segmented image in (b).

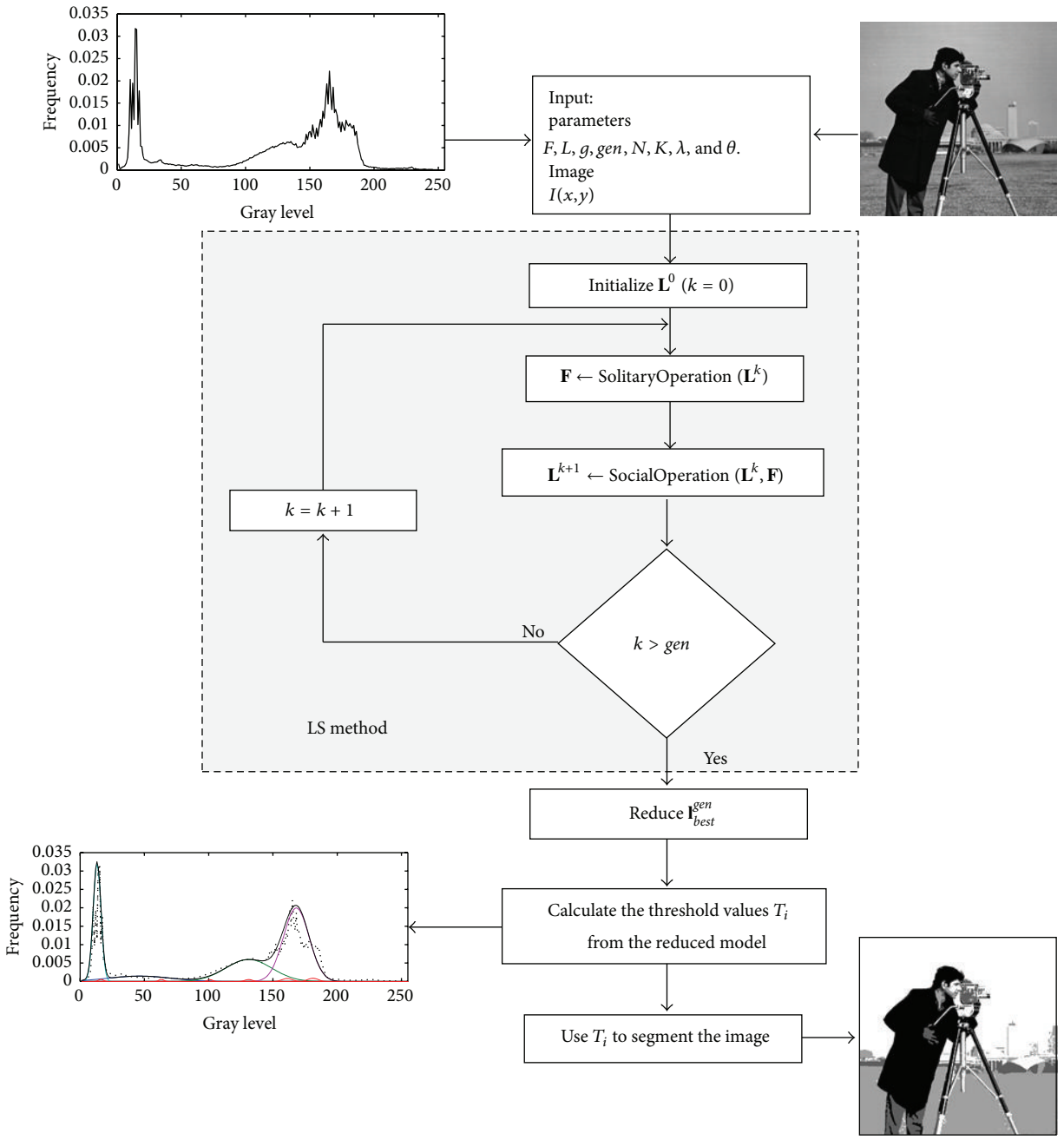
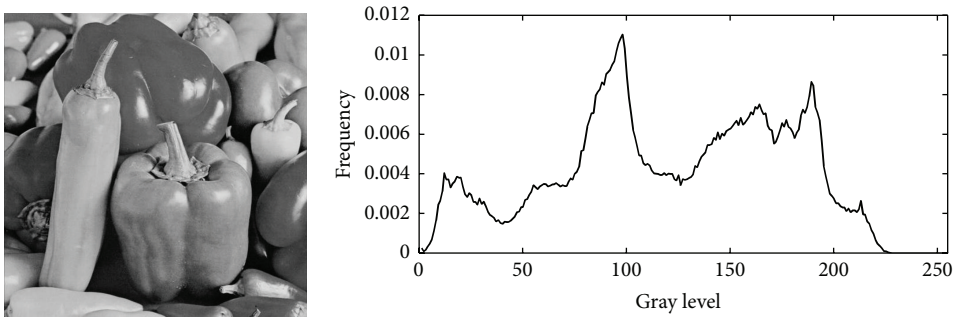


Figure 8: Flowchart of the complete segmentation procedure.



(a) (b)

Figure 9: (a) Original first image used on the first experiment and (b) its histogram.

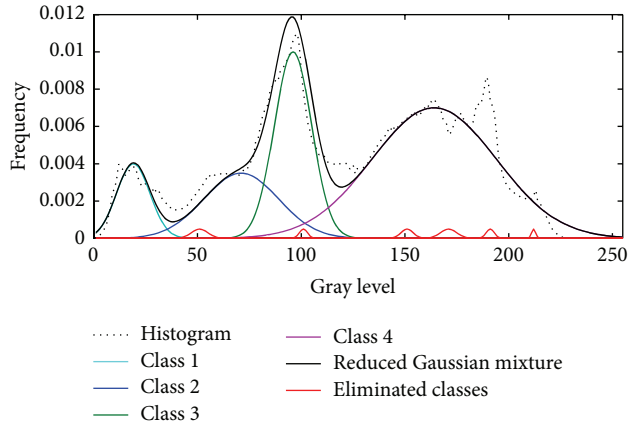
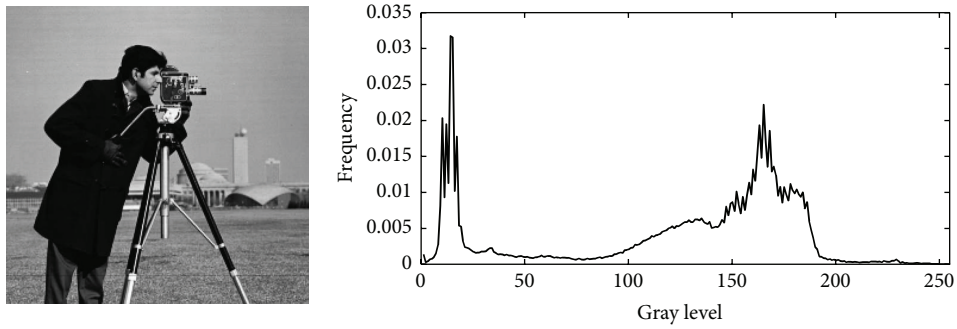


Figure 10: Gaussian mixture obtained by LS for the first image.

