


Research Article

Comparison of Recent Metaheuristic Algorithms for Shape Detection in Images

Erik Cuevas^{*,} , Angel Trujillo, Mario A. Navarro, Primitivo Diaz

Departamento de Electrónica, Universidad de Guadalajara, CUCEI, Av. Revolución 1500, C.P 44430, Guadalajara, Jal, Mexico

ARTICLE INFO

Article History

Received 19 Dec 2019

Accepted 27 Jul 2020

Keywords

Metaheuristics
Shape detection
Image processing
Machine learning

ABSTRACT

Shape recognition in images represents one of the complex and hard-solving problems in computer vision due to its nonlinear, stochastic and incomplete nature. Classical image processing techniques have been normally used to solve this problem. Alternatively, shape recognition has also been conducted through metaheuristic algorithms. They have demonstrated to have a competitive performance in terms of robustness and accuracy. However, all of these schemes use old metaheuristic algorithms as the basis to identify geometrical structures in images. Original metaheuristic approaches experiment several limitations such as premature convergence and low diversity. Through the introduction of new models and evolutionary operators, recent metaheuristic methods have addressed these difficulties providing in general better results. This paper presents a comparative analysis on the application of five recent metaheuristic schemes to the shape recognition problem such as the Grey Wolf Optimizer (GWO), Whale Optimizer Algorithm (WOA), Crow Search Algorithm (CSA), Gravitational Search Algorithm (GSA) and Cuckoo Search (CS). Since such approaches have been successful in several new applications, the objective is to determine their efficiency when they face a complex problem such as shape detection. Numerical simulations, performed on a set of experiments composed of images with different difficulty levels, demonstrates the capacities of each approach.

© 2020 The Authors. Published by Atlantis Press B.V.

This is an open access article distributed under the CC BY-NC 4.0 license (<http://creativecommons.org/licenses/by-nc/4.0/>).

1. INTRODUCTION

The problem of shape detection such as lines, circles, circular arcs and ellipses appears in several fields of pattern recognition with uses in many industrial applications from automatic object recognition to car navigation [1]. Shape detection is normally conducted through the Hough Transform (HT) [1,2]. A standard Hough-based scheme uses edge information produced by an edge detector algorithm to obtain candidate geometrical shapes. In the process, each edge pixel, which coincides with a hypothetical geometrical shape, votes in the parameter space with the information that corresponds to its parameters. Then, the most probable shape is obtained by averaging, filtering and histogramming the parameter space. HT is a robust algorithm; however, it maintains some important flaws. HT requires a large amount of space to store all considered parameters of the geometrical shape. It also needs a high computational complexity, which yields a low performance in terms of speed. Another critical problem is accuracy. In HT, the precision of the obtained parameters for the detected shape is reduced, especially in noise conditions [3].

Alternatively to the HT, the problem of shape detection has also been faced through metaheuristic approaches. In general terms, they have shown better results than those that use the HT concerning accuracy, speed and robustness [4]. A detector, based on a

metaheuristic scheme, utilizes a set of edge points as decision variables to determine hypothetical shapes (possible solutions). A matching function evaluates if a hypothetical shape is actually contained in the image. Therefore, conducted by the values of the matching function, the set of encoded points are operated through a particular metaheuristic approach so that the best solutions represent the original shapes inside the image. Different pioneer metaheuristic algorithms have been used to produce several interesting shape detectors such as Genetic algorithms (GAs) [5,6] and Particle Swarm Optimization (PSO) [7], Differential Evolution (DE) [8], Cloning Selection method (CSM) [9], Harmony Search (HS) [10], Artificial Bee Colony (ABC) [11] and Animal Behavior (AB) [12,13].

Recently, metaheuristic approaches have consolidated their popularity with exceptional growth and a large number of applications [14]. Such approaches represent difficult opposition to classical optimization techniques, which are prone to fail in several simple contexts. On the other hand, initial metaheuristic schemes maintain in their design several limitations such as premature convergence and inability to maintain population diversity [15]. Recent metaheuristic methods have addressed these difficulties providing in general better results [16]. Many of these novel metaheuristic approaches have also been introduced lately. In general, they propose new models and innovative evolutionary operators for producing an adequate exploration and exploitation of large search spaces considering a significant number of dimensions. This

*Corresponding author. Email: erik.cuevas@cucei.udg.mx

new generation of metaheuristic approaches is large, some examples include Crow Search Algorithm (CSA) [17], Cuckoo Search (CS) [18], Moth-flame Optimization Algorithm (MFOA) [19], Grey Wolf Optimizer (GWO) [20], Whale Optimization Algorithm (WOA) [21], Gravitational Search Algorithm (GSA) [22], Sine Cosine Algorithm (SCA) [23], Cricket Behavior-Based Evolutionary (CBE) [24], Dragonfly Algorithm (DF) [25], Hybrid Cognitive Optimization Algorithm (HCOA) [26], Interactive Search Algorithm (ISA) [27], Farmland Fertility Algorithm (FFA) [28], Binary Artificial Algae Algorithm (BAAA) [29], Electro-Search algorithm (ESA) [30], Island Bat Algorithm for Optimization (IBAO) [31], to name a few.

According to the literature, there is no single metaheuristic scheme that can solve every problem effectively. Metaheuristic schemes have been designed with special search models that allow them to solve competitively specific problems. Under such conditions, to evaluate the performance of a certain metaheuristic method appropriately, its relative efficiency should be assessed in the scenario in which it would be applied. Different comparisons among metaheuristic methods have been presented in several studies where it is analyzed their solutions over a representative set of benchmark functions with exact solutions and well-known behaviors. However, their conclusions are limited since they do not consider any application context. This paper presents a comparative analysis on the application of five recent and most popular metaheuristic schemes to the shape recognition problem such as the GWO, Whale Optimizer Algorithm (WOA), CSA, GSA and CS. These algorithms have been successful in several new domains. Under this condition, the objective is to determine their efficiency when they face a complex problem such as shape detection. Such approaches have been selected for our analysis, since they are considered the methods most currently in use according to the popular scientific databases, ScienceDirect (<http://www.sciencedirect.com>), SpringerLink (<http://www.springerlink.com>) and IEEE Xplore (<http://ieeexplore.ieee.org>), over the last five years. The problem of shape detection involves the identification of lines, circles, circular arcs and ellipses. In this work, in order to provide an interesting comparison among all the methods, the ellipse detection has been selected. It is considered the most complex recognition task with regard to lines and circular features. The comparison is divided into two parts. In the first part, the performance of all metaheuristic algorithms is evaluated with regard to the ellipse detection in natural images, while, in the second part, the robustness in the detection of each algorithm in the presence of complex scenarios is analyzed. In the study, several performance indexes have been considered to evaluate the accuracy of detection and computational overload. Additionally, test methods are achieved over the experimental data in order to validate the results.

The remainder of this paper is organized as follows: Section 2 formulates the detection problem such as an optimization task; in Section 3 the main characteristic of each compared algorithms are presented; in Section 4, a comparative perspective of the involved metaheuristic schemes is analyzed; in Section 5, the numerical results are presented. Finally, in Section 6, conclusions are given.

2. PROBLEM FORMULATION

To identify ellipse in images, they must be firstly preprocessed by an edge detection algorithm that produces a single-pixel-contour edge

map (such as the Canny method). Next, the location (x_i, y_i) of each edge element e_i is collected within an edge vector $E = \{e_1, e_2, \dots, e_{N_p}\}$; where N_p correspond to the total number of detected edge pixels in the image. For the sake of clarity, Table 1 presents a description of each symbol used in this section.

In this scheme, a candidate solution C (ellipse candidate) is coded by a combination of five different edge points contained on its perimeter. Therefore, candidate ellipses are produced by randomly selecting edge points within the edge array E . Figure 1 shows an illustration from the codification process.

