Neural Computing and Applications (2019) 31:5497–5526
https://doi.org/10.1007/s00521-018-3376-6

ORIGINAL ARTICLE

Hybrid SCA–TLBO: a novel optimization algorithm for global optimization and visual tracking

Hathiram Nenavath[1] · Ravi Kumar Jatoth[1]

Received: 30 January 2017 / Accepted: 21 February 2018 / Published online: 13 March 2018

Abstract
A novel optimization algorithm called hybrid sine–cosine algorithm with teaching–learning-based optimization algorithm (SCA–TLBO) is proposed in this paper, for solving optimization problems and visual tracking. The proposed hybrid algorithm has better capability to escape from local optima with faster convergence than the standard SCA and TLBO. The effectiveness of this algorithm is evaluated using 23 benchmark functions. Statistical parameters are employed to observe the efficiency of the hybrid SCA–TLBO qualitatively, and results prove that the proposed algorithm is very competitive compared to the state-of-the-art metaheuristic algorithms. The hybrid SCA–TLBO algorithm is applied for visual tracking as a real thought-provoking case study. The hybrid SCA–TLBO-based tracking framework is used to experimentally measure object tracking error, absolute error, tracking detection rate, root mean square error and time cost as parameters. To reveal the capability of the proposed algorithm, a comparison of hybrid SCA–TLBO-based tracking framework and other trackers, viz. alpha–beta filter, linear Kalman filter and extended Kalman filter, particle filter, scale-invariant feature transform, particle swarm optimization and bat algorithm, is presented.

Keywords Metaheuristic · Global optimization · Visual tracking · Population-based algorithm

1 Introduction

Global optimization problems are continually unavoidable in current engineering and science fields. Mathematically, an optimization problem can be expressed as

$$\underset{x \in \Re^n}{\text{Minimize}}\ f_i(x), \quad (i = 1, 2, \ldots, M) \tag{1}$$

$$\text{s.t.} \quad h_j(x) = 0, \quad (j = 1, 2, \ldots, J) \tag{2}$$

$$g_k(x) \leq 0, \quad (k = 1, 2, \ldots, K) \tag{3}$$

Here $f_i(x)$, $g_k(x)$ and $h_j(x)$ are design vector functions

$$x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}. \tag{4}$$

Here the factor x_i of x is termed decision or design variables, and they are real discrete, continuous or the combination of two.

The $f_i(x)$ functions where $i = 1, 2, \ldots, M$ are termed the cost or objective functions, and in $M = 1$ case, there is only a single objective. The area covered by the design variables is called the search space or design space $\Re^n$, while the area formed by the cost function results is termed as the response or solution space. The inequalities for g_k and equalities for h_j are termed as constraints.

Grouping of optimization method can be accomplished in several ways. An uncomplicated way is to view the type of the method, and this segregates the algorithms into two groups: deterministic algorithms and stochastic algorithms. Deterministic algorithms seek a complicated process, and its value and paths of both decision variables and the functions are repeatable. Hill climbing [1] is an example of a deterministic algorithm, which follows the same path every time it is executed for the same preliminary point. The next set is stochastic algorithms, and it consistently has some of the randomness values. Genetic algorithms (GAs) [2] are the best example for stochastic algorithms, whose

✉ Hathiram Nenavath
hathiram.iisc@gmail.com

Ravi Kumar Jatoth
ravikumar@nitw.ac.in

1 Department of Electronics and Communication Engineering, National Institute of Technology, Warangal 506004, India

response to the population changes every time it is executed since it uses various pseudorandom values, and yet, there may not be any vast difference in the final solutions.

Also, there is a third group of algorithms which is a hybrid or a mixture of the stochastic algorithm and deterministic algorithm. The key idea is to start the deterministic algorithm with different initial points. Hill climbing with a random restart is the best example of this group.

Stochastic optimization algorithms are grouped into three main classifications: evolutionary-based [3], physics-based [4] and swarm intelligence-based [5] algorithms. Physics-based methods are divided into two categories: population-based and individual-based algorithms. Evolutionary-based methods are motivated by the laws of a natural progression. The search action begins with a randomly engender population which is going forward over consequent generations. The strong point of these algorithms is that the best entity is consistently joined to form the following generation of the entity. It allows the population to be optimized in the way of generations. GA is the best important evolution motivated method that replicates the Darwinian evolution. Evolution strategy (ES) [6], genetic programming (GP) [7], probability-based incremental learning (PBIL) [8] and biogeography-based optimizer (BBO) [9] are other evolution-based algorithms.

Physics-based algorithms impersonate the physical guidelines in the space. Gravitational search algorithm (GSA) [10], simulated annealing (SA) [11, 12], small world optimization algorithm (SWOA) [13], ray optimization (RO) [14] algorithm, central force optimization (CFO) [15], artificial chemical reaction optimization algorithm (ACROA) [16], galaxy-based search algorithm (GbSA) [17], colliding bodies optimization (CBO) [18] and charged system search (CSS) [19] are the physics-based algorithms.

The third classification of nature-inspired algorithms has swarm-based techniques that mimic the social behavior of groups of animals. Particle swarm optimization (PSO) [20], ant colony optimization (ACO) [21] and animal migration optimization (AMO) [22] are some of the swarm-based algorithms.

In the literature, there are also more metaheuristic algorithms motivated by human behaviors in the literature. Harmony search [23] (HS), teaching–learning-based optimization [24, 25] (TLBO), group search optimizer [26] (GSO), tabu (taboo) search [27] (TS), imperialist competitive algorithm [28] (ICA), group counseling optimization [29] (GCO) algorithm, colliding bodies optimization [18] (CBO), league championship algorithm (LCA) [30], soccer league competition [31] (SLC) algorithm, exchange market algorithm [32] (EMA), mine blast algorithm [33] (MBA), social-based algorithm [34] (SBA), firework algorithm [35] and seeker optimization algorithm [36] (SOA) are some of the popular algorithms.

The research in general in the present area may be divided into three key directions: coming up with novel algorithms, improving the existing algorithms and hybridizing different algorithms. The third research direction deals with hybridizing or mixing different algorithms to reform the achievement or solve distinct problems [37]. Recently, hybridization of algorithms is getting more popular due to their efficiency in handling many real-world issues affecting uncertainty, intricacy, imprecision, noisy habitat and ambiguity.

Despite the symbolic amount of newly recommended algorithms in optimization field, there is an essential query here as to why we need other optimization algorithms. This query can be answered referring to no free lunch (NFL) [38] theorem. It logically substantiates that no one can recommend a method for solving all optimization problems. In other words, it cannot be assured that the achievement of an algorithm in solving a definitive set of problems could solve every optimization problem with different nature and type.

A novel optimization algorithm called hybrid sine–cosine algorithm with teaching–learning-based optimization algorithm (SCA–TLBO) is proposed in this paper, for solving optimization problems and visual tracking. The performance of hybrid SCA–TLBO is benchmarked in following three test phases. In the first phase, a set of eminent test functions comprising of unimodal, multimodal and fixed-dimension multimodal functions is used to verify convergence, exploitation, exploration and local optima avoidance of hybrid SCA–TLBO. In the second phase, performance metrics (the best solutions and average fitness of solution during optimization) are employed to qualitatively observe and substantiate the performance of hybrid SCA–TLBO on shifted two-dimensional test functions. In the final phase, the hybrid SCA–TLBO algorithm is applied for visual tracking as a real thought-provoking case study to demonstrate and verify the performance of this algorithm in practice. The hybrid SCA–TLBO-based tracking framework is used to experimentally measure object tracking error, absolute error, tracking detection rate, root mean square error and time cost as parameters. A comparison of hybrid SCA–TLBO-based tracking framework and three probability-based trackers, viz. alpha–beta (α–β) filter, linear Kalman filter (LKF) and extended Kalman filter (EKF), particle filter (PF), scale-invariant feature transform (SIFT), particle swarm optimization (PSO) and bat algorithm (BA), is presented to unveil the capability of proposed algorithm.

The organization of this paper is as follows. The basics of SCA and TLBO are briefly introduced in Sect. 2. The proposed algorithm is presented in Sect. 3. Section 4 deals with the evaluation of proposed algorithm using twenty-three well-known benchmark functions. Application of hybrid SCA–TLBO for visual tracking is proposed in Sect. 5. Finally, the conclusion is given in Sect. 6.

2 SCA and TLBO

In this section, basic theories of SCA and TLBO are introduced, followed by a detailed introduction of hybrid SCA–TLBO algorithm in the next section.

2.1 Basic of SCA

The sine–cosine algorithm (SCA) is a new metaheuristic algorithm [39], and the solutions are updated based on the sine or cosine function as in Eqs. (5) or (6), respectively:

$$X_i^{g+1} = X_i^g + r_1 \times \sin(r_2) \times |r_3 P_i^g - X_i^g| \tag{5}$$

$$X_i^{g+1} = X_i^g + r_1 \times \cos(r_2) \times |r_3 P_i^g - X_i^g| \tag{6}$$

In general, the above two functions are combined into one function as in the following equation [39]:

$$X_i^{g+1} = \begin{cases} X_i^g + r_1 \times \sin(r_2) \times |r_3 P_i^g - X_i^g|; & r_4 < 0.5 \\ X_i^g + r_1 \times \cos(r_2) \times |r_3 P_i^g - X_i^g|; & r_4 \geq 0.5 \end{cases} \tag{7}$$

where X_i^{g+1} is the location of the present solution with ith dimension and gth generation or iteration, P_i is the destination solution with ith dimension, $|\cdot|$ specifies the absolute cost and r_1, r_2, r_3 and r_4 are random variables. The effects of sine and cosine on Eqs. (5) and (6) are shown in Fig. 1; it shows how the equations describe a space amidst two solutions in the search area.

The parameter r_1 is a random variable which responsible for determining the area of the next solution; this area may be either outside space between X_i and P_i or inside them. In [39], the authors update the parameter r_1 using Eq. (8) to balance exploration and exploitation [40].

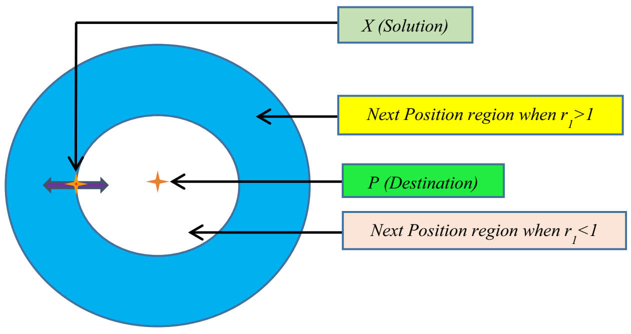

X (Solution)
Next Position region when r_1>1
P (Destination)
Next Position region when r_1<1

Fig. 1 Effects of sine and cosine in Eqs. (5) and (6) on the next position

$$r_1 = a - a\left(\frac{g}{G}\right) \tag{8}$$

where a is constant defined by the user, G is the maximum number of generations and g is the current generation.

The r_2 is a random variable which used to find the direction of the movement of the next solution (i.e., if it toward or outwards P_i). Also, the r_3 is a random variable which gives random weights for P_i in order to stochastically de-emphasize ($r_3 < 1$) or emphasize ($r_3 > 1$) the effect of desalination in defining the distance. The r_4 is used to switch between the sine and cosine functions as in Eq. (7). A theoretical model of the things of the sine and cosine functions with the range in [− 2, 2] is shown in Fig. 2.

2.2 Basic of TLBO

Rao et al. [24, 25] proposed a new population-based optimization algorithm called TLBO, inspired by the philosophy of teaching and learning, for the optimization of mechanical design problems. The main idea behind TLBO is the imitation of a traditional learning process consisting of a teacher phase and a learner phase. In the teacher phase, the teacher distributes his knowledge to all students (i.e., students learn from the teacher), whereas in the learner phase, students learn with the help of fellow students (i.e., students learn through interaction with other students). Under TLBO, the population is considered to be a group or class of learners. In optimization algorithms, the population consists of different design variables. A detailed description of TLBO can be found in [24, 25]. In this paper, only the two most important phases of the algorithm are considered.

2.2.1 Teacher phase

The teacher can be considered to be the most educated individual in the society. Hence, the student with the highest marks acts as a teacher during the teacher phase. The teacher tries to build up the mean of the class to her level. This, however, depends on the learning efficiency of the class. This is expressed as

$$X_{i_{\text{temp}}} = X_i + \text{random} \times (\text{Teacher} - \text{TF} \times \text{mean}) \tag{9}$$

where TF is the teaching factor that decides the value of mean to be changed and *mean* is the average of the class. TF can be either 1 or 2, (1 means student learns nothing from the teacher, and 2 means student learns everything from the teacher) which is again a heuristic step and decided randomly with equal probability as $\text{TF} = \text{ceil}\ [0.5 + \text{random}]$. The new solution, $X_{i_{\text{temp}}}$ is

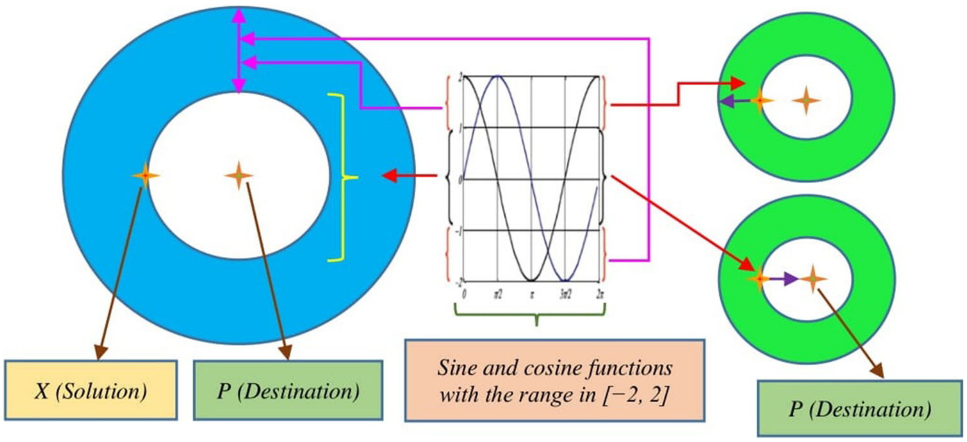


Fig. 2 Sine and cosine functions with the range in [− 2, 2] tolerate a solution to go beyond (outside the space among them) or around (inside the space among them) the destination

accepted only if it is better than the previous solution, that is,

$$X_i = \begin{cases} X_{i_{\text{temp}}}; & f\left(X_{i_{\text{temp}}}\right) > f(X_i) \\ X_i; & \text{otherwise} \end{cases}. \tag{10}$$

2.2.2 Learner phase

Teaching is not the only education students receive, but they also learn by interacting with each other. This is simulated in the learner phase. In each generation, two students X_m and X_n interact with each other, with the smarter one enhancing the others marks. It can be formulated as

$$X_{m_{\text{temp}}} = \begin{cases} X_m + \text{random} \times (X_m - X_n); & f(X_m) > f(X_n) \\ X_m + \text{random} \times (X_n - X_m); & f(X_n) > f(X_m) \end{cases} \tag{11}$$

In Eq. (11), $(X_m - X_n)$ is taken as a step. The temporary solution is accepted only if it is better than the previous solution, that is,

$$X_m = \begin{cases} X_{m_{\text{temp}}}; & f\left(X_{m_{\text{temp}}}\right) > f(X_m) \\ X_m; & \text{otherwise} \end{cases}. \tag{12}$$

3 Proposed algorithm

The hybridizing SCA with TLBO (hybrid SCA–TLBO) is introduced in detail in this section. Both SCA and TLBO are population-based algorithms with individuals being considered as positions in SCA and learners in TLBO. In SCA, the way algorithm moves toward the next position is based on random and adaptive variables; therefore, the satisfactory solution cannot always be found every time. For some cases, SCA is well capable of making full use of the population knowledge and has proved good performance on some benchmark functions. However, sometimes, SCA may not proceed to the optimal solutions to some complex problems. In order to overcome these disadvantages, the SCA method is hybridized with TLBO algorithm to form a novel hybrid SCA–TLBO method. In general, a hybrid metaheuristic method attempts to combine two or more metaheuristic methods. This can fully exert the useful features from the original algorithms. In the proposed hybrid SCA–TLBO, the idea of TLBO is introduced into the SCA in order to improve its search ability.