Figure 11: Image segmented with the reduced Gaussian mixture.



(a) (b)

Figure 12: (a) Original second image used on the first experiment and (b) its histogram.

7.2. Histogram Approximation Comparisons. This section discusses the comparison between the proposed segmentation algorithm and other evolutionary-segmentation methods that have been proposed in the literature. The set of methods used in the experiments includes the J + ABC [19], J + AIS [20], and J + DE [21]. These algorithms consider the combination between the original objective function J (17) and an evolutionary technique such as Artificial Bee Colony (ABC), the Artificial Immune Systems (AIS), and the Differential Evolution (DE) [21], respectively. Since the proposed segmentation approach considers the combination of the new objective function J^{new} (18) and the LS algorithm, it will be referred to as J^{new} + LS. The comparison focuses mainly on the capacities of each algorithm to accurately and robustly approximate the image histogram.

In the experiments, the populations have been set to 40 ($N = 40$) individuals. The maximum iteration number for all functions has been set to 1000. Such stop criterion has been considered to maintain compatibility to similar experiments

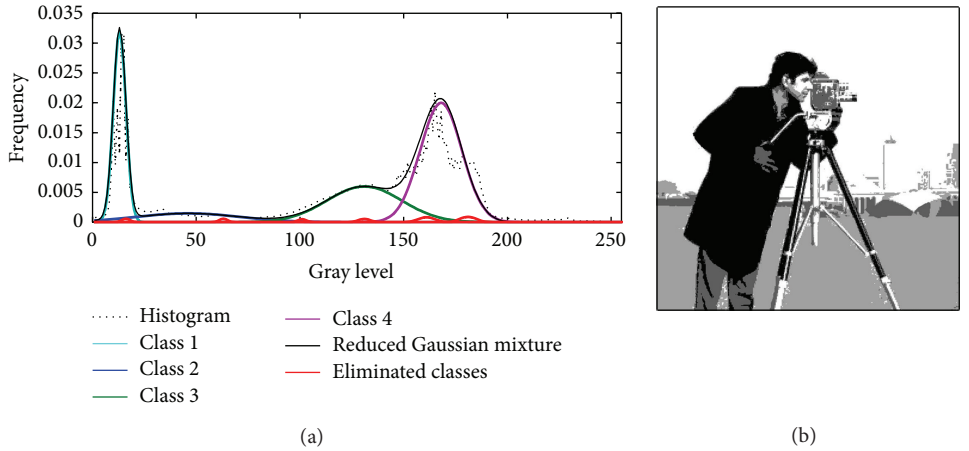


Figure 13: (a) Segmentation result obtained by LS for the first image and (b) the segmented image.

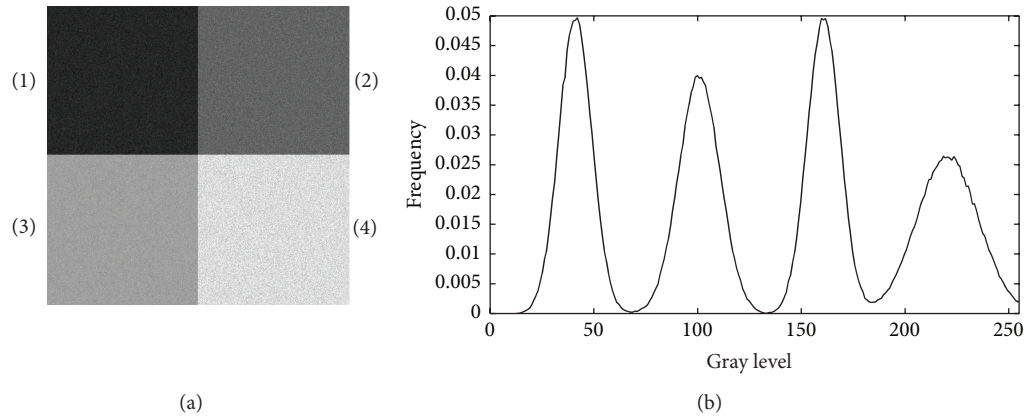


Figure 14: (a) Synthetic image used in the comparison and (b) its histogram.

that are reported in the literature [18]. The parameter setting for each of the segmentation algorithms in the comparison is described as follows:

(1) J + ABC [19]: in the algorithm, its parameters are configured as follows: the abandonment limit = 100, $\alpha = 0.05$ and limit = 30.

(2) J + AIS [20]: it presents the following parameters, $h = 120$, $N_c = 80$, $\rho = 10$, $P_r = 20$, $L = 22$, and $T_e = 0.01$.

(3) J + DE [21]: the DE/Rand/1 scheme is employed. The parameter settings follow the instructions in [21]. The crossover probability is CR = 0.9 and the weighting factor is $F = 0.8$.

(4) In J^{new} + LS, the method is set according to the values described in Table 5.

In order to conduct the experiments, a synthetic image is designed to be used as a reference in the comparisons. The main idea is to know in advance the exact number of classes (and their parameters) that are contained in the image so that the histogram can be considered as a ground truth. The synthetic image is divided into four sections. Each section corresponds to a different class which is produced by setting each gray level pixel Pv^i to a value that is determined by the following model:

$$Pv_i = e^{-((x-\mu_1)^2/2\sigma_i^2)}, \tag{21}$$

where i represents the section, whereas μ_i and σ_i are the mean and the dispersion of the gray level pixels, respectively. The comparative study employs the image of 512 × 512 that is shown in Figure 14(a) and the algorithm's parameters that have been presented in Table 7. Figure 14(b) illustrates the resultant histogram.

Table 7: Employed parameters for the design of the reference image.