With this method, according to Figure 1, a candidate shape is represented as the ellipse which passes through the five points e_p, e_q, e_r, e_s and e_t ($C = \{e_p, e_q, e_r, e_s, e_t\}$), which defines its parameter vector. An ellipse is modeled by the following scheme:

$$\hat{a}x_i^2 + 2\hat{h}x_iy_i + \hat{b}y_i^2 + 2\hat{g}x_i + 2\hat{f}y_i + \hat{c} = 0 \quad (1)$$

Consequently, by replacing the coordinates of each parameter point e_k ($k \in p, q, r, s, t$) from C into Eq. (1), we obtain a set of five simultaneous equations. Therefore, after determining the elements $\hat{a}, \hat{b}, \hat{c}, \hat{f}$ and \hat{g} , and dividing by the constant \hat{c} , the following model is produced:

$$ax_i^2 + 2hx_iy_i + by_i^2 + 2gx_i + 2fy_i + 1 = 0 \quad (2)$$

From this model, the main geometrical characteristics of the ellipse such as the center (x_0, y_0) , maximal radius r_{\max} , minimal radius r_{\min} and the orientation θ can be computed as follows:

$$x_0 = \frac{hf - bg}{D}, y_0 = \frac{gh - af}{D} \quad (3)$$

$$r_{\max} = \sqrt{\frac{-2\Delta}{D(a + b - R)}}, r_{\min} = \sqrt{\frac{-2\Delta}{D(a + b + R)}} \quad (4)$$

$$\theta = \frac{1}{2} \arctan\left(\frac{2h}{a - b}\right), \quad (5)$$

where

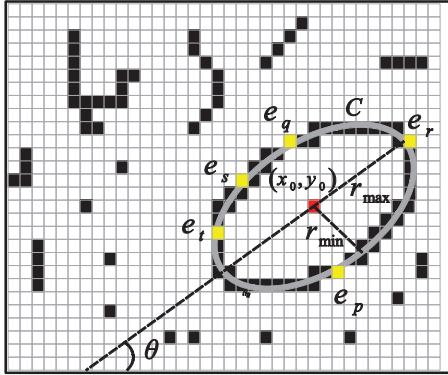
$$R = \sqrt{(a - b)^2 + 4h^2}, D = ab - h^2, \Delta = \det \begin{vmatrix} a & h & g \\ h & b & f \\ g & f & 1 \end{vmatrix} \quad (6)$$

To estimate the matching error $J(C)$ of each candidate ellipse C , the set P of its hypothetical coordinate points are firstly determined. This set P includes all the N_e pixel points $\{p_1, p_2, \dots, p_{N_e}\}$ that the ellipse would contain in its perimeter if it were present in the image. Then, it is validated, if such points really exist in the edge image.

The set P is produced by using the Midpoint Ellipse Algorithm (MEA) [32] which is a plotting scheme that finds the required points $\{p_1, p_2, \dots, p_{N_e}\}$ to draw an ellipse digitally. Each pixel coordinate $p_w = (x_w, y_w)$ (where $w \in 1, \dots, N_e$) on the perimeter of the ellipse C with parameters a, h, b, g , and f must fulfill the following relation $f_C(p_w) = r_{\max}^2x_w^2 + r_{\min}^2y_w^2 - r_{\max}^2r_{\min}^2$. Under this condition, the MEA method tests the position of every pixel point to

Table 1 | Description of each symbol used in this section.

(x_i, y_i)	Pixel position of the edge element e_i
$E = \{e_1, e_2, \dots, e_{N_p}\}$	Edge vector that contains all the edges presented in the image
N_p	Total number of detected edge pixels in the image
$C = \{e_p, e_q, e_r, e_s, e_t\}$	Ellipse candidate coded as the combination of five different edge points where $e_p, e_q, e_r, e_s, e_t \in E$.
$\hat{a}, \hat{b}, \hat{c}, \hat{f}, \hat{g}$ and \hat{h}	Coefficients that define the ellipse characteristics
a, b, f, g and h	Normalized coefficients of the ellipse so that $a = \hat{a}/\hat{c}$, $b = \hat{b}/\hat{c}$, $f = \hat{f}/\hat{c}$, $g = \hat{g}/\hat{c}$ and $h = \hat{h}/\hat{c}$
(x_0, y_0)	Center of the ellipse
r_{\max}, r_{\min}	Maximal radius and minimal radii, respectively. They correspond to the maximum axis and minimum axis
θ	Ellipse orientation
R and D	Temporary variables, where $R = \sqrt{(a-b)^2 + 4h^2}$ and $D = ab - h^2$
Δ	The determinant of the ellipse parameters so that
	$\det \begin{vmatrix} a & h & g \\ h & b & f \\ g & f & 1 \end{vmatrix}$
$P = \{p_1, p_2, \dots, p_{N_e}\}$	A hypothetical set of points that define the ellipse candidate C
N_e	Number of hypothetical points that involve the ellipse candidate C
$p_w = (x_w, y_w)$	Pixel position of each hypothetical point $w \in 1, \dots, N_e$.
$J(C)$	Matching error of the candidate ellipse C
$f_c(p_w)$	Relation to decide if p_w is located inside or outside the ellipse perimeter
$Q(p_w)$	Function that checks the existence of the ellipse perimeter point p_w in the edge vector E

**Figure 1** | Candidate ellipse shape produced from the combination of points .

decide if it is located inside or outside the ellipse perimeter. Therefore, if the pixel p_w is within the ellipse perimeter, the ellipse function $f_c(p_w)$ produces a negative value. On the other hand, if the pixel p_w is outside the ellipse boundary, the ellipse function $f_c(p_w)$ delivers a positive value. With this information, the MEA algorithm finds the pixel coordinates $\{p_1, p_2, \dots, p_{N_e}\}$ that yields values from $f_c(\cdot)$ close to zero. Such points correspond to the pixel elements that integrate the ellipse perimeter of C .

Once determined the perimeter points P of C , the matching error $J(C)$ is computed. This value can be considered as the objective

function to be minimized. $J(C)$ corresponds to the existent similarity between the perimeter points P of the ellipse candidate C and the pixels that actually are present in the edge image. Therefore, $J(C)$ is modeled as follows:

$$J(C) = 1 - \frac{\sum_{u=1}^{N_e} Q(p_w)}{N_e}, \quad (7)$$

where $Q(p_w)$ is a function that checks the existence of the ellipse perimeter point p_w in the edge vector E . This process can be formulated in this way:

$$Q(p_w) = \begin{cases} 1 & \text{if } p_w \in E \\ 0 & \text{otherwise} \end{cases}, \quad (8)$$

A value of $J(C)$ close to zero indicates a better approximation of the ellipse C with the actual ellipse present in the image. Figure 2 presents the process to evaluate the matching error of a candidate ellipse C in a typical image example.

Figure 2(a) illustrates the original edge map E from the image. Considering that the candidate ellipse C groups the edge pixels e_p, e_q, e_r, e_s and e_t from the edge vector E , Figure 2(b) shows the perimeter coordinates $P = \{p_1, p_2, \dots, p_{N_e}\}$ produced by the MEA method. It represents an ellipse of 52-pixel points ($N_e = 52$). Finally, in Figure 2(c), the perimeter points P are overlapped with the actual edge map E , point by point, to detect the coincidences between P and E . From Figure 2(c), only 29 coordinate points are

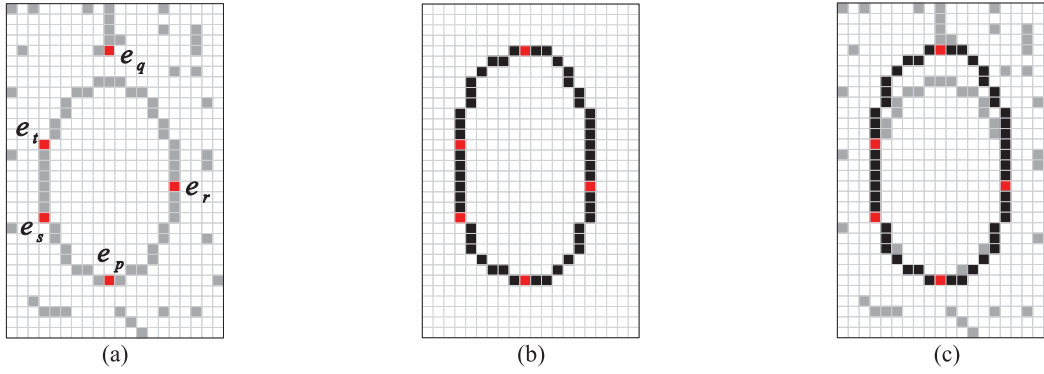


Figure 2 | Computation of the matching error .

present in both vectors P and E $\left(\sum_{u=1}^{N_e} Q(p_w) = 29\right)$. With these values, $J(C)$ is 0.4423.