In hybrid SCA–TLBO, standard SCA algorithm is utilized to search globally for the purpose of making most solution move toward a more promising area. After exploration stage, TLBO algorithm is utilized to search locally with a small step to get the final best solution. Based on the mainframe of hybrid SCA–TLBO, the SCA emphasizes the diversification at the beginning of the search with a big step to explore the search space extensively and evade trapping into local optima, while later TLBO algorithm focuses on the intensification and lets the individuals move toward the best solution at the later stage of the optimization process. Also, this method can cope with the conflict global and local search effectively.

Initially, it evaluated the fitness of each learner and obtained the best solution (teacher). Assign the best solution to SCA as the position of the destination point in Eq. 7. The process might be better understood by the pseudocode, and the flow chart of hybrid SCA–TLBO is shown in Figs. 3 and 4.

Fig. 3 Pseudocode of hybrid SCA–TLBO

```
1:  Initialize set of search agents (learners), $G, a, r_1, r_2, r_3, r_4, X_{Max}, X_{Min}$ and $TF$
2:  while (stopping condition not met)  do
3:  For each search agent
4:  Initialize population and evaluate the fitness of each learner Xi;
5:  Update best solution obtained as Teacher (destination position)
6:                      if $r_4 < 0.5$
7:                      $X_i^{g+1} = X_i^g + r_1 * \sin(r_2) * |Teacher \cdot r_3 - X_i^g|$
8:                      else
9:                      $X_i^{g+1} = X_i^g + r_1 * \cos(r_2) * |Teacher \cdot r_3 - X_i^g|$
10:                 end if
11: end for
12: For each search agent do
                                /*Teacher Phase* /
13: Select the best learner as teacher $(X_{Teacher})$ and calculate $X_{mean}^g$;
14: Calculate $X_{i,new}$ for learner i using Eq. (16);
15: Perform fitness evaluation on $X_{i,new}$
16:                     if $f(X_{i,new}) < f(X_i)$   then
17:                     $X_i = X_{i,new}$;   $f(X_i) = f(X_{i,new})$;
18:                     end if
                                /*Learner Phase*/
19: Choose $Teacher(i)$ and select a neighboring search agent $(X_m)$ randomly
20: Perform neighborhood search using the Eq. (17).
21:                     if $f(X_{i,new}) < f(X_i)$   then
22:                     $X_i = X_{i,new}$;   $f(X_i) = f(X_{i,new})$;
23:                     end if
24: Otherwise, select search agents randomly from the total class, $(X_m \& X_n)$
25:                 if  $f(X_m) < f(X_n)$ then
26:                         Calculate $X_{i,new}$ for learner i using Eq. (18) ;
27:                 else /* $f(X_n) < f(X_m)$ * /
28:                         Calculate $X_{i,new}$ for learner i using Eq. (19) ;
29:                 end if
30: Perform fitness evaluation on $X_{i,new}$
31:                 if $f(X_{i,new}) < f(X_i)$   then
32:                         $X_i = X_{i,new}$;   $f(X_i) = f(X_{i,new})$;
33:                 end if
34:                 end for
35:  $g = g + 1$;   Select a new Teacher
36: end while
```

3.1 Steps of hybrid SCA–TLBO algorithm

The clear steps of the proposed hybrid SCA–TLBO algorithm are presented below. Moreover, the detailed pseudocode and flow chart of hybrid SCA–TLBO are also outlined in Figs. 3 and 4.

Step 1 Initialize the parameters of the algorithm, such as maximum iteration or generation (G), number of search agents (N), a, r_2, r_3, r_4. The initial population is defined as follows:

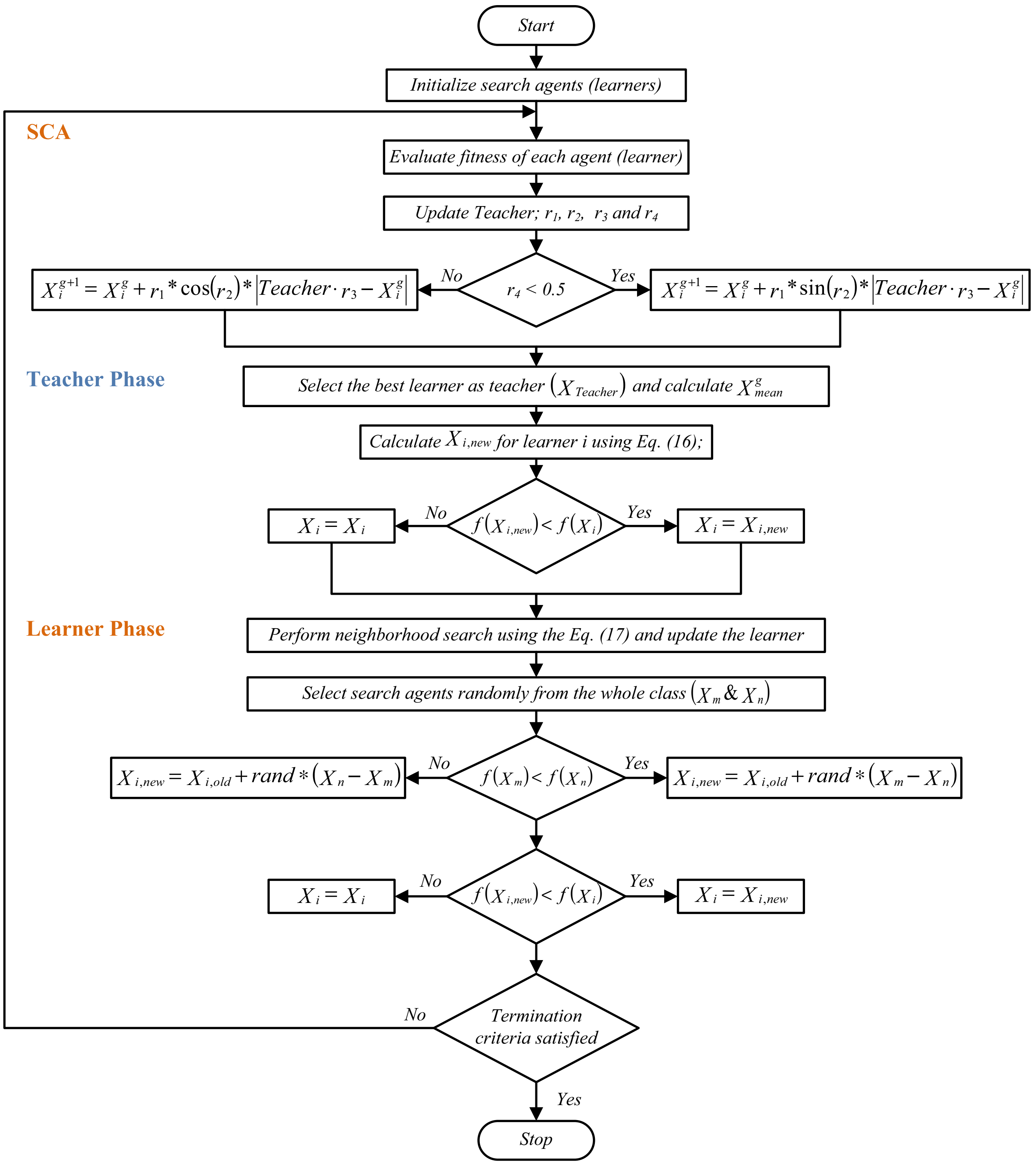

Start
Initialize search agents (learners)
SCA
Evaluate fitness of each agent (learner)
Update Teacher; r_1, r_2, r_3 and r_4
$r_4 < 0.5$
No
Yes
$X_i^{g+1} = X_i^g + r_1 * \cos(r_2) * \left|Teacher \cdot r_3 - X_i^g\right|$
$X_i^{g+1} = X_i^g + r_1 * \sin(r_2) * \left|Teacher \cdot r_3 - X_i^g\right|$
Teacher Phase
Select the best learner as teacher $(X_{Teacher})$ and calculate X_{mean}^g
Calculate $X_{i,new}$ for learner i using Eq. (16);
$f(X_{i,new}) < f(X_i)$
$X_i = X_i$
$X_i = X_{i,new}$
Learner Phase
Perform neighborhood search using the Eq. (17) and update the learner
Select search agents randomly from the whole class $(X_m \& X_n)$
$f(X_m) < f(X_n)$
$X_{i,new} = X_{i,old} + rand * (X_n - X_m)$
$X_{i,new} = X_{i,old} + rand * (X_m - X_n)$
$f(X_{i,new}) < f(X_i)$
$X_i = X_i$
$X_i = X_{i,new}$
Termination criteria satisfied
No
Yes
Stop

Fig. 4 Flow chart of proposed hybrid SCA–TLBO algorithm

$$X_i = X_{\mathrm{Min}} + \mathrm{rand} \cdot (X_{\mathrm{Max}} - X_{\mathrm{Min}}) \tag{13}$$

where X_{Min} and X_{Max} are the lower bound and upper bound of the variables, rand generates random numbers uniformly between zero and one.

Step 2 Evaluate the fitness of all search agents (learners) X, i.e., calculate the function values of X for all N and choose the best fitness value to be the teacher (destination position) of the present generation.

Step 3 Generate r_1 and teaching factor (TF) as follows:

Table 1 Twenty-three benchmark functions used in the experimental studies

Function	Formulation	D	Range	C	$F_{\min}$
Sphere	$F_1(x) = \sum_{i=1}^{D} x_i^2$	30	[− 100, 100]	US	0
Schwefel 2.22	$F_2(x) = \sum_{=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	[− 10, 10]	UN	0
Schwefel 1.2	$F_3(x) = \sum_{=1}^{D} (\sum_{j=1}^{i} x_j^2)$	30	[− 100, 100]	UN	0
Schwefel 2.21	$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \leq \mathrm{i} \leq \mathrm{D}\}$	30	[− 100, 100]	US	0
Rosenbrock	$F_5(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[− 30, 30]	UN	0
Step	$F_6(x) = \sum_{i=1}^{D} ([x_i + 0.5])^2$	30	[− 100, 100]	US	0
Quartic	$F_7(x) = \sum_{i=1}^{D} i x_i^4 + \mathrm{random}[0, 1)$	30	[− 1.28, 1.28]	US	0
Schwefel	$F_8(x) = -\sum_{i=1}^{D} \left(x_i \sin \sqrt{\lvert x_i \rvert} \right)$	10	[− 500, 500]	MS	− 418.9829*D
Rastrigin	$F_9(x) = 10D + \sum_{i=1}^{D} (x_i^2 - 10 \cos (2\pi x_i))$	30	[− 5.12, 5.12]	MS	0
Ackley	$F_{10}(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i) \right) + 20 + e$	30	[− 32, 32]	MN	0
Griewank	$F_{11}(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	[− 600, 600]	MN	0
Penalized	$F_{12}(x) = \frac{\pi}{D} \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \left[1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_D - 1)^2 \right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $where, y_i = 1 + \frac{x_i + 1}{4}, and, u(x_i, a, k, m) = \begin{cases} k\,(x_i - a)^m; x_i > a \\ 0; -a < x_i < a \\ k\,(-x_i - a)^m; x_i < -a \end{cases}$	30	[− 50, 50]	MN	0
Penalize 2	$F_{13}(x) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{D} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_i) + 1 \right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_n) \right] \right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	30	[− 50, 50]	MN	0

Table 1 (continued)

Function	Formulation	*D*	Range	*C*	$F_{\min}$
Foxholes	$F_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{D} (x_i - a_{ij})^6}\right]^{-1}$	2	[− 65.536, 65.536]	MS	0.998004
Kowalik	$F_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[− 5, 5]	MN	0.0003075
Six-hump camel back	$F_{16}(x) = 4 x_1^2 - 2.1 x_1^4 + \frac{1}{3} x_1^6 + x_1 x_2 - 4 x_2^2 + 4 x_2^4$	2	[− 5, 5]	MN	− 1.0316285
Branin	$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right) \cos x_1 + 10$	2	[− 5,5]	MN	0.398
Goldstein–Price	$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2 (10 - 14 x_1 + 3 x_1^2 - 14 x_2 + 6 x_1 x_2 + 3 x_2^2)\right] * \left[30 + (2 x_1 - 3 x_2^2)(18 - 32 x_1 + 12 x_1^2 + 48 x_2 - 36 x_1 x_2 + 27 x_2^2)\right]$	2	[− 5, 5]	MN	3
Hartman 3	$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2\right]$	3	[− 5, 5]	MN	− 3.862782
Hartman 6	$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2\right]$	6	[− 5, 5]	MN	− 3.32236

Table 1 (continued)

Function	Formulation	D	Range	C	$F_{\min}$
Shekel 5	$F_{21}(x) = -\sum_{i=1}^{5}[(x-a_i)(x-a_i)^T + c_i]^{-1}$	4	[− 5, 5]	MN	− 10.1532
Shekel 7	$F_{22}(x) = -\sum_{i=1}^{7}[(x-a_i)(x-a_i)^T + c_i]^{-1}$	4	[− 5, 5]	MN	− 10.4029
Shekel 10	$F_{23}(x) = -\sum_{i=1}^{10}[(x-a_i)(x-a_i)^T + c_i]^{-1}$	4	[− 5, 5]	MN	− 10.5364

$F_{\min}$ global minima, C function characteristics, U unimodal, M multimodal, S separable, N non-separable, D dimension

$$r_1 = a - a\left(\frac{g}{G}\right) \tag{14}$$

$$\mathrm{TF} = \mathrm{ceil}\ [0.5 + \mathrm{random}] \tag{15}$$

where g is the current generation.