	P_i	μ_i	σ_i
(1)	0.05	40	8
(2)	0.04	100	10
(3)	0.05	160	8
(4)	0.027	220	15

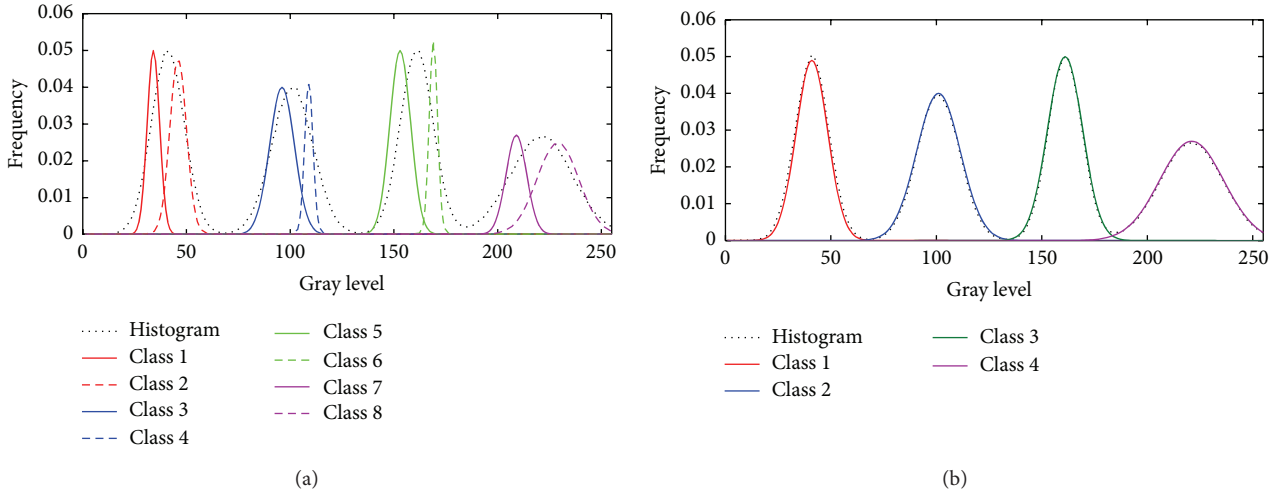


FIGURE 15: Convergence results. (a) Convergence of the following methods: J + ABC, J + AIS, and J + DE considering Gaussian mixtures of 8 classes. (b) Convergence of the proposed method (reduced Gaussian mixture).

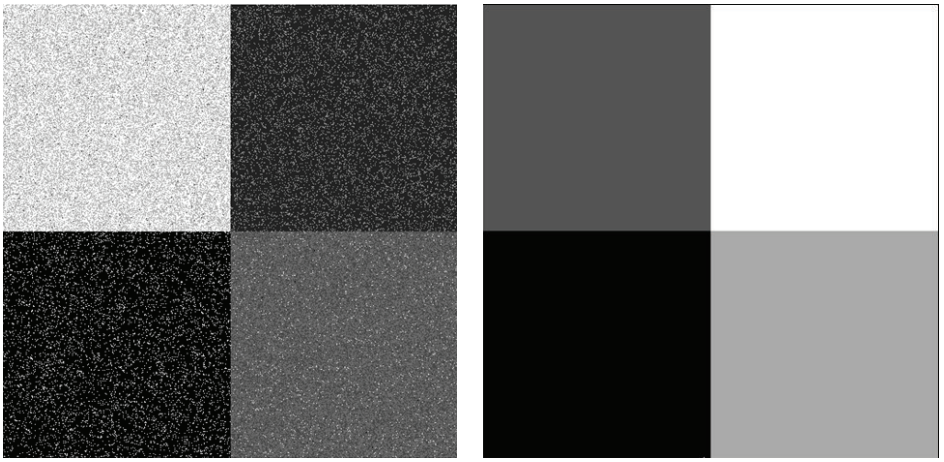


FIGURE 16: Segmentation results obtained by (a) several methods including J + ABC, J + AIS, and J + DE considering Gaussian mixtures of 8 classes and (b) the proposed method (reduced Gaussian mixture).

In the comparison, the discussion focuses on the following issues: first of all, accuracy; second, convergence; and third, computational cost.

Convergence. This section analyzes the approximation convergence when the number of classes that are used by the algorithm during the evolution process is different to the actual number of classes in the image. Recall that the proposed approach automatically finds the reduced Gaussian mixture which better adapts to the image histogram.

In the experiment, the methods, J + ABC, J + AIS, and J + DE, are executed considering Gaussian mixtures composed of 8 functions. Under such circumstances, the number of classes to be detected is higher than the actual number of classes in the image. On the other hand, the proposed algorithm maintains the same configuration of Table 5. Therefore, it can detect and calculate up to ten classes ($K = 10$).

As a result, the techniques J + ABC, J + AIS, and J + DE tend to overestimate the image histogram. This effect can be seen in Figure 15(a), where the resulting Gaussian functions are concentrated within actual classes. Such a behavior is a consequence of the evaluation that is considered by the original objective function J, which privileges only the approximation between the Gaussian mixture and the image histogram. This effect can be graphically illustrated by Figure 16(a) that shows the pixel misclassification produced by the wrong segmentation of Figure 14(a). On the other hand, the proposed approach obtains a reduced Gaussian mixture model which allows the detection of each class from