2.1. Multiple Ellipse Detection

To detect multiple instances, the shape detector stores the best ellipse candidate in each iteration k within a memory $\mathbf{M}(k)$. Therefore, once the metaheuristic method has finished, an exploration of $\mathbf{M}(k)$ is conducted. The idea is to evaluate which solutions in $\mathbf{M}(k)$ corresponds to an actual ellipse in the edge map. Considering that several elements from $\mathbf{M}(k)$ could describe the same ellipse (i.e., ellipses with small variations), a distinctiveness factor D_{S_1, S_2} is necessary to calculate the dissimilarity between two given shapes S_1 and S_2 . D_{S_1, S_2} is modeled through the following expression:

$$D_{S_1, S_2} = |x_0^{S_1} - x_0^{S_2}| + |y_0^{S_1} - y_0^{S_2}| + |r_{\max}^{S_1} - r_{\max}^{S_2}| + |r_{\min}^{S_1} - r_{\min}^{S_2}| + |\theta_{S_1} - \theta_{S_2}| \quad (9)$$

Once computed D_{S_1, S_2} , a threshold value Th is also defined to determine whether S_1 and S_2 can be considered as similar or not. Th is formulated as follows:

$$Th = \frac{(|r_{\max}^l - r_{\max}^r| + |r_{\min}^l - r_{\min}^r|)}{4}, \quad (10)$$

where $[r_{\max}^l, r_{\max}^r]$ and $[r_{\min}^l, r_{\min}^r]$ corresponds to the feasible ranges for the maximal radius r_{\max} and the minimal radius r_{\min} , respectively. Under such conditions, the ellipses S_1 and S_2 represent the identical ellipse, if $D_{S_1, S_2} < Th$, otherwise, they are different. Therefore, each solution from $\mathbf{M}(k)$ is compared to each other. The idea is to find the ellipses that correspond to the same ellipse in order to select the best one in terms of the objective function $J(\cdot)$.

3. OPTIMIZATION TECHNIQUES

Optimization methods have been devised to find the global solution G_s of a nonlinear formulation with box constraints under the following scheme [33]:

$$G_s = \arg \min_{\mathbf{x}=(x_1, \dots, x_n) \in X} J(\mathbf{x}) \quad (11)$$

where $J(\cdot)$ represents a nonlinear objective function while $\mathbf{X} = \{\mathbf{x} \in \mathbb{R}^n | l_i \leq x_i \leq u_i, i = 1, \dots, n\}$ corresponds to constrained feasible search space, limited by the lower (l_i) and upper (u_i) limits.

To find the best solution for Eq. (11), from a metaheuristic viewpoint [33], a group of $\mathbf{P}(k)$ $\left(\{\mathbf{p}_1(k), \mathbf{p}_2(k), \dots, \mathbf{p}_{N_i}(k)\}\right)$ of N_i candidate solutions (particles) are evolved from an initial generation ($k = 0$) to a maximum number of iterations ($k = Maxgen$). As an initial step, a metaheuristic approach produces a collection of N_i candidate solutions whose values are selected according to a uniform distribution between the prespecified lower (l_i) and upper (u_i) bounds. In every iteration k , according to each metaheuristic algorithm, a set of evolutionary operators is used to produce a new population $\mathbf{P}(k + 1)$ from the original $\mathbf{P}(k)$. Each candidate solution $\mathbf{p}_i(k)$ in the population $\mathbf{P}(k)$ represents an n -dimensional vector $\{p_{i,1}(k), p_{i,2}(k), \dots, p_{i,n}(k)\}$ whose elements correspond to the decision variables in the optimization problem. The capacity of each candidate solution $\mathbf{p}_i(k)$ to solve successfully an optimization formulation is evaluated in terms of the objective function $J(\mathbf{p}_i(k))$ whose final value represents its quality. During the evolution process, the best candidate solution \mathbf{b} seen so far is maintained, because it corresponds to the best solution currently available.

This section presents the main characteristics of the five metaheuristic optimization approaches used in the comparison the GWO, WOA, CSA, GSA and CS.

3.1. GWO Algorithm

The GWO [20] is a metaheuristic algorithm that models its search strategy considering the social hierarchy and hunting process conducted by wolves.

3.1.1. Hierarchy

To implement a mathematical model of the social hierarchy in the algorithm, the best candidate solution \mathbf{b} is symbolized as \mathbf{p}_α solution. The adjacent two best solutions represent the \mathbf{p}_β and the \mathbf{p}_δ elements ($\alpha, \beta, \delta \in \mathbf{P}(k)$). The remainder of the elements are considered ω wolves ($\omega \in \mathbf{P}(k), \omega \neq \alpha, \beta, \delta$). Therefore, during the evolution process, the search strategy is conducted by the candidate solutions \mathbf{p}_α , \mathbf{p}_β , and \mathbf{p}_δ while the rest of the $(\mathbf{p})_\omega$ wolves follow them.

3.1.2. Covering the prey

In Grey wolf packs, wolves try to encircle prey during their hunting behavior. This process is modeled as follows:

$$\mathbf{d} = |C\mathbf{p}_p(k) - \mathbf{p}_\omega(k)| \quad (12)$$

$$\mathbf{p}_\omega(k+1) = \mathbf{p}_p(k) - A\mathbf{d}, \quad (13)$$

where k represents the current generation, A and C symbolize two fixed coefficients. \mathbf{p}_p and \mathbf{p}_ω represent the locations of the prey and a standard wolf agent, respectively. In Eqs. (10) and (11), the values of A and C are defined as:

$$A = 2ar_1 - 1 \quad (14)$$

$$C = 2r_2, \quad (15)$$

where a is a parameter which is decreased linearly from 2 to 0 during the optimization process. r_1 and r_2 represent two random numbers within the interval $[0, 1]$.

3.1.3. Hunting

In the hunting behavior, the search strategy is conducted mainly by the Alfa agent \mathbf{p}_α and occasionally by the agents \mathbf{p}_β and the \mathbf{p}_δ . With this behavior, it is considered that these agents maintain better information about the position of the optimum solution. Under this condition, the other agents \mathbf{p}_ω modify their locations in terms of the positions of \mathbf{p}_α , \mathbf{p}_β and the \mathbf{p}_δ . This process can be modeled as follows:

$$\mathbf{d}_\alpha = |C_1\mathbf{p}_\alpha(k) - \mathbf{p}_\omega(k)|, \quad (16)$$

$$\mathbf{d}_\beta = |C_2\mathbf{p}_\beta(k) - \mathbf{p}_\omega(k)|,$$

$$\mathbf{d}_\delta = |C_3\mathbf{p}_\delta(k) - \mathbf{p}_\omega(k)|$$

$$\mathbf{L}_1 = \mathbf{p}_\alpha(k) - A_1\mathbf{d}_\alpha, \quad (17)$$

$$\mathbf{L}_2 = \mathbf{p}_\beta(k) - A_2\mathbf{d}_\beta,$$

$$\mathbf{L}_3 = \mathbf{p}_\delta(k) - A_3\mathbf{d}_\delta,$$

where A_1, A_2, A_3, C_1, C_2 and C_3 are fixed coefficients computed by using Eqs. (12) and (13). Once computed these elements, the position in the next iteration ($k+1$) of each candidate solution \mathbf{p}_ω is adapted as follows:

$$\mathbf{p}_\omega(k+1) = (\mathbf{L}_1 + \mathbf{L}_2 + \mathbf{L}_3)/3, \quad (18)$$

3.1.4. Exploitation and exploitation

Both processes are implemented in the search strategy through the value of a in Eq. (12). Its value decreases linearly from 2 to 0 during the optimization process. As a diminishes its value, the variation interval of A also presents a reduction. Therefore, when $|A| > 1$ the algorithm explores the search space, while $|A| < 1$ the strategy concentrates their candidate solutions \mathbf{x}_ω around the best position.

3.2. Whale Optimizer Algorithm

The WOA [21] is an interesting metaheuristic method based on the particular spiral bubble-net hunting behavior found in humpback whales.

3.2.1. Encircling a prey

Under the WOA scheme, it is considered that the best candidate solution \mathbf{b} seen so far is the objective agent \mathbf{p}_b . Therefore, the other humpback whales \mathbf{p}_G ($\mathbf{p}_G \in \mathbf{P}(k)$) will be attracted by the objective solution. This process can be modeled as follows:

$$\mathbf{s} = |C\mathbf{p}_b(k) - \mathbf{p}_G(k)| \quad (19)$$

$$\mathbf{p}_G(k+1) = \mathbf{p}_b(k) - A\mathbf{s}, \quad (20)$$

where k represents the current generation, A and C symbolize two fixed coefficients, which are computed by using Eqs. (12) and (13). \mathbf{p}_b and \mathbf{p}_G represent the locations of the target and a standard whale agent, respectively.