Step 4 During the SCA, all the search agents are updated according to Eq. (7), and the search agent solutions which are accepted are passed to the teacher phase.

Step 5 For each search agent X_i, the teacher phase is implemented as follows:

$$X_{i,\mathrm{new}} = X_{i,\mathrm{old}} + \mathrm{rand} \times \left(X_{\mathrm{Teacher}} - \mathrm{TF} \cdot X^{g}_{\mathrm{mean}}\right) \tag{16}$$

where $X_{i,\mathrm{old}}$ and $X_{i,\mathrm{new}}$ are the old and new positions of the ith learner, rand is a random number, the search agent or learner with the value of best fitness in the present generation is chosen as X_{Teacher}. If the ith search agent's new solution (or position) is best, then it will be accepted; otherwise, the old one is unchanged.

Step 6 In case of the learner phase, each learner or search agent can learn from their neighbors or the entire class. The primary learning process ensures excellent local search ability, and the next assures good global searching characteristic. In case of neighborhood search, choose a best search agent neighbor (Teacher(i)), select a neighboring search agent (X_m) randomly and update the search agents according to the following equation:

$$X_{i,\mathrm{new}} = X_{i,\mathrm{old}} + \mathrm{rand} \times \left(\mathrm{Teacher}(i) - X_{i,\mathrm{old}}\right) + \mathrm{rand} \times \left(X_m - X_{i,\mathrm{old}}\right) \tag{17}$$

Accept fitness value of $X_{i,\mathrm{new}}$ if it is enhanced; otherwise, the position of learner's remnant unchanged.

Step 7 Otherwise, select search agents randomly from the total class, and update the search agents according to the following Equations:

If X_m is better than X_n, then

$$X_{i,\mathrm{new}} = X_{i,\mathrm{old}} + \mathrm{rand} \times (X_m - X_n) \tag{18}$$

Else

$$X_{i,\mathrm{new}} = X_{i,\mathrm{old}} + \mathrm{rand} \times (X_n - X_m) \tag{19}$$

Step 8 If the new position or solution $X_{i,\mathrm{new}}$ is better than the old one $X_{i,\mathrm{old}}$, $X_{i,\mathrm{new}}$ will be accepted; otherwise, the position of the ith search agent is unchanged. The accepted search agent solutions are passed to the next generation.

Step 9 If the stopping criteria are satisfied, then terminate the process and print the final value of the solution. Otherwise, go to Step 3.

4 Experimental results and discussion

To check the efficiency of the proposed algorithm, the hybrid SCA–TLBO algorithm is benchmarked with twenty-three popular and classical benchmark functions employed by many researchers [39, 41, 42]. The twenty-three benchmark functions (shown in Table 1) can be classified into three groups: unimodal, multimodal and fixed-dimension multimodal benchmark functions. Hybrid SCA–TLBO, SCA, PSO, DE and TLBO algorithms have several parameters that should be initialized before running. Table 2 shows the initial values of the basic parameters for these algorithms. For SCA, TLBO and hybrid SCA–TLBO, the same parameters are used.

The hybrid SCA–TLBO algorithm and other algorithms (SCA, PSO, DE and TLBO) are simulated with the initial population size NP $=$ 30, 50 and 100 on each twenty-three benchmark functions for comparison. The experimental results are comprised of several statistical parameters, such as average, standard deviation, median and worst of the best-so-far solution in the last generation are reported. Statistical parameter results are presented in Tables 3, 4, 5, 6, 7 and 8. To check the efficiency of the proposed algorithm, the same maximum number of generations and the same population size are set for all algorithms. In the present work, the initial population size NP $=$ 30, 50 and 100 is selected. The outcomes are averaged over initial population size, and the best outcomes are shown in bold type in tables.

According to Derrac et al. [43], to improve the evaluation of evolutionary algorithms achievement, statistical tests should be conducted. A statistical analysis is essential to verify that a proposed novel algorithm is offering a significant advancement over other existing algorithms for a particular problem.

4.1 Unimodal functions

The unimodal functions have no local solution, and there is only one global solution. Consequently, they are used to examine heuristic optimization algorithms in terms of convergence rate. Functions F_1–F_7 are unimodal functions, and the results for these unimodal functions are shown in Tables 3, 4, 5, 6, 7 and 8. As given in these tables, the hybrid SCA–TLBO outperforms the other algorithms on the unimodal benchmark functions regarding the mean, standard deviation, median and worst value of the results. Therefore, this is evidence that the proposed algorithm has high performance in finding the global solution of unimodal benchmark functions. Figure 5 shows the graphical analysis results of the ANOVA tests.

4.2 Multimodal high-dimensional functions

For multimodal functions F_8–F_{13} with many local minima, the final results are more important because this function can reflect the algorithm's ability to escape from poor local optima and obtain the near-global optimum. We have tested the experiments on F_8–F_{13} where the number of local minima increases exponentially as long as the dimension of the function increases. As the results of mean, standard deviation, median and the worst value are shown in Tables 3, 4, 5, 6, 7 and 8, the hybrid SCA–TLBO performs better than the other algorithms on the multimodal high-dimensional benchmark functions.

Table 2 Initial parameters for hybrid SCA–TLBO, SCA, PSO, DE and TLBO (rand: random number between [0, 1])

SCA		PSO		DE		TLBO	
Parameters	Values	Parameters	Values	Parameters	Values	Parameters	Values
Number of search agents (NP)	30, 50, 100	Number of particles (NP)	30, 50, 100	Number of search agents (NP)	30, 50, 100	Population size (NP)	30, 50, 100
r_2	$2\pi \times$ rand	w	Varies linearly 0.9–0.4	Scaling factor (beta)	0.2–0.8	TF	1 or 2
r_3	$2 \times$ rand	C_1	2	Crossover probability (p_{CR})	0.2	Random	ε [0,1]
r_4	Rand	C_2	2	Max generation	1000	Max generation	1000
Max generation	1000	$V_{\max}$	4	Stopping criterion	Max generation	Stopping criterion	Max generation
a	2	Stopping criterion	Max generation				
Stopping criterion	Max generation						

Table 3 Minimization results of twenty-three benchmark functions with initial population size NP = 30 (Ave: average, SD: standard deviation)

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_1	**0**	**0**	3.88E−27	9.89E−27	21.2022	21.7187	4.95E−42	6.07E−42	5.47E−222	0
F_2	**0**	**0**	1.27E−19	2.07E−19	2.443	2.4105	8.36E−25	5.35E−25	1.16E−111	1.82E−111
F_3	**0**	**0**	4.58E−08	1.28E−07	95.5016	52.5843	0.29075	0.254	2.71E−100	5.67E−100
F_4	**0**	**0**	2.54E−08	5.35E−08	8.6909	4.2718	2.35E−07	1.42E−07	2.35E−93	4.78E−93
F_5	4.9488	0.41834	7.118	0.25018	5021.7729	9520.2362	2.0313	1.909	0.48528	0.31321
F_6	**1.57E−32**	**1.69E−32**	0.31179	0.11916	31.0031	41.1201	6.88E−21	8.17E−21	2.33E−07	9.74E−08
F_7	**0.00017634**	**6.40E−05**	0.0007298	0.0005702	0.014187	0.012918	0.003026	0.0012265	0.0029011	0.0017035
F_8	− 3609.2222	87.8182	− 2313.3951	248.3506	− 3986.0388	1688.1824	− 4177.99	37.4535	0.0005031	290.5444
F_9	**0**	**0**	1.5968	5.0495	24.7889	10.6151	0	0	2.501	1.8793
F_{10}	**3.02E−15**	**1.83E−15**	2.40E−14	3.04E−14	5.1873	1.1398	4.44E−15	0	4.44E−15	0
F_{11}	**0**	**0**	0.090781	0.19771	1.254	0.68186	0	0	0.008377	0.011378
F_{12}	1.01E−07	9.72E−08	0.070837	0.019619	5.7617	5.6169	4.71E−32	1.15E−47	5.05E−32	2.47E−33
F_{13}	0.03001	0.067453	0.22513	0.07115	9.4578	6.1732	1.35E−32	2.89E−48	0.0021975	0.0046327
F_{14}	**0.998**	**0**	1.5935	0.95823	1.295	0.93927	0.998	0	0.998	0
F_{15}	**0.00038276**	**0.000238**	0.0008991	0.000331	0.0058969	0.0036945	0.000665	0.00013999	0.0003991	0.0002896
F_{16}	− **1.0316**	**0**	− 1.0316	1.60E−05	− 1.0316	1.48E−16	− 1.0316	2.8514E−05	− 1.0316	0
F_{17}	**0.39789**	**0**	0.39898	0.0011636	0.39871	0.0014329	0.39789	0	0.39789	0
F_{18}	**3**	**0**	3	3.96E−05	3	5.34E−16	3	6.94E−16	3	8.63E−16
F_{19}	− **3.8628**	**0**	− 3.8542	0.0006098	− 3.8628	8.63E−16	− 3.8628	9.36E−16	− 3.8628	9.36E−16
F_{20}	− 3.3101	0.037588	− 2.8029	0.38047	− 3.2735	0.061513	− 3.322	4.44E−16	− 3.2781	− 3.2781
F_{21}	− **10.1532**	**0**	− 3.0066	2.2011	− 8.8957	2.7143	− 10.0205	0.4186	− 4.879	0.07975
F_{22}	− **10.4029**	**1.18E−15**	− 4.1502	1.3397	− 9.6392	2.415	− 9.6272	2.4111	− 9.8714	1.6808
F_{23}	− **10.5364**	**1.87E−15**	− 4.4381	0.97997	− 8.1416	3.8574	− 10.5082	0.089204	− 5.1399	0.72598

Bold values represent best results

Table 4 Minimization results of twenty-three benchmark functions with initial population size NP = 30 (med: median)

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Med	Worst	Med	Worst	Med	Worst	Med	Worst	Med	Worst
F_1	**0**	**0**	1.16E−29	3.12E−26	12.9394	71.7866	2.50E−42	1.74E−41	1.71E−223	3.34E−221
F_2	**0**	**0**	6.58E−20	6.83E−19	1.7354	9.0009	7.54E−25	1.74E−24	6.88E−112	6.15E−111
F_3	**0**	**0**	2.84E−11	4.07E−07	105.5108	173.7757	0.19803	0.7853	2.94E−101	1.72E−99
F_4	**0**	**0**	4.92E−10	1.68E−07	7.0856	14.8275	2.56E−07	4.49E−07	6.85E−94	1.57E−92
F_5	5.1937	5.3704	7.2289	7.3661	937.3661	2842.7823	1.4967	5.1151	0.41529	1.1795
F_6	**1.23E−32**	**6.16E−32**	0.29383	0.48389	16.4782	125.0165	3.14E−21	2.35E−20	2.51E−07	3.31E−07
F_7	**0.00017149**	**0.0002608**	0.0006444	0.0020895	0.011674	0.040038	0.003121	0.004668	0.0029279	0.005031
F_8	− 3597.6256	− 3474.2817	− 2197.3255	− 2136.702	− 3817.0586	− 2363.8598	− 4189.83	− 4071.39	− 3716.0755	− 3084.2989
F_9	**0**	**0**	6.54E−12	15.9679	20.3734	48.0048	0	0	2.0579	4.9748
F_{10}	**4.44E−15**	**4.44E−15**	7.99E−15	1.00E−13	5.4343	7.1843	4.44E−15	4.44E−15	4.44E−15	4.44E−15
F_{11}	**0**	**0**	9.80E−07	0.61623	1.1417	2.6858	0	0	1.28E−15	0.027037
F_{12}	6.61E−08	3.38E−07	0.075662	0.095117	4.251	16.3192	4.71E−32	4.71E−32	5.11E−32	5.49E−32
F_{13}	4.38E−07	0.19968	0.24505	0.32012	7.7523	19.6452	1.35E−32	1.35E−32	1.41E−32	0.010987
F_{14}	**0.998**	**0.998**	0.99809	2.9821	0.998	3.9683	0.998	0.998	0.998	0.998
F_{15}	**0.00030749**	**0.0010602**	0.0007456	0.0014997	0.0054318	0.013469	0.00072	0.000763	0.0003075	0.0012232
F_{16}	− **1.0316**	− **1.0316**	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316
F_{17}	**0.39789**	**0.39789**	0.39835	0.40126	0.39829	0.40274	0.39789	0.39789	0.39789	0.39789
F_{18}	**3**	**3**	3	3.0001	3	3	3	3	3	3
F_{19}	− **3.8628**	− **3.8628**	− 3.8545	− 3.853	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628
F_{20}	− 3.322	− 3.2031	− 3.0014	− 1.9211	− 3.3196	− 3.1995	− 3.322	− 3.322	− 3.2781	− 3.2781
F_{21}	− **10.1532**	− **10.1532**	− 3.1737	− 0.4982	− 10.1532	− 2.6305	− 10.1532	− 8.8291	− 4.8987	− 4.7351
F_{22}	− **10.4029**	− **10.4029**	− 4.707	− 0.90936	− 10.4029	− 2.7659	− 10.4029	− 2.7659	− 10.4029	− 5.0877
F_{23}	− **10.5364**	− **10.5364**	− 4.6828	− 2.8115	− 10.5361	− 2.4273	− 10.5362	− 10.2543	− 4.9967	− 4.629