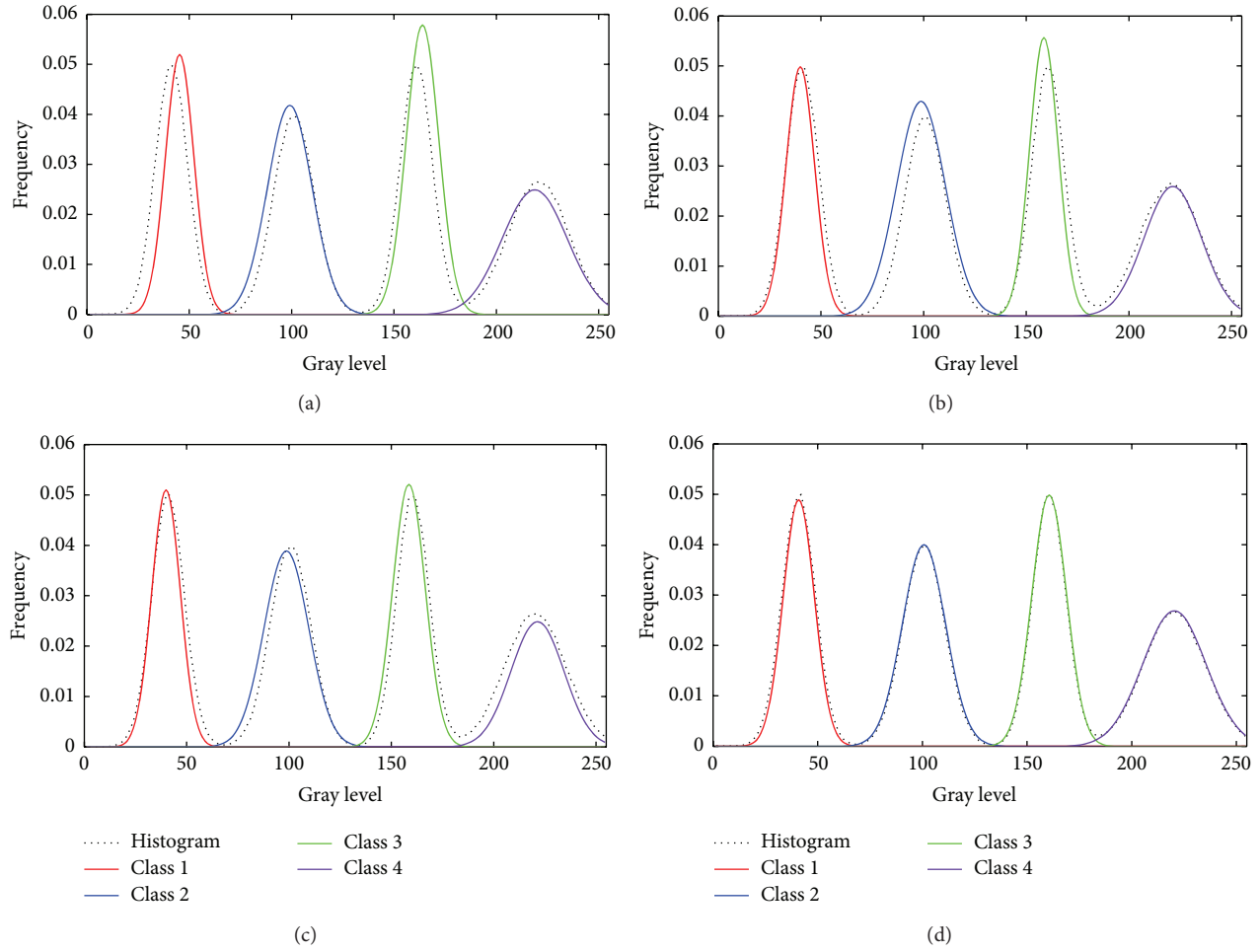


FIGURE 17: Approximation results in terms of accuracy. (a) J + ABC, (b) J + AIS, (c) J + DE, and (d) the proposed J^{new} + LS approach.

the actual histogram (see Figure 15(b)). As a consequence, the segmentation is significantly improved by eliminating the pixel misclassification, as it is shown by Figure 16(b).

It is evident from Figure 15 that the techniques, J + ABC, J + AIS, and J + DE, all need an a priori knowledge of the number of classes that are contained in the actual histogram in order to obtain a satisfactory result. On the other hand, the proposed algorithm is able to find a reduced Gaussian mixture whose classes coincide with the actual number of classes that are contained in the image histogram.

Accuracy. In this section, the comparison among the algorithms in terms of accuracy is reported. Most of the reported comparisons [19–26] are concerned about comparing the parameters of the resultant Gaussian mixtures by using real images. Under such circumstances, it is difficult to consider a clear reference in order to define a meaningful error. Therefore, the image defined in Figure 14 has been used in the experiments because its construction parameters are clearly defined in Table 7.

Since the parameter values from Table 7 act as ground truth, a simple averaged difference between them and the values that are computed by each algorithm could be used as comparison error. However, as each parameter maintains different active intervals, it is necessary to express the differences in terms of percentage. Therefore, if $\Delta\beta$ is the parametric difference and β the ground truth parameter, the percentage error $\Delta\beta\%$ can be defined as follows:

$$\Delta\beta\% = \frac{\Delta\beta}{\beta} \cdot 100\%. \tag{22}$$

In the segmentation problem, each Gaussian mixture represents a K-dimensional model where each dimension corresponds to a Gaussian function of the optimization problem to be solved. Since each Gaussian function possesses three parameters P_i, μ_i, and σ_i, the complete number of parameters is $3 \cdot K$ dimensions. Therefore, the final error E produced by the final Gaussian mixture is

$$E = \frac{1}{K \cdot 3} \sum_{v=1}^{K \cdot 3} \Delta\beta_v\%, \tag{23}$$

where $\beta_v \in (P_i, \mu_i, \sigma_i)$.

In order to compare accuracy, the algorithms, J + ABC, J + AIS, J + DE, and the proposed approach are all executed over the image shown by Figure 14(a). The experiment aims