3.2.2. Bubble-net attacking method

In this step, humpback whales use two procedures: shrinking (I) encircling and (II) spiraling. Both processes consider the modification of agent positions around the target element. In the approach, it is assumed that both procedures maintain an identical probability p_r to be executed. These strategies can be defined as follows [R11]:

$$\mathbf{p}_G(k+1) = \begin{cases} \mathbf{p}_b(k) - A\mathbf{s} & \text{(I) if } p_r < 0.5 \\ \mathbf{s} \cdot \exp(bl) \cdot \cos(2\pi l) + \mathbf{p}_b(k) & \text{(II) otherwise} \end{cases}, \quad (21)$$

where b is a fixed coefficient that defines the logarithmic shape of the distribution. l and p_r represent two random number within the interval $[0, 1]$.

3.3. Crow Search Algorithm

The CSA [17] is a metaheuristic approach inspired by the complex process presented in crows when they search for food resources. In CSA, each search agent \mathbf{p}_c ($\mathbf{p}_c \in \mathbf{P}(k)$) emulates the position of a certain crow in the group. Associated to each search agent \mathbf{p}_c , there exists a memory \mathbf{m}_c that stores the best-obtained solution of the agent c during the optimization process.

Under the CSA scheme, the location of each \mathbf{p}_c is adjusted considering two actions: (I) Pursuit and (II) evasion.

(I) Pursuit. In this process, an agent \mathbf{p}_j follows an agent \mathbf{p}_i with the objective of finding its hidden food.

(II) Evasion. The agent \mathbf{p}_i realizes the proximity of the agent \mathbf{p}_j . As a consequence, the agent \mathbf{p}_i deliberately select a random trajectory as a strategy to protect its food. This process is emulated in CSA through the assignation of a random position.

Each agent \mathbf{p}_i selects a behavior of pursuit or evasion based on an awareness probability (AP). Under this condition, a random number r_i is calculated considering a uniform distribution within the

interval $[0, 1]$. Therefore, when r_i is greater or equal than AP , the pursuit action (I) is chosen; otherwise, the evasion behavior (II) is selected. This operator can be modeled as follows:

$$\mathbf{p}_i(k+1) = \begin{cases} \mathbf{p}_i(k) + r_i \cdot fl_c (\mathbf{m}_j(k) - \mathbf{p}_c(k)) & \text{(I) if } r_i > AP \\ \text{random position} & \text{(II) otherwise} \end{cases}, \quad (22)$$

where fl_c represents the flight length which corresponds to the magnitude of the adjustment from the agent \mathbf{p}_c to the best location \mathbf{m}_j of the agent \mathbf{p}_i . Once computed the new positions for each agent \mathbf{p}_c , its memory \mathbf{m}_c is also updated ($c \in \{1, \dots, N_i\}$).

3.4. Gravitational Search Algorithm

The GSA [22] is a metaheuristic approach inspired by the laws of gravitation. In GSA, the first step is to calculate the total gravitational force. It represents the influence that exerts the set of the best K agents over a particular element, where K can also represent the total number of elements in the population N_i . For each solution $\mathbf{p}_i(k)$ the gravitational force \mathbf{F}_i can be obtained as follows:

$$\mathbf{F}_i(k) = \sum_{j \in K, j \neq i} r_2 G(k) \frac{M_j(k) M_i(k)}{R_{i,j}} (\mathbf{p}_j(k) - \mathbf{p}_i(k)), \quad (23)$$

$$G(k) = 100 \exp \left(-20 \frac{k}{Maxgen} \right), \quad (24)$$

where $i \in \{1, \dots, N_i\}$; k refers to the current iteration; r_2 corresponds to a uniform random number within $[0, 1]$; $R_{i,j}$ represents the Euclidian distance between the agents $\mathbf{p}_i(k)$ and $\mathbf{p}_j(k)$. $M_j(k)$ and $M_i(k)$ are computed by using the following model:

$$M_i(k) = \frac{q_i(k)}{\sum_{v=1}^{N_i} q_v(k)} \quad (25)$$

$$q_i(k) = \frac{J(\mathbf{p}_i(k)) - J(\mathbf{a})}{J(\mathbf{b}) - J(\mathbf{a})}, \quad (26)$$

where \mathbf{b} and \mathbf{a} represent the best and the worst elements in the current population $\mathbf{P}(k)$. Once calculated the values of \mathbf{F}_i , the acceleration of each agent $\mathbf{u}_i(k)$ is computed. It is obtained by the following equation:

$$\mathbf{u}_i(k) = \frac{\mathbf{F}_i(k)}{M_i(k)} \quad (27)$$

Finally, the position of each agent is updated as follows:

$$\mathbf{p}_i(k+1) = \mathbf{p}_i(k) + \mathbf{v}_i(k+1) \quad (28)$$

$$\mathbf{v}_i(k+1) = r_3 \mathbf{v}_i(k) + \mathbf{u}_i(k),$$

where r_3 corresponds to a uniform random number within $[0, 1]$ while the initial value of $\mathbf{v}_i(0)$ is set to 0.

3.5. CS Method

The CS [18] method is a metaheuristic algorithm for global optimization inspired by the behavior of cuckoo birds. Under the CS scheme, Levy flight random walk and biased random walk are employed as the basis for its search strategy. In a Levy flight random walk, a displacement is produced by sampling a Levy probability distribution which can be modeled as follows:

$$Levy(\lambda) = \frac{u}{|v|^{1/\beta}} \quad (29)$$

where β represents a distribution parameter that defines the shape of the probability distribution. It corresponds to a coefficient inside the range $[1, 2]$. On the other hand, u and v obey a normal distribution behavior. Accordingly, they can be defined as

$$u \sim N(0, \sigma_u^2), \quad v \sim N(0, 1) \quad (30)$$

$$\sigma_u^2 = \left[\frac{\Gamma(1+\beta) \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \beta 2^{(\beta-1)/2}} \right]^{1/\beta},$$

where $\Gamma(\cdot)$ represents the Gamma function. Therefore, the new position of a certain agent $\mathbf{p}_i(k+1)$ by using the Levy flight, random walk can be characterized as follows:

$$\mathbf{p}_i(k+1) = \mathbf{p}_i(k) + step \oplus Levy(\lambda) \quad (31)$$

$$step = \alpha \oplus (\mathbf{p}_i(k) - \mathbf{b}) \quad (32)$$

where α symbolizes the step size while \oplus corresponds to the pairwise multiplication. Finally, the operation of the biased random walk can be modeled with the following formulation:

$$\mathbf{p}_i(k+1) = \begin{cases} \mathbf{p}_i(k) + r_4 (\mathbf{p}_m(k) - \mathbf{p}_n(k)) & r_5 > p_a \\ \mathbf{p}_i(k) & \text{otherwise} \end{cases} \quad (33)$$

where r_4 and r_5 correspond to uniform random numbers within $[0, 1]$ while p_a represents the discovery probability. $\mathbf{p}_m(k)$ and $\mathbf{p}_n(k)$ symbolize two randomly selected agents from the population $\mathbf{P}(k)$.

4. COMPARATIVE PERSPECTIVE OF THE FIVE METAHEURISTIC METHODS

The five metaheuristic methods used in this study: GWO, WOA, CSA, GSA and CS are based on different search mechanisms. In this section, a comparative perspective of these methods is analyzed. The discussion considers the observable properties of each metaheuristic algorithms and the way in which such characteristics present an effect in the performance of these schemes.

It is widely accepted in the metaheuristic community that a proper balance between exploration and exploitation determines the potential of a metaheuristic approach [34]. Exploration refers to the process of searching for new points of the solution space. On the other hand, exploitation represents the operation of polishing those previously visited solutions to improve their relative quality by using a local search mechanism. The use of the only exploration diminishes the accuracy of the solutions but raises the ability of the algorithm to obtain new promising solutions [35]. On the contrary,

the application of exclusive exploitation allows increasing the precision of existing solutions. However, unfavorably provokes that the optimization process converges prematurely to locally optimal solutions [36]. The rate between exploration and exploitation experimented by a metaheuristic method depends on the mechanism of its search strategy.

In the analysis of the five metaheuristic methods, it has been considered the study of their search strategy and implementations. The search strategy has been divided into two different parts: selection approach or attraction scheme and independence on the evolution process.