Bold values represent best results

Table 5 Minimization results of twenty-three benchmark functions with initial population size NP = 50

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_1	**0**	**0**	3.49E−32	6.90E−32	0.90315	1.1512	6.96E−42	1.08E−41	2.99E−219	0
F_2	**0**	**0**	3.41E−21	9.41E−21	0.3673	0.27514	1.63E−24	9.17E−25	1.26E−109	1.14E−109
F_3	**0**	**0**	7.20E−15	1.29E−14	37.2915	33.5569	0.22967	0.11196	1.93E−96	4.90E−96
F_4	**0**	**0**	3.95E−11	4.50E−11	3.5424	1.8715	2.26E−07	1.16E−07	3.00E−92	3.26E−92
F_5	4.3248	0.16152	6.7985	0.36179	91.5765	74.5071	1.5134	1.5438	0.028543	0.077837
F_6	**0**	**0**	0.32168	0.17546	1.0488	1.0618	6.88E−23	8.17E−22	6.16E−34	1.30E−33
F_7	**0.00022126**	**6.03E−05**	0.000683	0.000561	0.005534	0.006057	0.00278	0.0008253	0.0021902	1.10E−03
F_8	− 3620.1876	91.3188	− 2302.54	93.9009	− 3986.04	1688.182	− 4189.83	0	− 3682.985	180.3579
F_9	**0**	**0**	0.000288	0.000912	22.448	11.5368	0	0	1.99	2.2494
F_{10}	**3.02E−16**	**0**	4.80E−15	1.12E−15	3.0004	1.1084	4.44E−15	0	4.44E−15	0
F_{11}	**0**	**0**	4.17E−10	1.32E−09	0.67802	0.16914	0	0	0.0025027	0.0077556
F_{12}	4.63E−08	2.17E−08	0.062079	0.019957	2.4396	1.1245	4.71E−32	1.15E−47	4.76E−32	1.21E−33
F_{13}	2.19E−07	1.09E−07	0.20168	0.080588	0.31852	0.34653	1.35E−32	2.89E−48	1.35E−32	2.89E−48
F_{14}	**0.998**	**0**	1.3935	0.75823	1.1964	0.62743	0.998	0	0.998	0
F_{15}	**0.00038276**	1.29E−06	0.000792	0.000468	0.000647	0.000488	0.000605	7.84E−05	0.0003079	0.00023803
F_{16}	− **1.0316**	**0**	− 1.0316	1.25E−05	− 1.0316	0.00E+00	− 1.0316	0	− 1.0316	0
F_{17}	**0.39789**	**0**	0.39838	0.00041	0.39789	0	0.39789	0	0.39789	0
F_{18}	**3**	**0**	3	1.91E−06	3	3.63E−16	3	6.10E−16	3	2.96E−16
F_{19}	− **3.8628**	**0**	− 3.8546	0.000291	− 3.8628	8.50E−16	− 3.8628	9.36E−16	− 3.8628	9.36E−16
F_{20}	− **3.322**	**1.03E−04**	− 3.0678	0.060446	− 3.2982	0.050136	− 3.322	0.0013901	− 3.2483	0.08567
F_{21}	− **10.1532**	**0**	− 3.5548	2.4856	− 5.8911	3.1247	− 10.1532	3.08E−15	− 10.1532	1.18E−15
F_{22}	− **10.4029**	**1.18E−15**	− 3.7603	2.5092	− 5.9614	3.9106	− 10.4029	2.37E−15	− 10.4029	1.32E−15
F_{23}	− **10.5364**	**1.78E−15**	− 4.6381	0.77997	− 5.9094	3.3467	− 10.5364	1.79E−15	− 10.5364	2.13E−15

Bold values represent best results

4.3 Multimodal low-dimensional functions

For F_{14}–F_{23} with only a few local minima, the dimension of the function is also small. The significant difference compared with functions F_8–F_{13} is that functions F_{14}–F_{23} appear to be simpler than F_8–F_{13} due to their low dimensionalities and a lower number of local minima. As the results of mean, standard deviation, median and the worst value are shown in Tables 3, 4, 5, 6, 7 and 8, the hybrid SCA–TLBO performs better than the other algorithms on the multimodal low-dimensional benchmark functions.

4.4 Convergence behavior analysis

To confirm the convergence of the hybrid SCA–TLBO algorithm, two metrics are employed: convergence rate and average fitness of all search agents.

Figure 6 show the convergence rate of the of the hybrid SCA–TLBO, SCA, PSO, DE and TLBO algorithms for functions F_1–F_{23}. The fitness of the best solution in each generation is saved and drawn as the convergence curves in Fig. 6. As evident from the Fig. 6, the proposed hybrid SCA–TLBO algorithm gives optimum solution for functions F1, F2, F3, F4 at the end of first iteration itself. The descending trend is quite evident in the convergence curves of hybrid SCA–TLBO on many of the test functions investigated. This strongly proves the ability of the hybrid SCA–TLBO algorithm in obtaining a better approximation of the global optimum over the course of generations.

Figure 7a, b shows a quantitative measure and averages the fitness of all search agents in each generation. If an algorithm improves its candidate solutions, apparently, the average of fitness should be enhanced over the course of generations. As the average fitness curves in Fig. 7a, b suggest, the hybrid SCA–TLBO algorithm shows degrading fitness on all of the test functions. Another fact worth mentioning here is the accelerated decrease in the average fitness curves, which shows that the improvement of the candidate solutions becomes faster and better over the course of generations.

5 Application of hybrid SCA–TLBO for visual tracking

The computer vision is the advancement of science and technology that include methods for acquiring, processing, analyzing and understanding images. As a scientific

Table 6 Minimization results of twenty-three benchmark functions with initial population size NP = 50

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Med	Worst	Med	Worst	Med	Worst	Med	Worst	Med	Worst
F_1	**0**	**0**	6.19E−33	2.16E−31	0.27489	3.3649	2.39E−42	3.37E−41	5.79E−220	1.72E−218
F_2	**0**	**0**	1.64E−22	1.64E−22	0.25312	0.9347	1.37E−24	3.64E−24	1.00E−109	3.41E−109
F_3	**0**	**0**	1.00E−15	3.25E−14	28.7138	115.7895	0.22399	0.42301	9.79E−98	1.58E−95
F_4	**0**	**0**	2.39E−11	1.42E−10	3.2528	6.6687	2.06E−07	4.40E−07	1.61E−92	1.05E−91
F_5	4.3904	4.4706	6.8806	7.2247	75.4953	211.1554	0.86751	4.7443	0.0019269	0.24962
F_6	**0**	**0**	0.25785	0.61762	0.74588	3.464	3.14E−22	2.35E−21	0.00E+00	3.08E−33
F_7	**0.00023601**	**0.00030175**	0.000527	0.001899	0.003339	0.020413	0.00277	0.004436	0.0027468	0.0057286
F_8	− 3597.6355	− 3479.196	− 2286.97	− 2127.11	− 3817.06	− 2363.86	4.44E−15	4.44E−15	− 3725.944	− 3360.7276
F_9	**0**	**0**	0.00E+00	0.002883	20.6204	47.7901	0	0	1.4924	5.9698
F_{10}	**4.44E−16**	**4.44E−16**	4.44E−15	7.99E−15	3.1716	4.0685	4.44E−15	4.44E−15	4.44E−15	4.44E−15
F_{11}	**0**	**0**	0.00E+00	4.17E−09	0.65952	0.99189	0	0	4.17E−12	0.024573
F_{12}	3.77E−08	8.85E−08	0.062491	0.088449	2.2663	4.1762	4.71E−32	4.71E−32	4.71E−32	5.10E−32
F_{13}	1.93E−07	4.52E−07	0.20012	0.31576	0.17384	1.0777	1.35E−32	1.35E−32	1.35E−32	1.35E−32
F_{14}	**0.998**	**0.998**	0.79809	1.9821	0.998	2.9821	0.998	0.998	0.998	0.998
F_{15}	**0.00030749**	**0.00010602**	0.000608	0.001535	0.000367	0.001597	0.00058	0.000738	0.00030749	0.00031155
F_{16}	− **1.0316**	− **1.0316**	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316
F_{17}	**0.39789**	**0.39789**	0.39827	0.39902	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
F_{18}	**3**	**3**	3	3	3	3	3	3	3	3
F_{19}	− **3.8628**	− **3.8628**	− 3.8547	− 3.8539	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628
F_{20}	− **3.322**	− **3.3178**	− 3.071	− 2.9989	− 3.322	− 3.2031	− 3.322	− 3.3217	− 3.2623	− 3.0779
F_{21}	− **10.1532**	− **10.1532**	− 4.5015	− 0.49653	− 5.1008	− 2.6829	− 10.1532	− 10.1532	− 10.1532	− 10.1532
F_{22}	− **10.4029**	− **10.4029**	− 4.9814	− 0.90776	− 3.9474	− 1.8376	− 10.4029	− 10.4029	− 10.4029	− 10.4029
F_{23}	− **10.5364**	− **10.5364**	− 4.8828	− 2.9115	− 5.1521	− 2.4273	− 10.5364	− 10.5364	− 10.5364	− 10.5364

Bold values represent best results

Table 7 Minimization results of twenty-three benchmark functions with initial population size NP = 100

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_1	**0**	**0**	6.29E−36	1.97E−35	1.39E−06	1.64E−06	5.31E−42	4.19E−42	1.11E−216	0
F_2	**0**	**0**	5.30E−24	1.49E−23	0.071544	0.10078	1.32E−24	5.44E−25	3.68E−108	6.24E−108
F_3	**0**	**0**	4.23E−15	1.18E−14	0.0049691	0.006396	0.20464	0.12047	2.58E−95	3.64E−95
F_4	**0**	**0**	7.73E−13	1.07E−12	0.37307	0.42685	1.85E−07	6.09E−08	4.80E−91	4.27E−91
F_5	4.7371	0.32147	6.9886	0.50031	14.3711	25.0218	1.3297	1.3999	0.0015253	0.0018532
F_6	**0**	**0**	0.19571	0.088459	1.75E−06	2.82E−06	0	0	0	0
F_7	**0.00016915**	**9.27E−05**	0.00064198	0.00060193	0.001719	0.00154	0.0020001	0.00081	0.0021902	1.10E−03
F_8	− 3652.1718	97.2607	− 2419.3478	126.0674	− 8276.4567	1257.597	− 4189.8289	9.59E−13	− 3777.2381	194.8172
F_9	**0**	**0**	0	0	18.2077	6.5505	0	0	0.39799	0.96122
F_{10}	**3.02E−16**	**0.00E+00**	4.09E−15	1.12E−15	2.0193	1.2307	4.44E−15	0	4.44E−15	0
F_{11}	**0**	**0**	2.20E−02	4.70E−02	0.34182	0.29467	0	0	0.0023027	0.0075556
F_{12}	1.82E−08	1.14E−08	0.048159	0.017097	0.93884	1.122	4.71E−32	1.15E−47	4.73E−32	4.08E−34
F_{13}	2.19E−07	1.09E−07	0.1552	0.056837	0.0069748	0.01267	1.35E−32	2.89E−48	1.35E−32	2.89E−48
F_{14}	**0.998**	**0**	1.1964	0.62743	0.998	0	0.998	0	0.998	0
F_{15}	0.00030785	1.29E−07	0.00078621	0.00038333	0.00035935	0.000164	0.00058149	9.67E−05	0.00038276	9.93E−06
F_{16}	− **1.0316**	**0**	− 1.0316	1.01E−05	− 1.0316	0	− 1.0316	0	− 1.0316	0
F_{17}	**0.39789**	**0**	0.39806	0.00013427	0.39789	0	0.39789	0	0.39789	0
F_{18}	**3**	**0**	3	2.59E−06	3	9.13E−16	3	9.59E−16	3	8.63E−16
F_{19}	− **3.8628**	**0**	− 3.8554	0.0022894	− 3.8628	9.36E−16	− 3.8628	9.36E−16	− 3.8628	9.36E−16
F_{20}	− **3.322**	4.68E−16	− 3.0638	0.056895	− 3.2744	0.061396	− 3.322	1.03E−04	− 3.322	5.34E−16
F_{21}	− **10.1532**	**0**	− 5.4003	0.90068	− 5.6441	3.2875	− 10.1532	1.57E−15	− 10.1532	0.00E+00
F_{22}	− **10.4029**	**1.18E−15**	− 4.7446	1.7447	− 8.0529	3.1088	− 10.4029	1.45E−15	− 10.4029	1.18E−15
F_{23}	− **10.5364**	**1.78E−15**	− 4.9297	1.642	− 6.7894	4.027	− 10.5364	2.05E−15	− 10.5364	1.87E−15

Bold values represent best results

Table 8 Minimization results of twenty− three benchmark functions with initial population size NP = 100