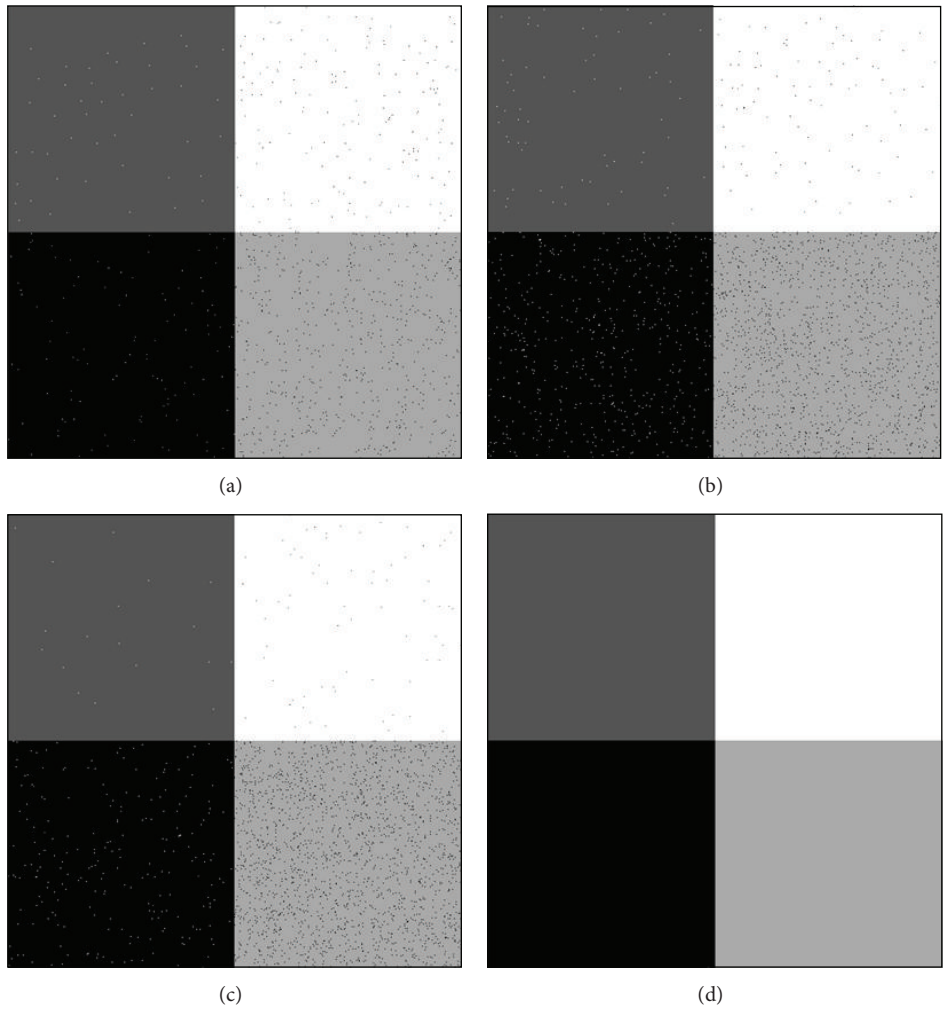


Figure 18: Segmentation results in terms of accuracy. (a) J + ABC, (b) J + AIS, (c) J + DE, and (d) the proposed J^{new} + LS approach.

to estimate the Gaussian mixture that better approximates the actual image histogram. Methods J + ABC, J + AIS, and J + DE consider Gaussian mixtures composed of 4 functions ($K = 4$). In case of the J^{new} + LS method, although the algorithm finds a reduced Gaussian mixture of four functions, it is initially set with ten functions ($K = 10$). Table 8 presents the final Gaussian mixture parameters and the final error E. The final Gaussian mixture parameters have been averaged over 30 independent executions in order to assure consistency. A close inspection of Table 8 reveals that the proposed method is able to achieve the smallest error (E) in comparison to the other algorithms. Figure 16 presents the histogram approximations that are produced by each algorithm whereas Figure 17 shows their correspondent segmented images. Both illustrations present the median case obtained throughout 30 runs. Figure 18 exhibits that J + ABC, J + AIS, and J + DE present different levels of misclassifications which are nearly absent in the proposed approach case.

Computational Cost. The experiment aims to measure the complexity and the computing time spent by the J + ABC,

Table 8: Results of the reduced Gaussian mixture in terms of accuracy.

Algorithm	Gaussian function	$\overline{P_i}$	$\overline{\mu_i}$	$\overline{\sigma_i}$	E
J + ABC	(1)	0.052	44.5	6.4	11.79%
	(2)	0.084	98.12	12.87	
	(3)	0.058	163.50	8.94	
	(4)	0.025	218.84	17.5	
J + AIS	(1)	0.072	31.01	6.14	22.01%
	(2)	0.054	88.52	12.21	
	(3)	0.039	149.21	9.14	
	(4)	0.034	248.41	13.84	
J + DE	(1)	0.041	35.74	7.86	13.57%
	(2)	0.036	90.57	11.97	
	(3)	0.059	148.47	9.01	
	(4)	0.020	201.34	13.02	
J^{new} + LS	(1)	0.049	40.12	7.5	3.98%
	(2)	0.041	102.04	10.4	
	(3)	0.052	168.66	8.3	
	(4)	0.025	110.92	15.2	

Figure 19: Images employed in the computational cost analysis.

the J + AIS, the J + DE, and the J^{new} + LS algorithm while calculating the parameters of the Gaussian mixture in benchmark images (see Figures 19(a)–19(d)). J + ABC, J + AIS, and J + DE consider Gaussian mixtures that are composed of 4 functions ($K = 4$). In case of the J^{new} + LS method, although the algorithm finds a reduced Gaussian mixture of four functions despite being initialized with ten functions ($K = 10$), Table 9 shows the averaged measurements after 30 experiments. It is evident that the J + ABC and J + DE are the slowest to converge (iterations) and the J + AIS shows the highest computational cost (time elapsed) because it requires operators which demand long times. On the other hand, the J^{new} + LS shows an acceptable trade-off between its convergence time and its computational cost. Therefore, although the implementation of J^{new} + LS in general requires more code than most of other evolution-based segmentators, such a fact is not reflected in the execution time. Finally, Figure 19 below shows the segmented images as they are

(a) (b) (c) (d)

Figure 20: Experimental set used in the evaluation of the segmentation results.

Table 9: Iterations and time requirements of the J + ABC, the J + AIS, the J + DE. and the J^{new} + LS algorithm as they are applied to segment benchmark images (see Figure 17).

Iterations Time elapsed	(a)	(b)	(c)	(d)
J + ABC	855	833	870	997
	2.72 s	2.70 s	2.73 s	3.1 s
J + AIS	725	704	754	812
	1.78 s	1.61 s	1.41 s	2.01 s
J + DE	657	627	694	742
	1.25 s	1.12 s	1.45 s	1.88 s
J^{new} + LS	314	298	307	402
	0.98 s	0.84 s	0.72 s	1.02 s

Table 10: Evaluation of the segmentation results in terms of the ROS index.

Number of classes Image	$N_R = 4$ (a)	$N_R = 3$ (b)	$N_R = 4$ (c)	$N_R = 4$ (d)
J + ABC	0.534	0.328	0.411	0.457
J + AIS	0.522	0.321	0.427	0.437
J + DE	0.512	0.312	0.408	0.418
J^{new} + LS	0.674	0.401	0.514	0.527

generated by each algorithm. It can be seen that the proposed approach generate more homogeneous regions whereas J + ABC, J + AIS, and J + DE present several artifacts that are produced by an incorrect pixel classification.

7.3. Performance Evaluation of the Segmentation Results. This section presents an objective evaluation of segmentation results that are produced by all algorithms in the comparisons. The ill-defined nature of the segmentation problem makes the evaluation of a candidate algorithm difficult [57]. Traditionally, the evaluation has been conducted by using some supervised criteria [58] which are based on the computation of a dissimilarity measure between a segmentation result and a ground truth image. Recently, the use of unsupervised measures has substituted supervised indexes for the objective evaluation of segmentation results [59]. They

Table 11: Unimodal test functions.

Test function	$\mathbf{S}$	f_{opt}
$f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$	0
$f_2(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^n$	0
$f_3(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^n$	0
$f_4(\mathbf{x}) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	$[-100, 100]^n$	0
$f_5(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right]$	$[-30, 30]^n$	0
$f_6(\mathbf{x}) = \sum_{i=1}^{n} \left(x_i + 0.5 \right)^2$	$[-100, 100]^n$	0
$f_7(\mathbf{x}) = \sum_{i=1}^{n} i x_i^4 + \text{rand}(0, 1)$	$[-1.28, 1.28]^n$	0

enable the quantification of the quality of a segmentation result without a priori knowledge (ground truth image).