The selection approach refers to the way in which each metaheuristic method collects solutions among the entire population to produce new ones. Such selected solutions are normally used as main elements to produce attraction operators in order to modify the positions of the current population. Basically, there exist two different schemes [34]: greedy, random and complete. Greedy selection considers the use of the best element or elements in terms of fitness values. Greedy selection ensures that most fitting individuals remain intact during the evolution process. Under such conditions, this scheme promotes the exploitation of the search space, increasing the fast convergence toward promising solutions. The algorithms GWO and WOA are considered within this category. The WOA uses the best individual (only one element) as an important element in its search strategy. In the case of the GWO method, it is remarkable that its selection mechanism uses the three best elements of the population. This fact allows not only favor exploitation but also increase diversity through the use of distinct solution types. On the other hand, random selection considers the use of an individual randomly selected within the complete population. The use of random elements promotes the exploration of the search space. However, adversely, it negatively affects to obtain consistent results or good accuracy. The algorithms CSA and CS consider the use of the random solution as part of their search strategy. Finally, the complete selection considers the use of all elements in the population to produce a new solution. This strategy uses, in general, the averaged position of all solutions to generate search direction patterns. This mechanism produces very low search schemes where it is necessary for an additional number of iterations to obtain the best optimization results. The GSA method uses the complete selection mechanism as in its search paradigm.

Regarding the independence of the evolution process [37], two different processes are distinguished: dependent and independent. Dependent processes represent strategies in which the number of iterations is considered to modify the search strategy. The idea under these schemes is to adapt the search strategy so that in the beginning, the exploration of the search space would be promoted while in the ending, the exploitation of the found solutions would be favored [38]. Representative schemes of this category are the GSA and CSA. Dependent methods present several difficulties since the results of the search process are not considering in the strategy; it is more important the number of iterations. Independent algorithms do not consider the stage of the evolution process to conduct their own search strategy. These algorithms use the knowledge produced by the obtained results to operate their search strategies [39]. In general terms, independent methods deliver better results than dependent methods. The algorithms GWO, WOA and CS are considered independent methods.

Table 2 | Comparative characteristics analyzed for each from the five algorithms.

Algorithm	Search Strategy		Implementation
	Selection or Attraction Scheme	Independence on the Evolution Process	
GWO	Greedy	Independent	Low
WOA	Greedy	Independent	Low
CSA	Random	Dependent	Medium
CS	Random	Independent	High
GSA	Complete	Dependent	High

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search.

In terms of implementation, it is compared to the difficulty that represents the use and operation of each from the five metaheuristic methods. The algorithms are classified according to their implementation issues [R1] in low, medium and high. Such categories consider two factors: the number of configurable parameters and the extension of code necessary for its programming. The problem of parameter tuning depends on the number of configurable parameters [40]. It is important to remark that an appropriate configuration produces a correct balance between exploration and exploitation and thus, competitive results. Currently, there is no standard methodology that allows the proper setting of a metaheuristic scheme. In fact, researchers and practitioners configure the parameters of metaheuristics algorithms by hand, guided only by the process of trial and error. Under such circumstances, the GWO and WOA are the metaheuristic schemes that do not consider any parameter to calibrate. The CS and CSA use four parameters, while the GSA employs three elements. Among the five schemes presented in this study, some show a high level of simplicity so that they can be promptly translated onto computational code with relative easiness [41]. In contrast, others are relatively complex due to the behaviors and rules considered in the model. According to these levels, the extension of necessary code to implement a certain metaheuristic method can be used as a criterion to classified each scheme in simple and complex. Therefore, the GWO, WOA, CSA present a simple level of programming while CS and GSA maintain a complex programming model. According to their implementation, the algorithms considered in this analysis can be classified as follows: GWO and WOA as low, CSA as medium and CS and GSA as high. Table 2 summarizes all the comparative characteristics analyzed for each of the five algorithms.

5. EXPERIMENTAL RESULTS

An experimental analysis has been conducted with the objective of comparing the performance of all detectors. The experiments consider the capacity of each detector to recognize accurately ellipsoid shapes.

5.1. Performance Indexes

In the study, two performance indexes have been considered in the comparisons: the multiple error (*ME*) and the computational time (*CT*). The first index evaluates the accuracy of the detection, whereas the second one considers the computational overload.

Images do not typically include perfect ellipses. Therefore, to examine the precision in the detection of a single ellipse, it is compared with a ground-truth ellipse that is obtained visually from the original image. In consequence, the parameters x_0^{true} , y_0^{true} , r_{max}^{true} , r_{min}^{true} and θ_{true} of the ground-truth ellipse are determined in terms of the best-fitted shape that a human observer is able to recognize by using a drawing program such as MSPaint, CorelDraw, and so on. Hence, if the parameters x_0^D , y_0^D , r_{max}^D , r_{min}^D and θ_D correspond to the detected ellipse, then the Error Score (Es) is formulated as

$$Es = c_1 \cdot CD + c_2 \cdot RM + c_3 \cdot AE \quad (34)$$

$$CD = |x_0^{true} - x_0^D| + |y_0^{true} - y_0^D|,$$

$$RM = |r_{max}^{true} - r_{max}^D| + |r_{min}^{true} - r_{min}^D|,$$

$$AE = |\theta_{true} - \theta_D|,$$

In Eq. (34), CD corresponds to the central point difference between the detected and ground-truth ellipse. RM represents the radio mismatch while AE symbolizes the angle error between both ellipses. The coefficient c_1 , c_2 and c_3 determine the relative importance of each deviation CD , RM and AE in the final error Es . In our experiments, the coefficients c_1 , c_2 and c_3 are set to 0.05, 0.1 and 0.2, respectively. Under this configuration, the angle error AE is strongly weighted with regard to CD and RM . To extend the Es for multiple ellipse detection, the averaged Es produced by the localization of several shapes is considered as the performance index. It is called the ME which is computed as follows:

$$ME = \frac{1}{NC} \sum_{q=1}^{NC} Es_q, \quad (35)$$

where NC corresponds to the number of ellipses actually contained in the edge map. Consequently, Es_q represents the Es produced by the q ellipse from the digital image. A lower ME value corresponds to better accuracy of the metaheuristic detector.

Metaheuristic approaches are, in general, complex programs with different operations and stochastic branches. Therefore, it is problematic, from a deterministic perspective, to develop a complexity analysis. As a result, the CT is considered to estimate the computational effort. Under this condition, CT expresses the CPU time spent by a metaheuristic scheme when it is under execution.

5.2. Experimental Comparison

In this subsection, the five metaheuristic schemes GWO [20], the WOA [21], the CSA [17], the GSA [22] and the CS [18] are compared in the detection of ellipses in digital images. The experiments consider even the capacity of each detector to recognize occluded or incomplete shapes such as arcs segments. In the analysis, the population size N_i has been set to 50 agents for all metaheuristic approaches. In the test, all approaches have been set with the parameters, which, according to their reported references, reach their best possible performance. The code for each algorithm has been obtained from their original sources.

The experimental results are divided into two parts. In Section 5.2.1, the performance of all metaheuristic algorithms is evaluated with regard to the ellipse detection in natural images. In Section 5.2.2,

Table 3 | Detection results of each metaheuristic algorithm for natural images.

		D1.1	D1.2	D1.3	D1.4
GWO	\overline{ME}	0.2750	0.1991	0.1051	0.1679
	SD	0.0220	0.0194	0.0067	0.0205
WOA	\overline{ME}	0.3245	0.2298	0.1185	0.1921
	SD	0.0304	0.0404	0.0098	0.0205
CSA	\overline{ME}	0.9272	0.9250	0.9250	0.7793
	SD	0.0591	0.0435	0.0421	0.0719
GSA	\overline{ME}	0.4913	0.4658	0.2509	0.2426
	SD	0.0536	0.0435	0.0421	0.0719
CS	\overline{ME}	0.8442	0.6915	0.9381	0.9991
	SD	0.0381	0.0536	0.0009	0.0001

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; ME, Multiple Error.

the robustness in the detection of each algorithm in the presence of complex scenarios is analyzed.

5.2.1. Detection in natural images

In comparison, all the five metaheuristic methods have been executed considering a fixed number of 1000 iterations ($k = Maxgen$). This stop criterion has been established to maintain compatibility with other similar works found in the literature [7–13]. To minimize the random influence in the results, each experiment is tested for 30 independent executions.

In the study, a representative set of four different images collected from the literature have been used. Figure 3 shows the complete set of images. In the figure, each image is labeled as D1.X that refers to the image X from experiment one. In the study, the ground-truth solutions for all images have been visually produced from their original versions.