Function	SCA–TLBO		SCA		PSO		DE		TLBO	
	Med	Worst	Med	Worst	Med	Worst	Med	Worst	Med	Worst
F_1	**0**	**0**	7.69E−39	6.25E−35	6.18E−07	4.75E−06	3.48E−42	1.22E−41	6.07E−218	6.34E−216
F_2	**0**	**0**	5.13E−26	4.76E−23	0.022321	0.28804	1.24E−24	2.47E−24	7.74E−109	2.07E−107
F_3	**0**	**0**	2.90E−19	3.78E−14	0.0012475	0.017881	0.18584	0.47391	6.49E−96	1.05E−94
F_4	**0**	**0**	2.49E−13	3.35E−12	0.23111	1.4482	1.72E−07	2.78E−07	3.10E−91	1.44E−90
F_5	4.5069	5.2194	6.9215	8.0768	6.875	85.0444	0.68378	4.0039	0.001114	0.0062069
F_6	**0**	**0**	0.17619	0.36841	3.09E−07	7.95E−06	0	0	0	0
F_7	**0.00015145**	**0.00034356**	0.00035902	0.0019943	0.0011083	0.0051006	0.0018849	0.003812	0.0027468	0.0057286
F_8	− 3716.0728	− 3474.6295	− 2386.7896	− 2279.6889	− 8159.9714	− 6876.8634	− 4.19E+03	− 4.19E+03	− 3785.1632	− 3479.1989
F_9	**0**	**0**	0	0	18.4067	25.8689	0	0	0	2.9849
F_{10}	**4.44E−16**	**4.44E−16**	4.44E−15	4.44E−15	1.9815	4.0298	4.44E−15	4.44E−15	4.44E−15	4.44E−15
F_{11}	**0**	**0**	0.00E+00	1.26E−01	0.25097	1.115	0	0	4.17E−13	0.022573
F_{12}	1.52E−08	4.05E−08	0.051784	0.073414	0.58393	3.7998	4.71E−32	4.71E−32	4.71E−32	4.81E−32
F_{13}	6.36E−08	4.52E−07	0.15366	0.25887	0.0003646	0.040544	1.35E−32	1.35E−32	1.35E−32	1.35E−32
F_{14}	**0.998**	**0.998**	0.998	2.9821	0.998	0.998	0.998	0.998	0.998	0.998
F_{15}	**0.00030749**	**0.00010602**	0.00064609	0.0013554	0.00030749	0.00082614	0.0005938	0.0007215	0.00030749	0.00031065
F_{16}	− **1.0316**	− **1.0316**	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316	− 1.0316
F_{17}	**0.39789**	**0.39789**	0.39804	0.39836	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
F_{18}	**3**	**3**	3	3	3	3	3	3	3	3
F_{19}	− **3.8628**	− **3.8628**	− 3.8548	− 3.8541	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628	− 3.8628
F_{20}	− **3.322**	− **3.3178**	− 3.0622	− 3.0061	− 3.322	− 3.2031	− 3.322	− 3.322	− 3.322	− 3.322
F_{21}	− **10.1532**	− **10.1532**	− 4.9541	− 4.4954	− 5.1008	− 2.6305	− 10.1532	− 10.1532	− 10.1532	− 10.1532
F_{22}	− **10.4029**	− **10.4029**	− 5.0211	− 0.90959	− 10.4029	− 2.7659	− 10.4029	− 10.4029	− 10.4029	− 10.4029
F_{23}	− **10.5364**	− **10.5364**	− 5.0174	− 0.94423	− 7.8324	− 2.4217	− 10.5364	− 10.5364	− 10.5364	− 10.5364

Bold values represent best results

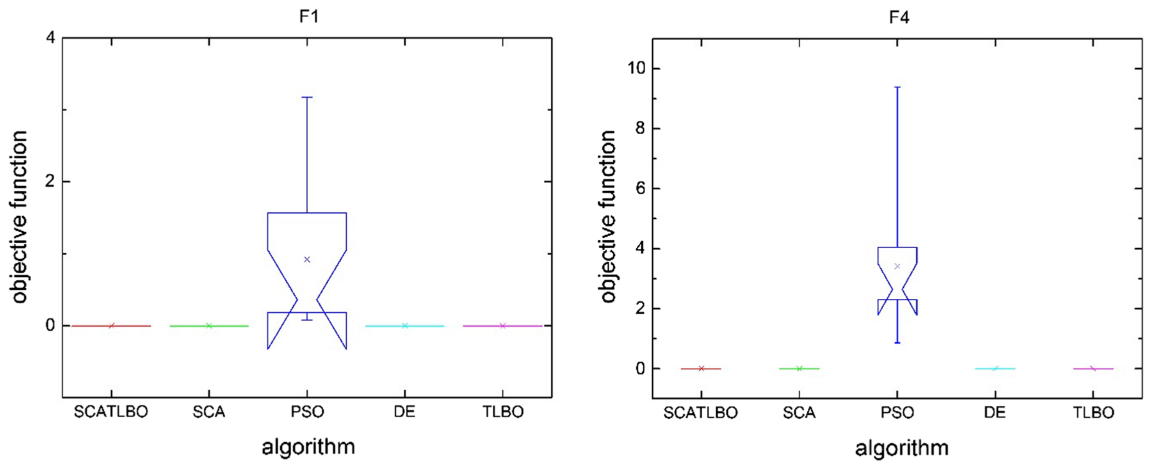


Fig. 5 ANOVA tests of the global minimum values, which are computed by using the SCA–TLBO, SCA, PSO, DE and TLBO for functions F1, F4, respectively

discipline, computer vision is fretful with the theory behind artificial systems that excerpt information from images. The image data can take various forms, such as video sequences, views from many cameras or multi-dimensional data from a medical scanner. As a scientific discipline, computer vision pursues to relate its theories and models to the design of computer vision systems. Many applications include: uses in security and surveillance, video communication and compression, navigation, display technology, high-level video analysis, traffic control, metrology, video editing, augmented reality and human–computer interfaces to medical imaging [44, 45]. Given the most important state (e.g., position and velocity) of a target object in the leading image, the objective of visual tracking is to estimate the states of the target in the subsequent frames. Even though visual tracking has been studied for more than a few decades and significant evolution has been made in recent years [46–51], it remains a challenging problem.

5.1 Optimization-based tracking system

Essentially, the tracking of an object in video sequences or the issue of tracing the target in every frame can be inferred as an optimization problem. The observation distance amidst the candidate and target forms a similarity function (fitness function). Tracing the target can be inferred as maximizing or minimizing the similarity function in the candidate solution. In this view, visual tracking, as an optimization problem, can be accomplished using optimization methods.

In order to compare the tracking achievement of optimization-based trackers, a general optimization-based tracking framework is designed to which other optimizers could also be applied. Tracking framework using the proposed hybrid SCA–TLBO algorithm is designed as shown in Fig. 8.

The target is preferred by the user or detected by some distinct object detectors in the first frame. Then, the state vector is initialized. The state vector in our work is defined as $X = [x, y, s]$, where x, y is the target position in pixel coordinates and s represents the scale parameter that restraints the size of the object. Then, after the target is selected in the first frame the state vector is initialized as $X_0 = [x_0, y_0, 1]$, where x_0, y_0 is the target's initial location and $s = 1$ specifies that there is no scale variation in the first frame.

After selection of target and initialization of state vector, a dynamic model produces new candidates' state vectors. Here, the random walk model is applied considering that there is very little movement of the object among frames. [Note: the vector can also be initialized using one of the motion models like constant velocity (CV) model or constant acceleration (CA) model, and in this instance, the state vector is a 5D problem]. In this work, a simple motion model is preferred to compare the novel method with the KF, α–β filter, EKF and BA. The correlation between the appearance and the state of the object is interpreted using an observation model. As known to all, a good observation model is essential to a tracker, but it is not easy to choose a certain observation model for all tracking scenarios. In this work, we are more concerned with the search performance, so we selected a standard kernel-based spatial color histogram [47] as the observation model.

Let $\{c_i\}_{i=1,\ldots,n}$ be the normalized pixel locations of the target candidate, centered at c in the present frame. $r = (h_x, h_y)$, where h_x and h_y, respectively, denote the width and the height of the target's rectangle. The kernel-based spatial color histogram is given by

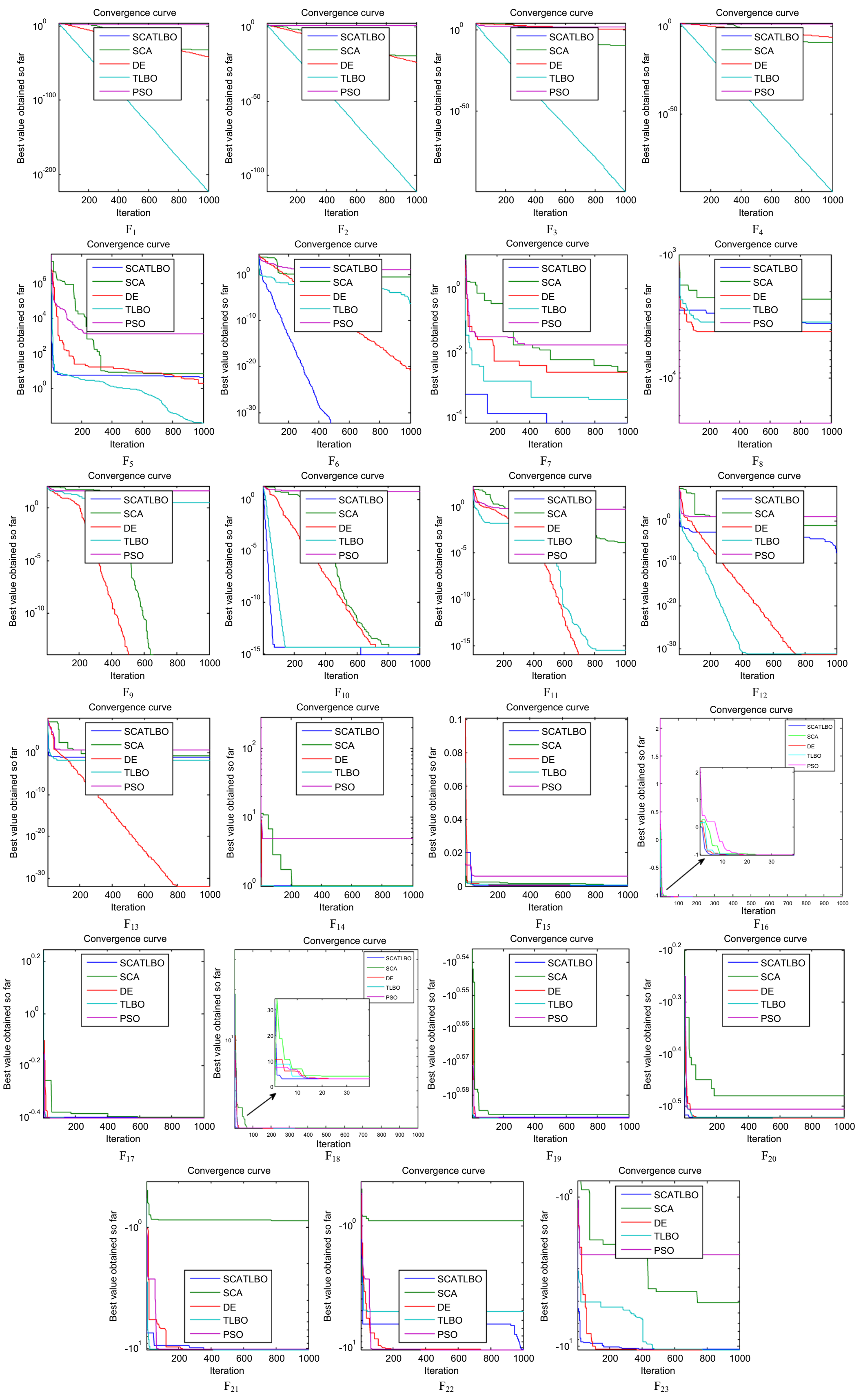


Fig. 6 Comparison between convergence curves of the SCA–TLBO, SCA, PSO, DE and TLBO algorithms for functions F_1–F_{23}, respectively

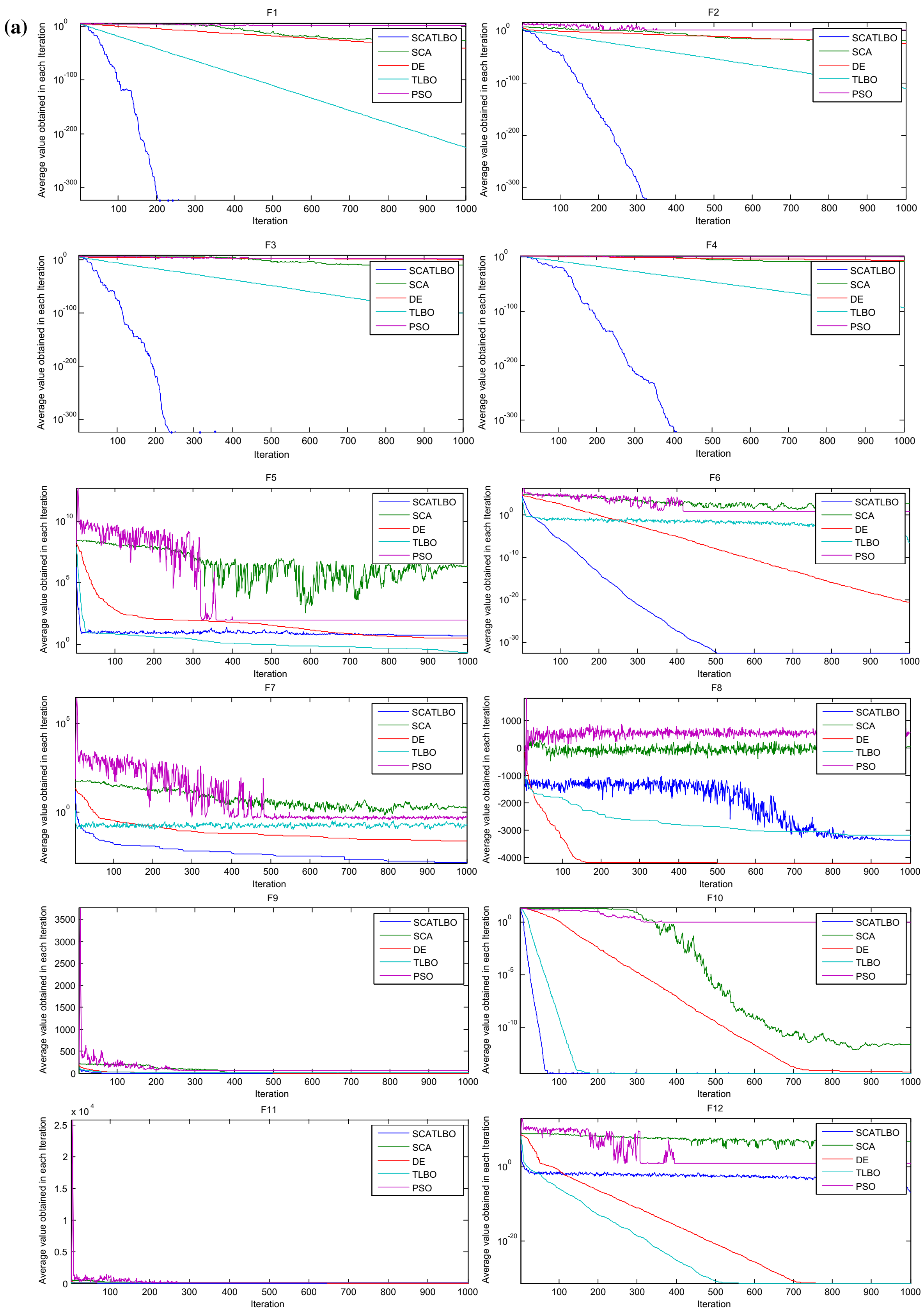


Fig. 7 Comparison of average fitness of search agents during optimization of the SCA–TLBO, SCA, PSO, DE and TLBO algorithms