Evaluation Criteria. In this paper, the unsupervised index ROS proposed by Chabrier et al. [60] has been used to objectively evaluate the performance of each candidate algorithm. This index evaluates the segmentation quality in terms of the homogeneity within segmented regions and the heterogeneity among the different regions. ROS can be computed as follows:

$$\text{ROS} = \frac{\overline{D} - \underline{D}}{2}, \tag{24}$$

where $\underline{D}$ quantifies the homogeneity within segmented regions. Similarly, $\overline{D}$ measures the disparity among the regions. A segmentation result S_1 is considered better than S_2, if $\text{ROS}_{S_1} > \text{ROS}_{S_2}$. The interregion homogeneity characterized by $\underline{D}$ is calculated considering the following formulation:

$$\underline{D} = \frac{1}{N_R} \sum_{c=1}^{N_R} \frac{R_c}{I} \cdot \frac{\sigma_c}{\left(\sum_{l=1}^{N_R} \sigma_l \right)}, \tag{25}$$

where N_R represents the number of partitions in which the image has been segmented. R_c symbolizes the number of

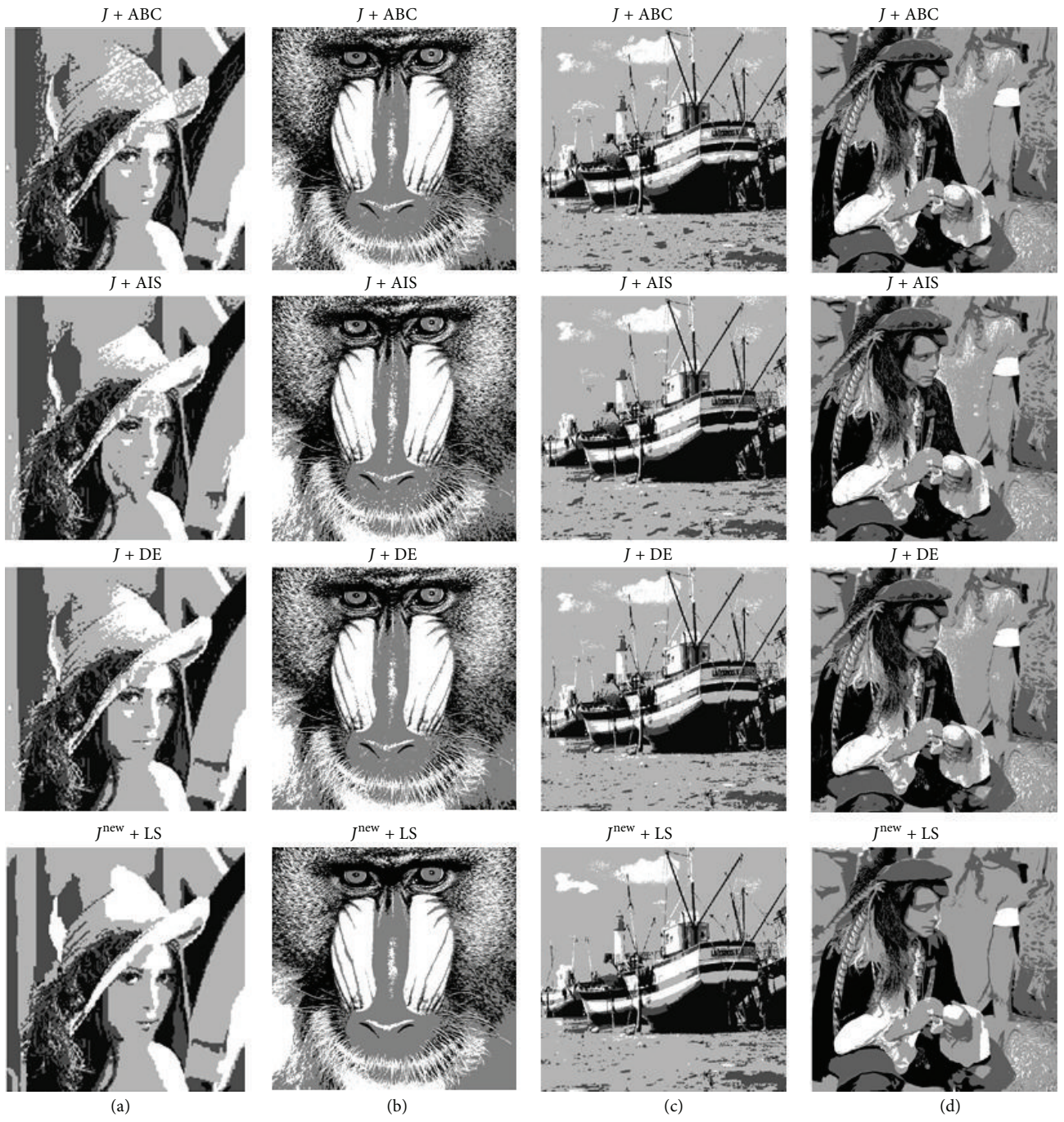


Figure 21: Segmentation results using in the evaluation.

pixels contained in the partition c whereas I considers the number of pixels that integrate the complete image. Similarly, σ_c represents the standard deviation from the partition c. On the other hand, disparity among the regions $\overline{D}$ is computed as follows:

$$\overline{D} = \frac{1}{N_R}\sum_{c=1}^{N_R}\frac{R_c}{I}\cdot\left[\frac{1}{(N_R-1)}\sum_{l=1}^{N_R}\frac{|\mu_c-\mu_l|}{255}\right], \quad (26)$$

where μ_c is the average gray level in the partition c.

Experimental Protocol. In the comparison of segmentation results, a set of four classical images has been chosen to integrate the experimental set (Figure 20). The segmentation methods used in the comparison are J + ABC [19], J + AIS [20], and J + DE [21].

From all segmentation methods used in the comparison, the proposed J^{new} + LS algorithm is the only one that has the capacity to automatically detect the number of segmentation partitions (classes). In order to conduct a fair comparison, all algorithms have been proved by using the same number

Table 12: Multimodal test functions.

Test function	S	f_{opt}
$f_8(\mathbf{x}) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$[-500, 500]^n$	$-418.98 * n$
$f_9(\mathbf{x}) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^n$	0
$f_{10}(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20$	$[-32, 32]^n$	0
$f_{11}(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^n$	0
$f_{12}(\mathbf{x}) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]^n$	0
$f_{13}(\mathbf{x}) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^n$	0

of partitions. Therefore, in the experiments, the J^{new} + LS segmentation algorithm is firstly applied to detect the best possible number of partitions N_R. Once we obtained the number of partitions N_R, the rest of the algorithms were configured to approximate the image histogram with this number of classes.