The visual results in the application of all methods are depicted in Figure 4. Since every metaheuristic algorithm is executed 30 times over the same image, the detection results are presented in terms of the average \overline{ME} and its standard deviation (SD) in Table 1. Considering the detection results from Figure 4, it is clear that the GWO method presents the best qualitative accuracy followed by the WOA and the GSA whereas the CSA and the CS produce the worst outcomes.

According to Table 3, in terms of the average \overline{ME} index, GWO method performs better than the other algorithms in all. The WOA maintains the second-best scores in terms of \overline{ME} . The CSA and the CS produce the worst detection results. This fact emphasizes that GWO and WOA present a better balance between exploration and exploitation for this kind of problem. Their search strategies allow finding the best close solutions for multiple ellipse detection. On the other hand, the rest of the algorithms are attracted to suboptimal solutions. Considering the standard deviation SD index from Table 1, it is evident that the metaheuristic algorithm of CS maintains the best consistency; since it normally produces the smallest variations. After the CS, the GWO obtains the next best SD score. Although both methods CS and GWO present a low SD value, their meanings are very different. In the case of the CS method, its search strategy guides the detection process to only a prominent local minimum. Under such conditions, this suboptimal solution would always be

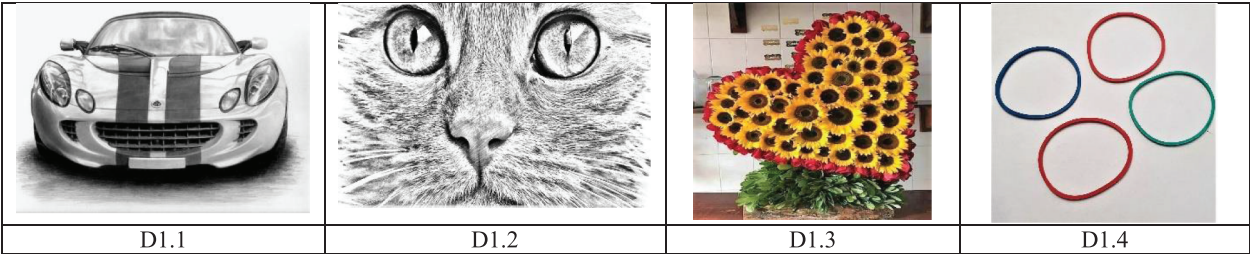


Figure 3 | Set of natural images used to evaluate the performance of each metaheuristic algorithm.



Figure 4 | Visual results of the metaheuristic detectors over natural images.

found in each execution, producing a low SD index. Different from CS, the GWO method obtains the best possible solution. However, due to its attraction mechanism, the solution moves around the best solution. This movement provokes a small displacement from the optimal detection. As a consequence, it can be established that the GWO preserves the best trade-off between accuracy (ME) and consistency (SD).

Once identified the GWO approach as the best shape detection method, a nonparametric analysis known as the Wilcoxon test [42,43] has been achieved to validate this conclusion. The Wilcoxon analysis permits to assess the performance differences between two computing methods. The experiment is conducted for the 5% (0.05) significance level considering the ME of 30 different executions. Table 4 presents the p -values produced by Wilcoxon test for the pairwise association of the two methods. For the experiment, four comparisons are generated: GWO vs. WOA, GWO vs. CSA, GWO vs. GSA and GWO vs. CS. Under the Wilcoxon method, it is considered as a null hypothesis that both related methods do not present an important statistic difference to each other. Contrarily, it is admitted as an alternative hypothesis that both algorithms maintain a notable difference. To show easily the results in Table 4, the figures ▲, ▼ and ► are assumed. ▲ represents that the GWO scheme performs statistically better than the compared method over a particular natural image. ▼ indicated that the GWO algorithm performs statistically worse than the compared approach and ► suggests that the Wilcoxon analysis cannot find any performance difference between both compared methods.

According to the results from Table 4, it is clear that all p -values in the GWO vs. CSA, GWO vs. GSA and GWO vs. CS columns are less than 0.05 (5% significance level) which is sufficient evidence against the null hypothesis indicating that the GWO algorithm maintains a better (▲) performance than the CSA, GSA and CS methods. The result of the Wilcoxon analysis establishes that the conclusion is statistically significant, which means that it has not happened because of the usual noise included in the experimental data. In the case of the images D1.3 and D1.4, the GWO approach presents a similar performance as the WOA scheme. This conclusion is observed from the column GWO vs. WOA, where the p -values of such images are higher than 0.05 (►). Under these conditions, there is no statistical difference regarding the performance between both methods. As a summary, the results of the Wilcoxon test confirm that the GWO method performs better than other approaches under analysis.

The comparison of the final average (ME) index cannot completely define the detection performance of a metaheuristic scheme. Under this condition, an analysis of the CT invested by every method has

been conducted. The objective is to evaluate the velocity with which a compared method reaches the final detection. Table 3 presents the average \overline{CT} of each algorithm considering 30 executions over the same image.

According to the results of Table 5, the CSA method obtain the best \overline{CT} scores, followed by the GWO, WOA, GSA and CS. However, the CSA method produces the worst detection results in terms of the average (ME). Therefore, as conclusion, the GWO presents the best balance between accuracy and computational cost.

5.2.2. Detection in complex scenarios

In this experiment, it is discussed the ability of each metaheuristic method to find suitable solutions in case of imperfect ellipses such as hand-drawn shapes or under noise conditions. This functionality is relevant since this problem appears typically in computer vision and image processing. In this paper, the ellipse detection problem has been conceived as an optimization task. Then, each metaheuristic method tries to find the best ellipse that matches a given shape under different conditions according to the objective function values $J(\cdot)$ for every candidate solution.

In the experiments, all metaheuristic algorithms have been executed considering a fixed number of 1000 iterations ($k = Maxgen$). Each experiment is tested for 30 independent executions in order to minimize the random influence in the results. In the analysis, a representative set of four different images collected from the literature have been used. Figure 5 shows the complete set of images. In the figure, each image is labeled as D2.X that refers to the image X from experiment two. In the study, the ground-truth solutions for all images have been visually produced from their original versions.

The visual results in the application of all methods are depicted in Figure 6. Since every metaheuristic algorithm is executed 30 times over the same image, the detection results are presented in terms of the average (ME) and its standard deviation (SD) in Table 6. From the detection results of Figure 6, it is also evident that the GWO method obtains the best qualitative accuracy in the shape localization followed by the WOA and the GSA, while the CSA and the CS produces the worst detections.

According to Table 6, the advantage of the GWO over CSA, GSA and CS is also confirmed in terms of the ME score. However, the GWO is only slightly better than the WOA approach. Similar to the case of natural images, the CS algorithm produces solutions with the smallest level of variation (SD) followed by the GWO

Table 4 | p -values generated by Wilcoxon analysis associating GWO vs. WOA, GWO vs. CSA, GWO vs. GSA and GWO vs. CS over the ME of 30 different executions (natural images).

	GWO vs. WOA	GWO vs. CSA	GWO vs. GSA	GWO vs. CS
D1.1	2.25E-04 ▲	8.76E-11 ▲	1.43E-09 ▲	4.84E-07 ▲
D1.2	6.67E-03 ▲	5.32E-11 ▲	3.97E-09 ▲	7.73E-07 ▲
D1.3	5.08E-01 ►	2.01E-11 ▲	6.28E-09 ▲	1.09E-07 ▲
D1.4	9.33E-02 ►	3.22E-11 ▲	9.43E-10 ▲	8.05E-08 ▲

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; ME, Multiple Error.

Table 5 | Average \overline{CT} in seconds for each metaheuristic detector for natural images.

	D1.1	D1.2	D1.3	D1.4
GWO	37.42	41.9	65.67	67.21
WOA	40.53	38.2	74.14	70.12
CSA	10.98	21.07	30.54	40.87
GSA	73.05	51.14	87.25	92.47
CS	85.12	67.29	97.18	128.14

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; CT, Computational Time.

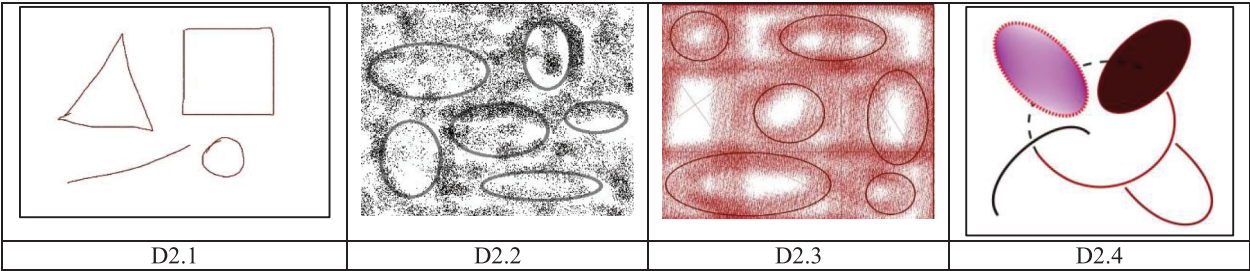


Figure 5 Set of images used to evaluate the performance of each metaheuristic algorithm under complex detection scenarios.