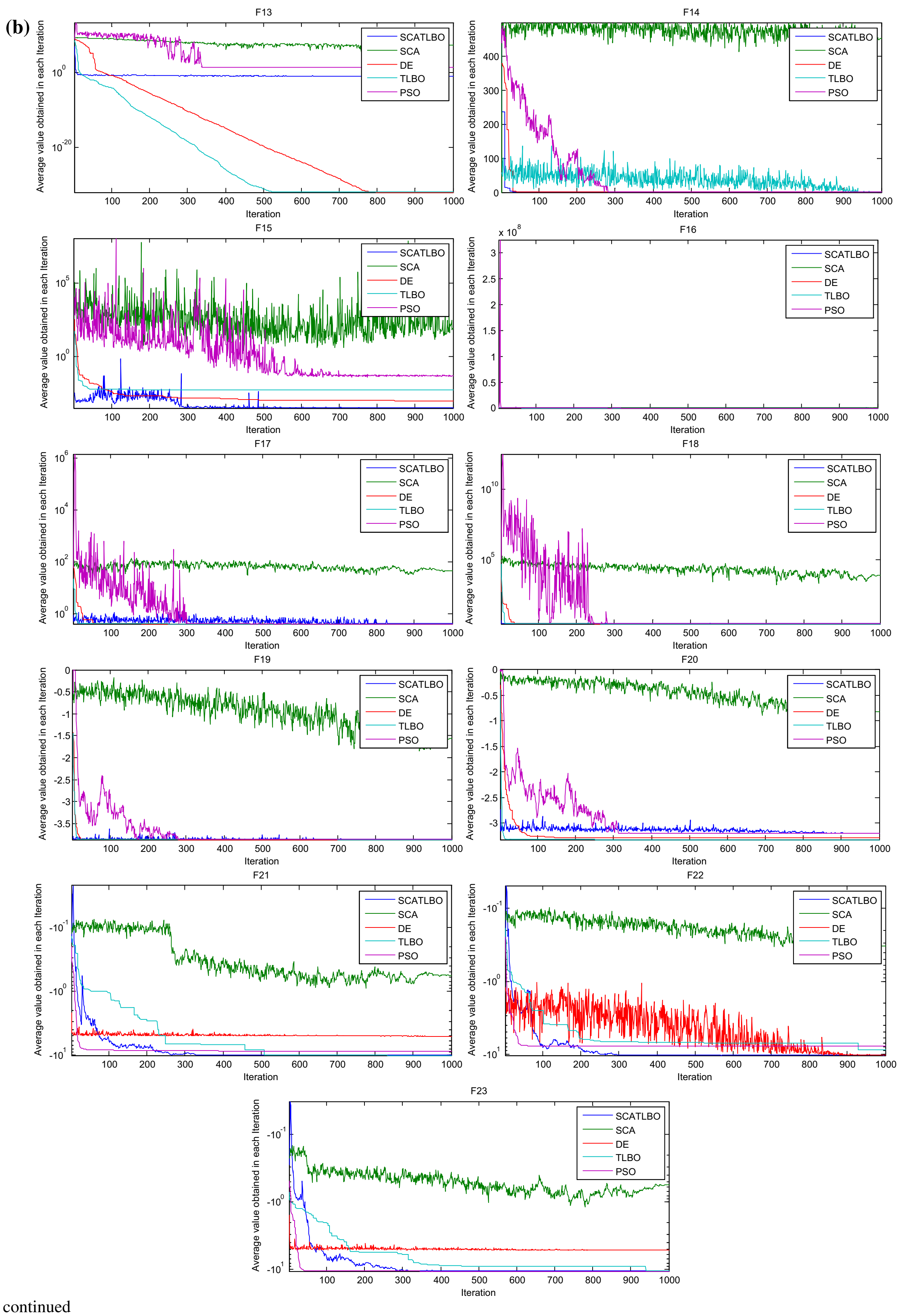


Fig. 7 continued

$$p_c^{(u)}(X_k) = C\sum_{i=1}^{M} k\left(\left\|\frac{c-c_i}{r}\right\|^2\right)\delta[b(c_i)-u],\quad u = 1,\ldots,m \tag{20}$$

where δ is the Kronecker delta function. The function $b{:}R^2 \rightarrow \{1, \ldots, m\}$ associates with the pixel at the location c_i and the index $b(c_i)$ of its bin in the quantized feature space. $k(x)$ is an isotropic kernel assigning a smaller weight to pixels farther from the center. $C = 1/\sum_{i=1}^{M} k\left(\left\|\frac{c-c_i}{r}\right\|^2\right)$ is the normalization constant.

When the state vector is described, a similarity (fitness) function is employed to calculate the observation distance between the candidate and target. Generally, the Bhattacharyya coefficient is employed to calculate the similarity between two histograms [47, 52]. It is given by,

$$B(h_1,h_2) = 1 - \sum_{i=1}^{N}\sqrt{h_1(i)\,h_2(i)} \tag{21}$$

where N is the number of bins in the histograms and h_1 and h_2 are the histograms being compared. The coefficient $B(h_1, h_2)$ is big when the histograms are alike and small when they are very unlike.

The dashed box in Fig. 8 represents the optimization process. This is the core part of the optimization-based visual tracking algorithm. In this part, an optimizer is adopted to select the candidate solution. This way can be carried out by maximizing or minimizing the similarity function. Each time the optimizer is queried for the target position, and the frame is displayed to indicate the position of the target. The entire loop extends until no other frame is available.

5.2 Experiments and discussion

5.2.1 Experimental setup details

In this work, we implemented our tracker in MATLAB R2013a software on PC machine with Intel i7-3770 CPU (3.4 GHz) with 2 GB memory, which runs 29 fps on this platform. The self-made video for the experiment consists of JPEG image sequence with 720 × 1280 pixels per frame resolution. The environment for the bad light condition is created by switching off all the lights in the room (improper illumination), while the environment is considered as a normal light condition when the room is properly illuminated. The distance considered for tracking the target object is approximately 6 m with target objects being balls of different sizes (small, medium and big) whose radii are 3.325, 4.9 and 8.5 cm, respectively.

5.2.2 Performance measures

5.2.2.1 Absolute error (AE) AE is the magnitude of the difference between the true value and the tracked value of the object.

$$\mathrm{AE} = |x_{\text{true}} - x_{\text{tracked}}| \tag{22}$$

where x_{true} is the true value of object parameters and x_{tracked} is the tracked value of the object parameters.

5.2.2.2 Root mean square error (RMSE) RMSE is one of the most widely used full-reference quality assessment metrics, which are computed by the square root of the average of squared intensity differences between tracked x_{tracked} and true image pixels x_{true}

$$\mathrm{RMSE} = \sqrt{\frac{1}{NM}\sum_{n=1}^{N}\sum_{m=1}^{M}(x_{\text{tracked}} - x_{\text{true}})^2} \tag{23}$$

where N and M are the image dimensions.

5.2.2.3 Tracking detection rate (TDR) Tracking detection rate is the ratio of a number of frames in which the object is detected to the total number of frames in which the object present

$$\mathrm{TDR} = \frac{\text{Object detected in number of frames}}{\text{Object present in number of frames}} \times 100. \tag{24}$$

5.2.2.4 Object tracking error (OTE) Object tracking error is the normal inconsistency in the centroid of the tracked object from its true value. It is given by,

$$\mathrm{OTE} = \frac{\sqrt{\sum_{i=1}^{N}(x_{\text{true}} - x_{\text{tracked}})^2 - (y_{\text{true}} - y_{\text{tracked}})^2}}{N} \tag{25}$$

where x_{true} and y_{true} are the actual 2D coordinates of the object, and x_{tracked} and y_{tracked} are the tracked 2D coordinates of the object.

The proposed hybrid SCA–TLBO-based tracking framework is used to experimentally measure object tracking error, absolute error, tracking detection rate, root mean square error and time cost as parameters for hybrid SCA–TLBO. To reveal the capability of tracker proposed in this work, a comparison of hybrid SCA–TLBO-based tracker and three probability-based trackers, viz. alpha–beta (α–β) filter [53], linear Kalman filter (LKF) [54], extended Kalman filter (EKF) [54] and bat algorithm (BA) [52], is presented. The α–β, LKF, EKF and BA have several parameters that should be initialized before tracking. Table 9 shows the initial values of basic parameters for these algorithms.

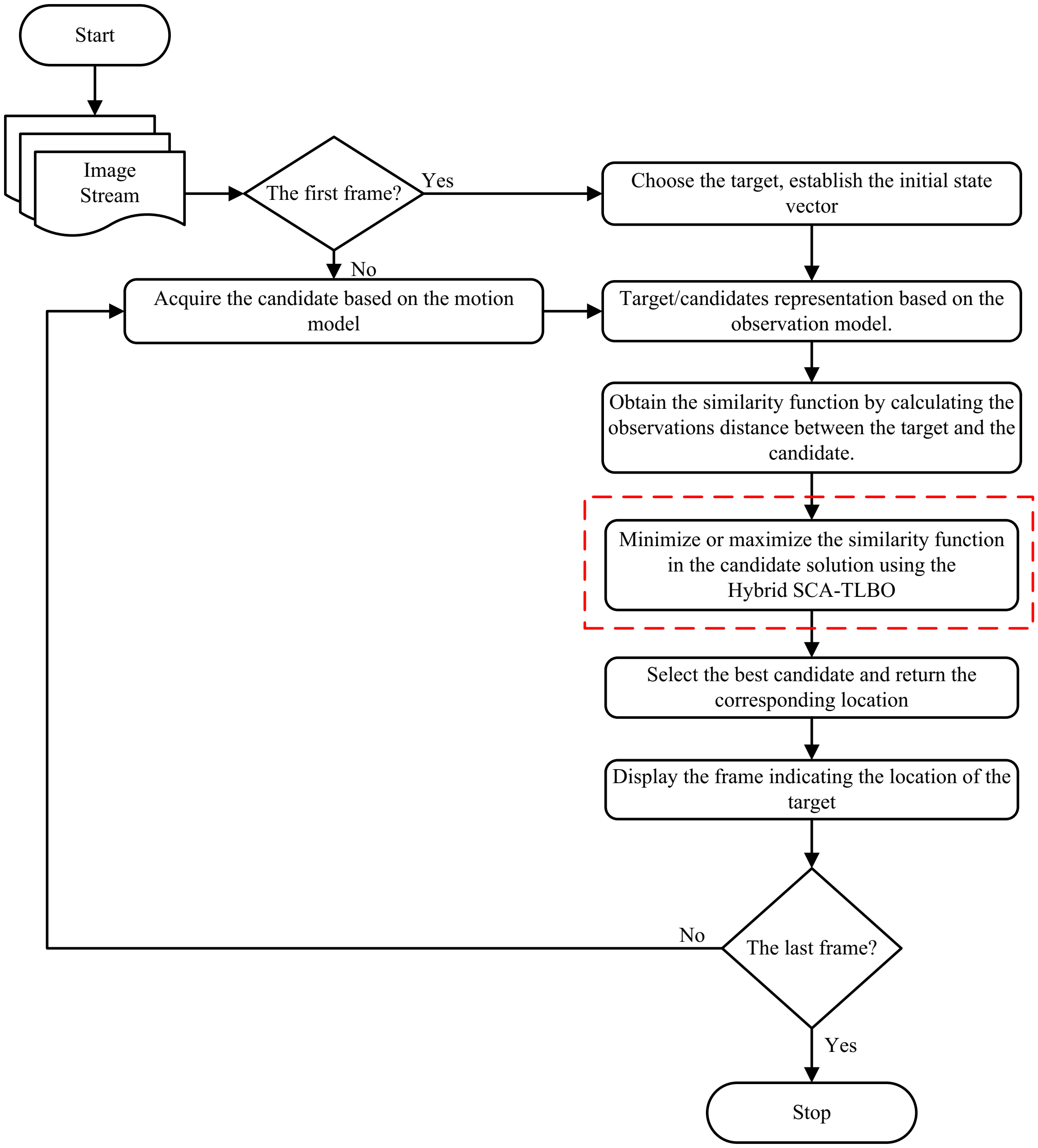

Start
Image Stream
The first frame?
Yes
No
Choose the target, establish the initial state vector
Acquire the candidate based on the motion model
Target/candidates representation based on the observation model.
Obtain the similarity function by calculating the observations distance between the target and the candidate.
Minimize or maximize the similarity function in the candidate solution using the Hybrid SCA-TLBO
Select the best candidate and return the corresponding location
Display the frame indicating the location of the target
No
The last frame?
Yes
Stop

Fig. 8 Proposed hybrid SCA–TLBO-based tracking framework

The RMSE, AE and OTEs of the hybrid SCA–TLBO, α–β filter, LKF, EKF and BA for different-size balls' dataset are given in Table 10 under normal light condition. As observed in Table 10, under the normal light condition, the average value of the RMSE was reduced after applying the hybrid SCA–TLBO to small-, medium- and big-size balls' data. On average, the minimum RMSE reduction was 0.01, 0.01 and 0.02 for small-, medium- and big-size balls' data, respectively, under normal light conditions for the hybrid SCA–TLBO algorithm. The average value of AE was reduced after applying the hybrid SCA–TLBO to small-, medium- and big-size balls' data. On average, the minimum AE was 0.06, 0.10 and 0.14 for small-, medium- and big-size balls' data, respectively, under normal light conditions for the hybrid SCA–TLBO algorithm. Also, the average value of the OTE was reduced after applying the hybrid SCA–TLBO to small-, medium- and big-size balls' data. On average, the minimum OTE was 0.01, 0.02 and 0.04 for small-, medium- and big-size balls' data, respectively, under normal light conditions for the hybrid SCA–TLBO algorithm.