Figure 21 presents the segmentation results obtained by each algorithm considering the experimental set from Figure 20. On the other hand, Table 10 shows the evaluation of the segmentation results in terms of the ROS index. Such values represent the averaged measurements after 30 executions. From them, it can be seen that the proposed J^{new} + LS method obtains the best ROS indexes. Such values indicate that the proposed algorithm maintains the best balance between the homogeneity within segmented regions and the heterogeneity among the different regions. From Figure 21, it can be seen that the proposed approach generates more homogeneous regions whereas J + ABC, J + AIS, and J + DE present several artifacts that are produced by an incorrect pixel classification.

8. Conclusions

Despite the fact that several evolutionary methods have been successfully applied to image segmentation with interesting results, most of them have exhibited two important limitations: (1) they frequently obtain suboptimal results (misclassifications) as a consequence of an inappropriate balance between exploration and exploitation in their search strategies; (2) the number of classes is fixed and known in advance.

In this paper, a new swarm algorithm for the automatic image segmentation, called the Locust Search (LS), has been presented. The proposed method eliminates the typical flaws presented by previous evolutionary approaches by combining a novel evolutionary method with the definition of a new objective function that appropriately evaluates the segmentation quality with respect to the number of classes. In order to illustrate the proficiency and robustness of the proposed approach, several numerical experiments have been conducted. Such experiments have been divided into two parts. First, the proposed LS method has been compared to other well-known evolutionary techniques on a set of benchmark functions. In the second part, the performance of the proposed segmentation algorithm has been compared to other segmentation methods based on evolutionary principles. The results in both cases validate the efficiency of the proposed technique with regard to accuracy and robustness.

Several research directions will be considered for future work such as the inclusion of other indexes to evaluate similarity between a candidate solution and the image histogram, the consideration of spatial pixel characteristics in the objective function, the modification of the evolutionary LS operators to control the exploration-exploitation balance, and the conversion of the segmentation procedure into a multiobjective problem.

Appendix

List of Benchmark Functions

In Table 11, n is the dimension of function, f_{opt} is the minimum value of the function, and $\mathbf{S}$ is a subset of R^n. The optimum location ($\mathbf{x}_{\text{opt}}$) for functions in Table 11 is in $[0]^n$, except for f_5 with $\mathbf{x}_{\text{opt}}$ in $[1]^n$.

The optimum locations ($\mathbf{x}_{\text{opt}}$) for functions in Table 12 are in $[0]^n$, except for f_8 in $[420.96]^n$ and f_{12}-f_{13} in $[1]^n$.

Conflict of Interests

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

Acknowledgment

The present research has been supported under the Grant CONACYT CB 181053.

References

[1] H. Zhang, J. E. Fritts, and S. A. Goldman, "Image segmentation evaluation: a survey of unsupervised methods," *Computer Vision and Image Understanding*, vol. 110, no. 2, pp. 260–280, 2008.

[2] T. Uemura, G. Koutaki, and K. Uchimura, "Image segmentation based on edge detection using boundary code," *International Journal of Innovative Computing*, vol. 7, pp. 6073–6083, 2011.

[3] L. Wang, H. Wu, and C. Pan, "Region-based image segmentation with local signed difference energy," *Pattern Recognition Letters*, vol. 34, no. 6, pp. 637–645, 2013.

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

[5] R. Peng and P. K. Varshney, "On performance limits of image segmentation algorithms," *Computer Vision and Image Understanding*, vol. 132, pp. 24–38, 2015.

[6] M. A. Balafar, "Gaussian mixture model based segmentation methods for brain MRI images," *Artificial Intelligence Review*, vol. 41, no. 3, pp. 429–439, 2014.

[7] G. J. McLachlan and S. Rathnayake, "On the number of components in a Gaussian mixture model," *Data Mining and Knowledge Discovery*, vol. 4, no. 5, pp. 341–355, 2014.

[8] D. Oliva, V. Osuna-Enciso, E. Cuevas, G. Pajares, M. Pérez-Cisneros, and D. Zaldívar, "Improving segmentation velocity using an evolutionary method," *Expert Systems with Applications*, vol. 42, no. 14, pp. 5874–5886, 2015.

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

[10] S. Sarkar, S. Das, and S. Sinha Chaudhuri, "A multilevel color image thresholding scheme based on minimum cross entropy and differential evolution," *Pattern Recognition Letters*, vol. 54, pp. 27–35, 2015.

[11] A. K. Bhandari, A. Kumar, and G. K. Singh, "Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1573–1601, 2015.

[12] H. Permuter, J. Francos, and I. Jermyn, "A study of Gaussian mixture models of color and texture features for image classification and segmentation," *Pattern Recognition*, vol. 39, no. 4, pp. 695–706, 2006.

[13] A. P. Dempster, A. P. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[14] Z. Zhang, C. Chen, J. Sun, and K. L. Chan, "EM algorithms for Gaussian mixtures with split-and-merge operation," *Pattern Recognition*, vol. 36, no. 9, pp. 1973–1983, 2003.

[15] H. Park, S.-I. Amari, and K. Fukumizu, "Adaptive natural gradient learning algorithms for various stochastic models," *Neural Networks*, vol. 13, no. 7, pp. 755–764, 2000.

[16] H. Park and T. Ozeki, "Singularity and slow convergence of the em algorithm for gaussian mixtures," *Neural Processing Letters*, vol. 29, no. 1, pp. 45–59, 2009.

[17] L. Gupta and T. Sortrakul, "A Gaussian-mixture-based image segmentation algorithm," *Pattern Recognition*, vol. 31, no. 3, pp. 315–325, 1998.

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

[19] E. Cuevas, F. Sención, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "A multi-threshold segmentation approach based on artificial bee colony optimization," *Applied Intelligence*, vol. 37, no. 3, pp. 321–336, 2012.

[20] E. Cuevas, V. Osuna-Enciso, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "Multithreshold segmentation based on artificial immune systems," *Mathematical Problems in Engineering*, vol. 2012, Article ID 874761, 20 pages, 2012.

[21] E. Cuevas, D. Zaldivar, and M. Pérez-Cisneros, "A novel multi-threshold segmentation approach based on differential evolution optimization," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5265–5271, 2010.

[22] D. Oliva, E. Cuevas, G. Pajares, D. Zaldivar, and V. Osuna, "A multilevel thresholding algorithm using electromagnetism optimization," *Neurocomputing*, vol. 139, pp. 357–381, 2014.