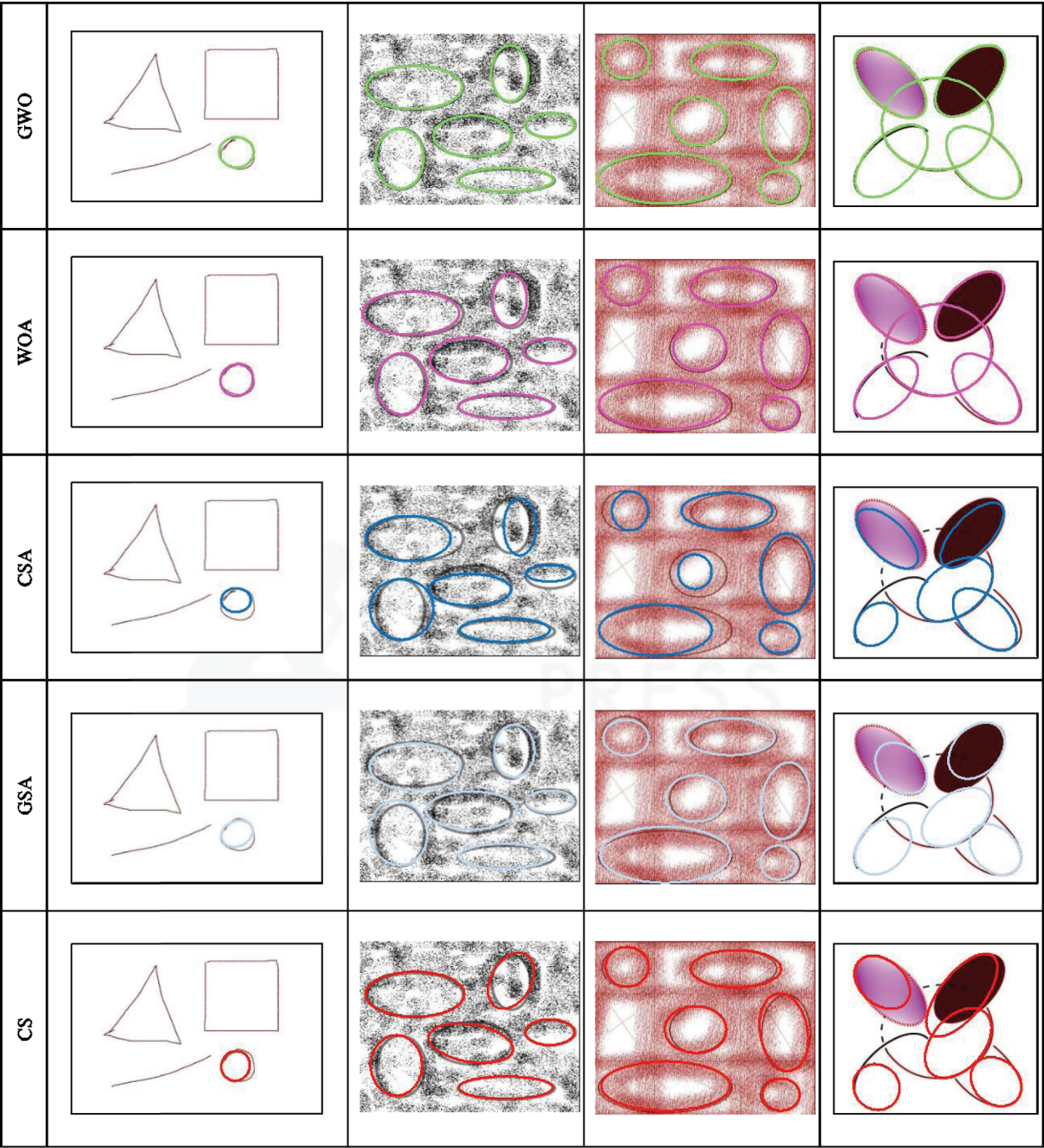


Figure 6 Visual results of the metaheuristic methods in the detection of complex scenarios.

Table 6 | Detection results of each metaheuristic algorithm for natural images.

		D2.1	D2.2	D2.3	D2.4
GWO	\overline{ME}	0.5388	0.09859	0.2974	0.3454
	SD	0.0215	0.0368	0.7253	0.0193
WOA	\overline{ME}	0.5494	0.2298	0.1185	0.1921
	SD	0.0335	0.0385	1.2178	0.0208
CSA	\overline{ME}	0.9225	0.8180	0.9790	0.9140
	SD	0.0514	0.0984	1.5961	0.0507
GSA	\overline{ME}	0.7072	0.1779	0.6915	0.4911
	SD	0.0361	0.0488	1.3925	0.0570
CS	\overline{ME}	0.8012	0.2813	0.4494	0.6330
	SD	0.0157	0.0225	0.4781	0.0313

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; ME, Multiple Error.

method. This robustness is valid for all images D2.1, D2.2, D2.3 and D2.4. The remarkable results of the GWO method are attributed to two factors. One of them is its attraction scheme. This mechanism allows incorporating elitism and diversity since it uses the three best so-far solutions. Such solutions are considered to obtain the direction for producing new search positions.

As a consequence, it is possible to find the best possible ellipse detection, even in complex conditions. This effect can be verified by the \overline{ME} value, which implies that the GWO method reaches the best accuracy among all metaheuristic schemes. The second factor is the use of its implementation characteristics. GWO does not include calibration parameters and its number of code lines is also relatively reduced. Under such conditions, it is possible to adapt the use of GWO to any application and produce consistent results.

The validation results of the Wilcoxon test for the detection of complex scenarios are presented in Table 7. They register the ME data produced during the 30 executions. According to Table 7, all p -values in the comparisons GWO vs. CSA, GWO vs. GSA and GWO vs. CS columns are less than 0.05, which implies that the GWO algorithm performs better than the CSA, GSA and CS schemes. In the case of the association GWO vs. WOA, the GWO algorithm reports a better performance in image D2.4, whereas in images D2.1, D2.2 and D1.3 the WOA produces a similar performance as the GWO approach.

To extend the performance analysis of each metaheuristic algorithm, the CT invested by them over images in complex scenarios has also been conducted. Table 8 presents the average \overline{CT} of each algorithm considering 30 executions over the same image. Similar to the case of the natural images, the CSA method obtain the best \overline{CT} scores, followed by the GWO, WOA, GSA and CS. Under such conditions, the GWO presents the best balance between accuracy and computational cost.

6. CONCLUSIONS

In this paper, a comparative analysis of the application of five recent metaheuristic schemes to the shape recognition problem has been presented. In the study, the GWO, WOA, CSA, GSA and CS have been considered. These algorithms have been successful in several new domains. Under this condition, the objective is to determine

Table 7 | p -values generated by Wilcoxon analysis associating GWO vs. WOA, GWO vs. CSA, GWO vs. GSA and GWO vs. CS over the ME of 30 different executions (detection in complex scenarios).

	GWO vs. WOA	GWO vs. CSA	GWO vs. GSA	GWO vs. CS
D2.1	8.24E-02 ►	3.02E-11 ▲	5.64E-07 ▲	3.21E-08 ▲
D2.2	6.73E-01 ►	1.41E-11 ▲	2.88E-10 ▲	4.46E-08 ▲
D2.3	2.40E-01 ►	4.98E-11 ▲	2.97E-10 ▲	5.10E-08 ▲
D2.4	7.95E-03 ▲	1.54E-11 ▲	7.18E-11 ▲	6.38E-09 ▲

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; ME, Multiple Error.

Table 8 | Average \overline{CT} in seconds for each metaheuristic detector for complex scenarios.