The RMSE, AE and OTEs of the hybrid SCA–TLBO, α–β filter, LKF, EKF and BA for only small-size balls' dataset are given in Table 10 under bad light condition. As

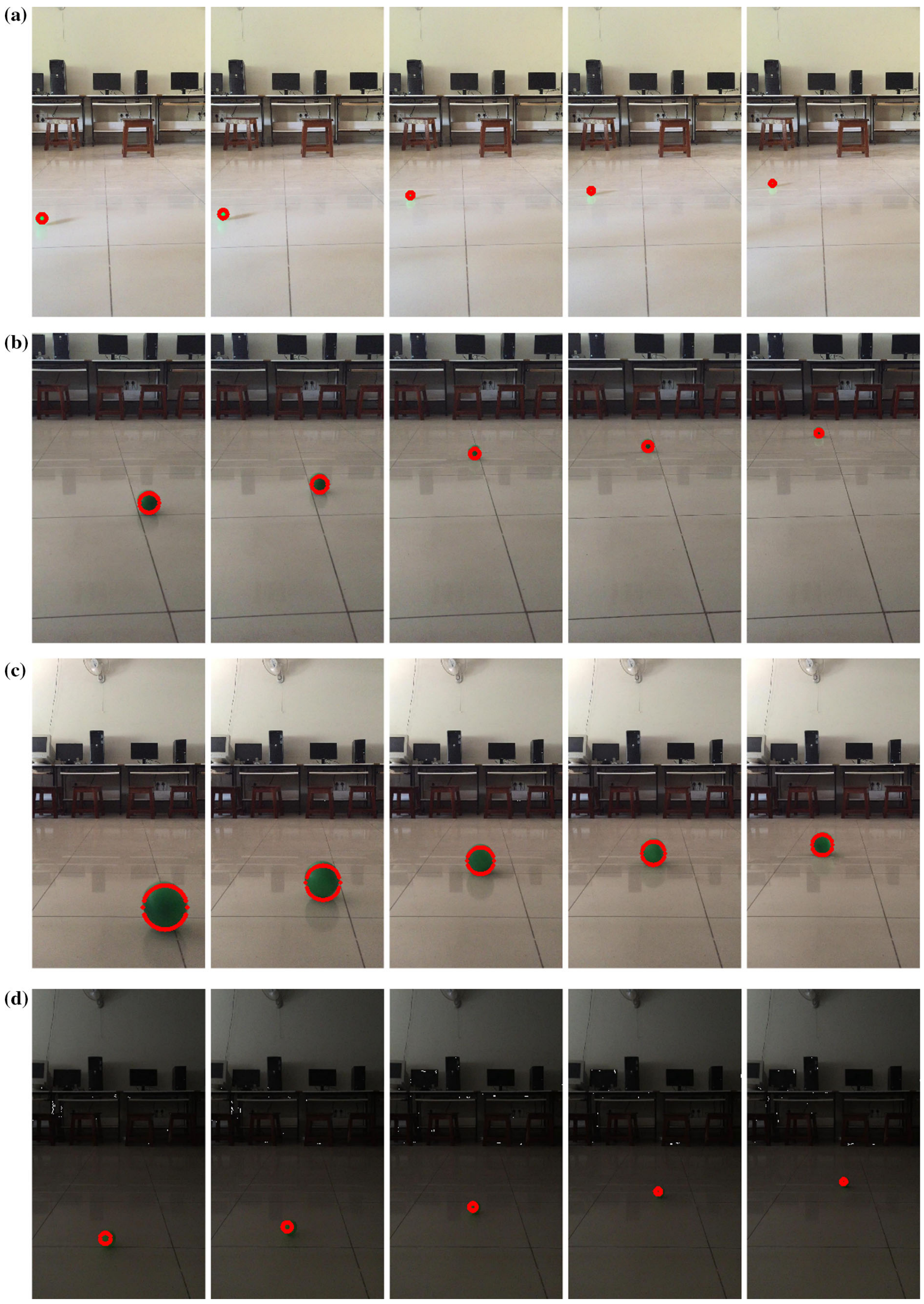
Table 9 Simulation parameters for trackers

α–β		LKF		EKF		BA	
Parameters	Values	Parameters	Values	Parameters	Values	Parameters	Values
α	0.9	*dt*	1	*dt*	1	Number of artificial bats	30
β	0.005	*A*	$\begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	*A*	$\begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$f_{\min}$	− 20
Δ_k	1	*H*	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$	*H*	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$	$f_{\max}$	20
–	–	*R*	$\begin{bmatrix} 0.1524 & 0.0143 \\ 0.055 & 0.0055 \end{bmatrix}$	*R*	$\begin{bmatrix} 0.2845 & 0.0045 \\ 0.0045 & 0.0045 \end{bmatrix}$	*A*	0.25
–	–	*Q*	0.01 × eye (4)	*Q*	0.01 × eye (4)	*r*	0.5
–	–	*P*	100 × eye (4)	*P*	100 × eye (4)	*β*	ε [0,1]
						ε	ε [− 1,1]

Table 10 Average parameters comparison for different sizes of balls using five trackers under normal and bad light conditions

Parameters	In Normal light condition															In bad light condition				
	For small-size ball					For medium-size ball					For big-size ball					For small-size ball				
	SCA–TLBO	α–β	LKF	EKF	BA	SCA–TLBO	α–β	LKF	EKF	BA	SCA–TLBO	α–β	LKF	EKF	BA	SCA–TLBO	α–β	LKF	EKF	BA
AE	**0.06**	0.98	0.12	0.76	0.12	**0.10**	0.74	0.21	1.25	0.17	**0.14**	1.31	0.63	1.34	0.17	**0.08**	0.45	0.11	1.05	0.11
RMSE	**0.01**	0.44	0.03	0.17	0.02	**0.01**	0.28	0.04	0.28	0.04	**0.02**	0.45	0.21	0.30	0.14	**0.01**	0.09	0.02	0.17	0.02
TDR (%)	100	96.2	100	100	100	100	91.6	100	91.6	100	100	93.1	100	93.1	100	100	85.7	100	85.7	100
OTE	**0.01**	1.35	0.04	0.16	0.03	**0.02**	0.69	0.05	0.29	0.03	**0.04**	1.01	0.48	0.29	0.20	**0.01**	0.22	0.02	0.14	0.02

Bold values represent best results


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

◀**Fig. 9** **a** Tracking result of hybrid SCA–TLBO for the small-size ball (6, 10, 21, 25 and 31 frames) under normal light condition. **b** Tracking result of hybrid SCA–TLBO for the medium-size ball (11, 15, 31, 36 and 52 frames) under normal light condition. **c** Tracking result of hybrid SCA–TLBO for the big-size ball (16, 25, 39, 44 and 55 frames) under normal light condition. **d** Tracking result of hybrid SCA–TLBO for the small-size ball (24, 28, 36, 40 and 52 frames) under bad light condition

observed in Table 10, under the bad light condition, the average value of the AE, RMSE and OTE was reduced after applying the hybrid SCA–TLBO to small ball data. On average, the minimum AE, RMSE and OTE were 0.08, 0.01 and 0.01 for small ball data, respectively, under the bad light condition for the hybrid SCA–TLBO algorithm. The best average values are emphasized in Table 10.

Table 11 Average time cost of the five trackers for different sizes of balls under normal and bad light conditions

	Image frames	Time cost (ms)				BA
		SCA–TLBO	α–β	LKF	EKF	
Normal light condition	Small-size ball	**29.3**	48.9	32.9	48.1	32.5
	Medium-size ball	**25.2**	27.0	29.7	29.1	29.3
Bad light condition	Big-size ball	**24.7**	30.0	28.7	28.4	27.8
	Small-size ball	**21.0**	25.6	24.9	25.7	23.6

Bold values represent best results

Table 12 Description of the tracking examples

Video	Frame numbers	Challenging factors
David	252	Occlusion (OCC), background clutters (BC), out-of-plane rotation (OPR), deformation (DEF)
Panda	1000	Scale variation (SV), occlusion (OCC), deformation (DEF), in-plane rotation (IPR), out-of-plane rotation (OPR), out of view (OV), low resolution (LR)
Subway	175	Occlusion (OCC), deformation (DEF), background clutters (BC)
Mountain Bike	228	In-plane rotation (IPR), background clutters (BC), out-of-plane rotation (OPR)
Boy	602	Scale variation (SV), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR)
Crossing	120	Scale variation (SV), background clutters (BC), deformation (DEF), fast motion (FM), out-of-plane rotation (OPR)
Car	1500	Illumination variation (IV), scale variation (SV), background clutters (BC)
Couple	140	Scale variation (SV), deformation (DEF), fast motion (FM), out-of-plane rotation (OPR), background clutters (BC)

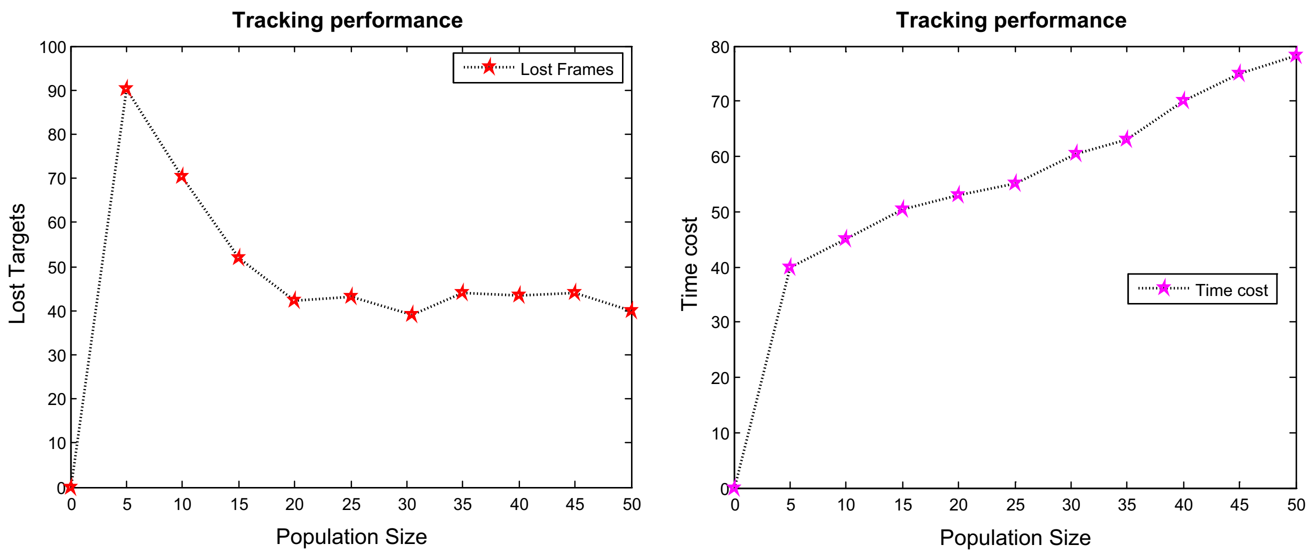


Fig. 10 Comparison of tracking performance with different population size NP

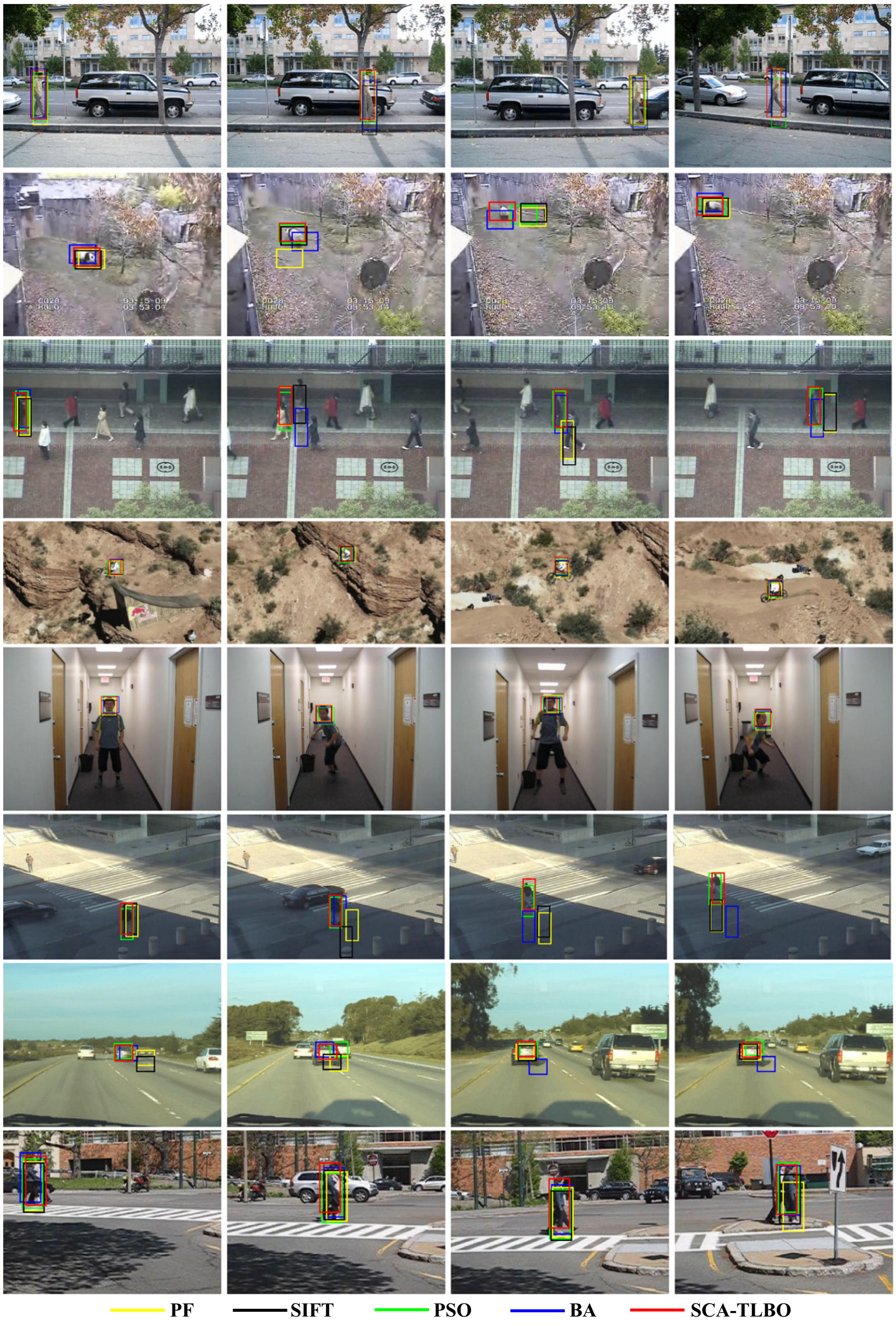


Fig. 11 Tracking result of PF, SIFT, PSO, BA and hybrid SCA–TLBO trackers for David, Panda, Subway, Mountain Bike, Boy, Crossing, Car and Couple Video sequences (from top to bottom)