[23] D. Oliva, E. Cuevas, G. Pajares, D. Zaldivar, and M. Perez-Cisneros, "Multilevel thresholding segmentation based on harmony search optimization," *Journal of Applied Mathematics*, vol. 2013, Article ID 575414, 24 pages, 2013.

[24] E. Cuevas, D. Zaldivar, and M. Pérez-Cisneros, "Seeking multithresholds for image segmentation with Learning Automata," *Machine Vision and Applications*, vol. 22, no. 5, pp. 805–818, 2011.

[25] K. C. Tan, S. C. Chiam, A. A. Mamun, and C. K. Goh, "Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization," *European Journal of Operational Research*, vol. 197, no. 2, pp. 701–713, 2009.

[26] G. Chen, C. P. Low, and Z. Yang, "Preserving and exploiting genetic diversity in evolutionary programming algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 661–673, 2009.

[27] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[28] R. Storn and K. Price, "Differential Evolution—a simple and efficient adaptive scheme for global optimisation over continuous spaces," Tech. Rep. TR-95–012, ICSI, Berkeley, Calif, USA, 1995.

[29] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, and Q. Tian, "Self-adaptive learning based particle swarm optimization," *Information Sciences*, vol. 181, no. 20, pp. 4515–4538, 2011.

[30] J. Tvrdík, "Adaptation in differential evolution: a numerical comparison," *Applied Soft Computing Journal*, vol. 9, no. 3, pp. 1149–1155, 2009.

[31] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J.-s. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013.

[32] W. Gong, Á. Fialho, Z. Cai, and H. Li, "Adaptive strategy selection in differential evolution for numerical optimization: an empirical study," *Information Sciences*, vol. 181, no. 24, pp. 5364–5386, 2011.

[33] D. M. Gordon, "The organization of work in social insect colonies," *Complexity*, vol. 8, no. 1, pp. 43–46, 2002.

[34] S. Kizaki and M. Katori, "Stochastic lattice model for locust outbreak," *Physica A: Statistical Mechanics and its Applications*, vol. 266, no. 1-4, pp. 339–342, 1999.

[35] S. M. Rogers, D. A. Cullen, M. L. Anstey et al., "Rapid behavioural gregarization in the desert locust, Schistocerca gregaria entails synchronous changes in both activity and attraction to conspecifics," *Journal of Insect Physiology*, vol. 65, pp. 9–26, 2014.

[36] C. M. Topaz, A. J. Bernoff, S. Logan, and W. Toolson, "A model for rolling swarms of locusts," *European Physical Journal: Special Topics*, vol. 157, no. 1, pp. 93–109, 2008.

[37] C. M. Topaz, M. R. D'Orsogna, L. Edelstein-Keshet, and A. J. Bernoff, "Locust dynamics: behavioral phase change and swarming," *PLoS Computational Biology*, vol. 8, no. 8, Article ID e1002642, 2012.

[38] G. Oster and E. Wilson, *Caste and Ecology in the Social Insects*, Princeton University Press, Princeton, NJ, USA, 1978.

[39] B. Hölldobler and E. O. Wilson, *Journey to the Ants: A Story of Scientific Exploration*, 1994.

[40] B. Hölldobler and E. O. Wilson, *The Ants*, Harvard University Press, Cambridge, Mass, USA, 1990.

[41] S. Tanaka and Y. Nishide, "Behavioral phase shift in nymphs of the desert locust, *Schistocerca gregaria*: special attention to attraction/avoidance behaviors and the role of serotonin," *Journal of Insect Physiology*, vol. 59, no. 1, pp. 101–112, 2013.

[42] E. Gaten, S. J. Huston, H. B. Dowse, and T. Matheson, "Solitary and gregarious locusts differ in circadian rhythmicity of a visual output neuron," *Journal of Biological Rhythms*, vol. 27, no. 3, pp. 196–205, 2012.

[43] I. Benaragama and J. R. Gray, "Responses of a pair of flying locusts to lateral looming visual stimuli," *Journal of Comparative Physiology A*, vol. 200, no. 8, pp. 723–738, 2014.

[44] M. G. Sergeev, "Distribution patterns of grasshoppers and their Kin in the boreal zone," *Psyche*, vol. 2011, Article ID 324130, 9 pages, 2011.

[45] S. O. Ely, P. G. N. Njagi, M. O. Bashir, S. El-Tom El-Amin, and A. Hassanali, "Diel behavioral activity patterns in adult solitarious desert locust, *Schistocerca gregaria* (Forskål)," *Psyche*, vol. 2011, Article ID 459315, 9 pages, 2011.

[46] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Beckington, UK, 2008.

[47] E. Cuevas, A. Echavarría, and M. A. Ramírez-Ortegón, "An optimization algorithm inspired by the States of Matter that improves the balance between exploration and exploitation," *Applied Intelligence*, vol. 40, no. 2, pp. 256–272, 2014.

[48] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky, "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems," *Journal of Global Optimization*, vol. 31, no. 4, pp. 635–672, 2005.

[49] R. Chelouah and P. Siarry, "Continuous genetic algorithm designed for the global optimization of multimodal functions," *Journal of Heuristics*, vol. 6, no. 2, pp. 191–213, 2000.

[50] F. Herrera, M. Lozano, and A. M. Sánchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study," *International Journal of Intelligent Systems*, vol. 18, no. 3, pp. 309–338, 2003.

[51] E. Cuevas, D. Zaldivar, M. Pérez-Cisneros, and M. Ramírez-Ortegón, "Circle detection using discrete differential evolution optimization," *Pattern Analysis & Applications*, vol. 14, no. 1, pp. 93–107, 2011.

[52] A. Sadollah, H. Eskandar, A. Bahreininejad, and J. H. Kim, "Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems," *Applied Soft Computing*, vol. 30, pp. 58–71, 2015.

[53] M. S. Kiran, H. Hakli, M. Gunduz, and H. Uguz, "Artificial bee colony algorithm with variable search strategy for continuous optimization," *Information Sciences*, vol. 300, pp. 140–157, 2015.

[54] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, pp. 3508–3531, 2011.

[55] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[56] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[57] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 929–944, 2007.

[58] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "A measure for objective evaluation of image segmentation algorithms," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, p. 34, San Diego, Calif, USA, June 2005.

[59] Y. J. Zhang, "A survey on evaluation methods for image segmentation," *Pattern Recognition*, vol. 29, no. 8, pp. 1335–1346, 1996.

[60] S. Chabrier, B. Emile, C. Rosenberger, and H. Laurent, "Unsupervised performance evaluation of image segmentation," *EURASIP Journal on Advances in Signal Processing*, vol. 2006, Article ID 96306, 12 pages, 2006.