	D1.1	D1.2	D1.3	D1.4
GWO	29.49	6.87	7.39	21.95
WOA	32.41	8.51	8.24	24.37
CSA	9.41	2.74	1.76	10.03
GSA	37.21	12.87	10.52	29.97
CS	35.23	11.43	10.87	27.08

GWO, Grey Wolf Optimizer; WOA, Whale Optimizer Algorithm; CSA, Crow Search Algorithm; GSA, Gravitational Search Algorithm; CS, Cuckoo Search; CT, Computational Time.

their efficiency when they face a complex problem such as shape detection.

In order to provide an interesting comparison among all the methods, the ellipse detection has been selected. It has been considered the most complex recognition task with regard to lines and circular features. The comparison is divided into two parts. In the first part, the performance of all metaheuristic algorithms is evaluated with regard to the ellipse detection in natural images, while, in the second part, the robustness in the detection of each algorithm in the presence of complex scenarios is analyzed. In the study, several performance indexes have been considered to evaluate the accuracy of the detection and computational overload.

The results demonstrated that the GWO obtained the best performance indexes in comparison with the other approaches. This conclusion has been statistically validated according to the Wilcoxon test.

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest regarding the publication of this paper.

AUTHORS' CONTRIBUTIONS

The authors have contributed to all processes and stages of this research.

Funding Statement

This research has not been found for any institution.

REFERENCES

- [1] X. Bai, X. Yang, L.J. Latecki, Detection and recognition of contour parts based on shape similarity, *Pattern Recognit.* 41 (2008), 2189–2199.
- [2] K. Schindler, D. Suter, Object detection by global contour shape, *Pattern Recognit.* 41 (2008), 3736–3748.
- [3] W. Lu, T. Jinglu, Detection of incomplete ellipse in images with strong noise by iterative randomized Hough transform (IRHT), *Pattern Recognit.* 41 (2008), 1268–1279.
- [4] V. Ayala-Ramirez, C.H. Garcia-Capulin, A. Perez-Garcia, R.E. Sanchez Yanez, Circle detection on images using genetic algorithms, *Pattern Recognit. Lett.* 27 (2006), 652–657.
- [5] E. Lutton, P. Martinez, A genetic algorithm for the detection of 2D geometric primitives in images, in *Proceedings of the 12th International Conference on Pattern Recognition*, Jerusalem, Israel, 1994, pp. 526–528.
- [6] Y. Jie, K. Nawwaf, G. Peter, A multi-population genetic algorithm for robust and fast ellipse detection, *Pattern Anal. Appl.* 8 (2005), 149–162.
- [7] H.D. Cheng, G. Yanhui, Z. Yingtao, A novel hough transform based on eliminating particle swarm optimization and its applications, *Pattern Recognit.* 42 (2009), 1959–1969.
- [8] S. Das, S. Dasgupta, A. Biswas, A. Abraham, Automatic circle detection on images with annealed differential evolution, in *Eighth International Conference on Hybrid Intelligent Systems (HIS '08)*, Barcelona, Spain, 2008.
- [9] E. Cuevas, V. Osuna-Enciso, D. Oliva, Circle detection on images based on the Clonal Selection Algorithm (CSA), *Imaging Sci. J.* 63 (2015), 34–44.
- [10] J. Fourie, Robust circle detection using harmony search, *J. Optim.* 2017 (2017), 1–11.
- [11] E. Cuevas, F. Sención-Echauri, D. Zaldivar, M. Pérez-Cisneros, Multi-circle detection on images using Artificial Bee Colony (ABC) optimization, *Soft Comput.* 16 (2012), 281–296.
- [12] E. Cuevas, M. González, D. Zaldivar, M. Pérez-Cisneros, Multi-ellipses detection on images inspired by collective animal behavior, *Neural Comput. Appl.* 24 (2014), 1019–1033.
- [13] E. Cuevas, M. González, Multi-circle detection on images inspired by collective animal behavior, *Appl. Intell.* 39 (2013), 101–120.
- [14] A. Piotrowski, J. Napiorkowski, Some metaheuristics should be simplified, *Inf. Sci.* 427 (2018), 32–62.
- [15] M.P. Camargo, J.L. Rueda, I. Erlich, O. Añó, Comparison of emerging metaheuristic algorithms for optimal hydrothermal system operation, *Swarm Evol. Comput.* 18 (2014), 83–96.
- [16] H.M. Dubey, M. Pandit, B.K. Panigrahi, An overview and comparative analysis of recent bio-inspired optimization techniques for wind integrated multi-objective power dispatch, *Swarm Evol. Comput.* 38 (2018), 12–34.
- [17] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, *Comput. Struct.* 169 (2016), 1–12.
- [18] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in *2009 World Congress on Nature and Biologically Inspired Computing (NABIC 2009)*, Coimbatore, India, 2009, pp. 210–214.
- [19] S. Mirjalili, Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm, *Knowl. Based Syst.* 89 (2015), 228–249.
- [20] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.* 69 (2014), 46–61.
- [21] S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Softw.* 95 (2016), 51–67.
- [22] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, *Inf. Sci.* 179 (2009), 2232–2248.
- [23] S. Mirjalili, SCA a Sine Cosine Algorithm for solving optimization problems, *Knowl. Based Syst.* 96 (2016), 120–133.
- [24] M. Canayaz, A. Karci, Cricket behaviour-based evolutionary computation technique in solving engineering optimization problems, *Appl. Intell.* 44 (2016), 362–376.
- [25] S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, *Neural Comput. Appl.* 27 (2016), 1053–1073.
- [26] Y. Zhou, K. Su, L. Shao, A new chaotic hybrid cognitive optimization algorithm, *Cogn. Syst. Res.* 52 (2018), 537–542.
- [27] A. Mortazavi, V. Toğan, A. Nuhoğlu, Interactive search algorithm: a new hybrid metaheuristic optimization algorithm, *Eng. Appl. Artif. Intell.* 71 (2018), 275–292.
- [28] H. Shayanfar, F.S. Gharehchopogh, Farmland fertility: a new metaheuristic algorithm for solving continuous optimization problems, *Appl. Soft Comput.* 71 (2018), 728–746.
- [29] M. Beşkirli, İ. Koç, H. Haklı, H. Kodaz, A new optimization algorithm for solving wind turbine placement problem: binary artificial algae algorithm, *Renew. Energy.* 121 (2018), 301–308.
- [30] A. Tabari, A. Ahmad, A new optimization method: Electro-Search algorithm, *Comput. Chem. Eng.* 103 (2017), 1–11.
- [31] M.A. Al-Betar, M.A. Awadallah, Island bat algorithm for optimization, *Expert Syst. Appl.* 107 (2018), 126–145.
- [32] J.E. Bresenham, A linear algorithm for incremental digital display of circular arcs, *Commun. ACM.* 20 (1987), 100–106.
- [33] C. Caraveo, F. Valdez, O. Castillo, Optimization of fuzzy controller design using a new bee colony algorithm with fuzzy dynamic parameter adaptation, *Appl. Soft Comput.* 43 (2016), 131–142.
- [34] F. Fausto, A. Reyna-Orta, E. Cuevas, Á.G. Andrade, M. Pérez-Cisneros, From ants to whales: metaheuristics for all tastes, *Artif. Intell. Rev.* 53 (2020), 753–810.
- [35] E. Cuevas, A. Reyna-Orta, M.-A. Díaz-Cortes, A multimodal optimization algorithm inspired by the states of matter, *Neural Process. Lett.* 48 (2018), 517–556.
- [36] K.C. Tan, S.C. Chiam, A.A. Mamun, C.K. Goh, Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization, *Eur. J. Oper. Res.* 197 (2009), 701–713.
- [37] H. Du, Z. Wang, W.E.I. Zhan, Elitism and distance strategy for selection of evolutionary algorithms, *IEEE Access.* 6 (2018), 44531–44541.
- [38] B. Morales-Castañeda, D. Zaldivar, E. Cuevas, F. Fausto, A. Rodríguez, A better balance in metaheuristic algorithms: does it exist?, *Swarm Evol. Comput.* 54 (2020), 100671.
- [39] M. Črepinšek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms, *ACM Comput. Surv.* 45 (2013), 1–33.
- [40] X.-S. Yang, Swarm-based metaheuristic algorithms and no-free-lunch theorems, *Intech Open.* 2 (2018), 64.
- [41] A.P. Piotrowski, J.J. Napiorkowski, Some metaheuristics should be simplified, *Inf. Sci.* 427 (2018), 32–62.
- [42] F. Wilcoxon, Individual comparisons by ranking methods, *Biometrics.* 1 (1945), 80–83.
- [43] S. Garcia, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: a case study on the CEC'2005 Special session on real parameter optimization, *J. Heurist.* 15 (2009), 617.