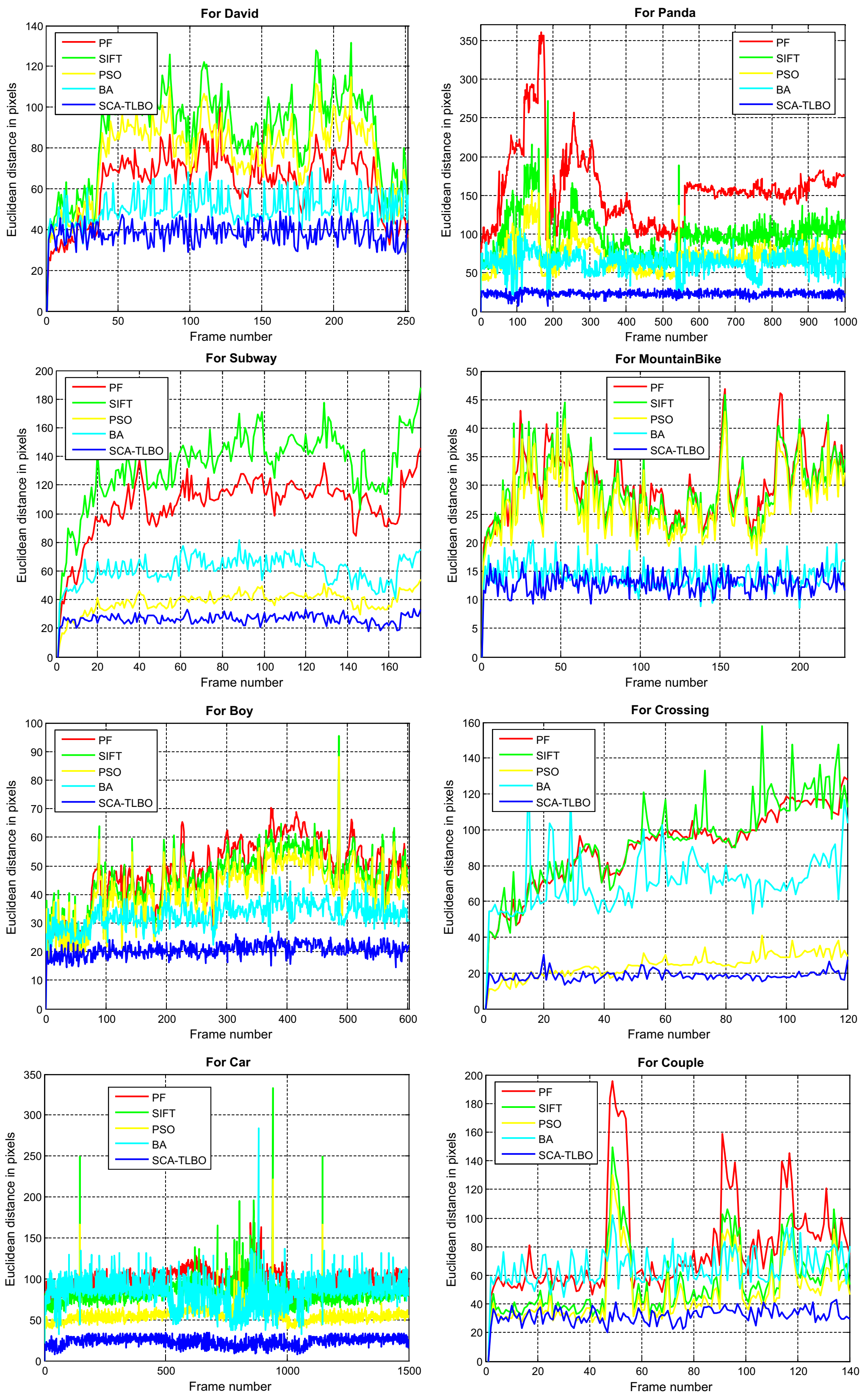


Fig. 12 Tracking accuracy comparisons of different trackers for David, Panda, Subway, Mountain Bike, Boy, Crossing, Car and Couple Video sequences

Figure 9a–c shows five frames out of the tracking results for small-, medium- and big-size balls' data under the normal light condition, and Fig. 9d shows under the bad light condition of the proposed hybrid SCA–TLBO algorithm. The proposed approach was evaluated using a self-made database, with the frame size being 720 × 1280. Frames shown in the figure were selected from the video while tracking the object continuously during its movement. The target object is marked by the bounding box (red circle). For normal light condition, in frames 25 and 31 of Fig. 9a, the target object tracked using hybrid SCA–TLBO is dominated by the red bounding box as the camera is static and the size of the object is reduced with distance. Even though the target object is small, the proposed tracking algorithm can detect the object (small ball) nevertheless.

For bad light condition, in frames 40 and 52 of Fig. 9d, the target object tracked using hybrid SCA–TLBO is dominated by the red bounding box as the camera is static and the size of the object is reduced with distance. Even though the target object is small, the proposed tracking algorithm can detect the object (small ball) under bad light condition.

To analyze the time complexity, the average time costs of the five trackers in the tracking process are calculated, and the comparative results are shown in Table 11. Table 11 shows that in all tracking examples (e.g., small-, medium- and big-size balls' data), the average time cost of hybrid SCA–TLBO is less than alpha–beta (α–β) filter, linear Kalman Filter, extended Kalman filter and bat algorithm.

To illustrate the efficiency of hybrid SCA–TLBO more clearly, the proposed algorithm is also compared with four other tracking algorithms including particle filter (PF) [55], scale-invariant feature transform (SIFT) [56], particle swarm optimization (PSO) [57] and bat algorithm (BA) [52]. To carry on the comparison, some videos [58] which cover all challenging factors are selected and used for evaluation. (Note: the tracking examples are available on the Web site http://visual-tracking.net.) The targets in these cases are suffered from various challenging factors as depicted in Table 12.

In hybrid SCA–TLBO, we used different population sizes (NP) from 5 to 50 and found that it is sufficient to use 15–20 population sizes for most tracking problems. The performance is evaluated using two measures, some of lost targets, i.e., the number of frames where the overlap region between its bounding box and the ground truth is less than 50%, and time complexity, i.e., the average time costs (ms). Figure 10 shows the performance comparison using different values of NP. Figure 10 shows that there is a general trend that when NP increases, the number of lost targets decreases and the time cost increases. This means that the increasing population size enhances the tracking accuracy and makes the tracker more time-consuming. Therefore, we have used the fixed population size of NP = 20 in all our experiments. Like in [47, 52], the optimization process was ended by using three termination conditions as follows:

- The fitness of f_{worst} (the worst solution) is good enough.
- The f_{best} (best solution) is also good enough and the Euclidean distance between the best and the lowest solution (d) is below a certain threshold.
- A maximum number of iterations or generations reached the algorithm ends and is set as 100 in this work.

It is worth mentioning that the values used in the first two conditions rely heavily on the fitness function, and in this case, it is the Bhattacharyya coefficient as mentioned in [47, 52]. Experimental results showed a general tendency that a region with Bhattacharyya coefficient being higher than 0.6 could mainly represent the target. Therefore, to be on the safe side, in the first termination condition, the value of (f_{worst}) is set as 0.6 to guarantee that all the potential solutions are close to the actual target. To ensure all candidates close together with the best candidate near the target, we set (f_{best}) as 0.8 and the distance d is 5. Same target model and motion model are used for fair comparison.

Tracking result of hybrid SCA–TLBO for David, Panda, Subway, Mountain Bike, Boy, Crossing, Car and Couple Video sequences are shown in Fig. 11, and experimental results depict that the hybrid SCA–TLBO-based tracker is capable of tracking a random target in various challenging situations. For evaluating the accuracy of trackers, by identifying the center of the tracked target in each frame visually, the videos have been manually labeled. Then, the Euclidean distance between the center estimated by the trackers and the actual center is calculated and is used as accuracy measure. The results of the comparison are shown in Fig. 12. Figure 12 shows that the hybrid SCA–TLBO-based tracker performs better in all challenging conditions. It is capable of successfully tracking the target during the whole tracking process.

6 Conclusion

In this paper, a novel optimization algorithm called hybrid SCA–TLBO is proposed, for solving optimization problems and visual tracking. This algorithm is tested using twenty-three eminent test functions. The experimental results show that the performance of the proposed algorithm is superior to that of the other existing algorithms in exploiting the optimum and it also has advantages in exploration. The performance measures show that the hybrid SCA–TLBO algorithm possesses a better capability to track an object as compared to other trackers.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Sullivan KA, Jacobson SH (2001) A convergence analysis of generalized hill climbing algorithms. IEEE Trans Autom Control 46(8):1288–1293. https://doi.org/10.1109/9.940936
2. Paravati G, Sanna A, Pralio B, Lamberti F (2009) A genetic algorithm for target tracking in FLIR video sequences using intensity variation function. IEEE Trans Instrum Meas 58(10):3457–3467. https://doi.org/10.1109/TIM.2009.2017665
3. Fonseca CM, Fleming PJ (1995) An overview of evolutionary algorithms in multi objective optimization. Evol Comput 3:1–16. https://doi.org/10.1162/evco.1995.3.1.1
4. Biswas A, Mishra KK, Tiwari S, Misra AK (2013) Physics-inspired optimization algorithms: a survey. J Optim. https://doi.org/10.1155/2013/438152
5. Parpinelli RS, Lopes HS (2011) New inspirations in swarm intelligence: a survey. Int J Bioinspired Comput 3:1–16. https://doi.org/10.1504/IJBIC.2011.038700
6. Li R et al (2013) Mixed integer evolution strategies for parameter optimization. Evol Comput 21(1):29–64. https://doi.org/10.1162/EVCO_a_00059
7. Chakraborty G (1999) Genetic programming for a class of constrained optimization problems. In: 1999 IEEE international conference on systems, man, and cybernetics, 1999. IEEE SMC'99 Conference Proceedings, vol 1, Tokyo, pp 314–319. https://doi.org/10.1109/icsmc.1999.814109
8. Dasgupta D, Zbigniew M (2013) Evolutionary algorithms in engineering applications. Springer Science & Business Media, Berlin. https://doi.org/10.1007/978-3-662-03423-1
9. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713. https://doi.org/10.1109/TEVC.2008.919004
10. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–224813. https://doi.org/10.1016/j.ins.2009.03.004
11. Rutenbar RA (1989) Simulated annealing algorithms: an overview. IEEE Circuits Devices Mag 5(1):19–26. https://doi.org/10.1109/101.17235
12. Kumar Singh H, Isaacs A, Ray T, Smith W (2008) A simulated annealing algorithm for constrained multi-objective optimization. In: 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence), Hong Kong, pp 1655–1662. https://doi.org/10.1109/cec.2008.4631013
13. Du H, Wu X, Zhuang J (2006) Small-world optimization algorithm for function optimization. Adv Nat Comput. https://doi.org/10.1007/11881223_33
14. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112–113:283–294
15. Formato RA (2007) Central force optimization: a new metaheuristic with applications in applied electromagnetics. Prog Electromagn Res 77:425–491. https://doi.org/10.2528/PIER07082403
16. Alatas B (2011) ACROA: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 38(10):13170–13180. https://doi.org/10.1016/j.eswa.2011.04.126
17. Shah-Hosseini H (2011) Principal components analysis by the galaxy based search algorithm: a novel metaheuristic for continuous optimisation. Int J Comput Sci Eng 6:132–140. https://doi.org/10.1504/IJCSE.2011.041221
18. Kaveh A, Mahdavi VR (2014) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27
19. Kanagaraj G, Ponnambalam SG, Loo CK (2015) Charged system search algorithm for robotic drill path optimization. In: 2015 international conference on advanced mechatronic systems (ICAMechS), Beijing, pp 125–130. https://doi.org/10.1109/icamechs.2015.7287141
20. Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the 6th international symposium on micro machine and human science, pp 39–43. https://doi.org/10.1109/mhs.1995.494215
21. Dorigo M, Birattari M (2010) Ant colony optimization. Encyclopedia of machine learning. Springer, Berlin, pp 36–39
22. Li X, Zhang J, Yin M (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behaviour. Neural Comput Appl 24:1867–1877. https://doi.org/10.1007/s00521-013-1433-8
23. Geem ZW, Kim JH, Loganathan G (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
24. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching learning based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315. https://doi.org/10.1016/j.cad.2010.12.015
25. Rao RV, Savsani VJ, Vakharia DP (2012) Teaching learning based optimization: an optimization method for continuous nonlinear large scale problems. Inf Sci 183(1):1–15. https://doi.org/10.1016/j.ins.2011.08.006
26. He S, Wu QH, Saunders JR (2009) Group search optimizer: an optimization algorithm inspired by animal searching behavior. IEEE Trans Evol Comput 13(5):973–990. https://doi.org/10.1109/TEVC.2009.2011992
27. Fanni A, Manunza A, Marchesi M, Pilo F (1998) Tabu search metaheuristics for global optimization of electromagnetic problems. IEEE Trans Magn 34(5):2960–2963. https://doi.org/10.1109/20.717691
28. Hosseini SM, Al Khaled A (2014) A survey on the Imperialist Competitive Algorithm metaheuristic: implementation in engineering domain and directions for future research. Appl Soft Comput 24:1078–1094
29. Eita MA, Fahmy MM (2014) Group counseling optimization. Appl Soft Comput 22:585–604
30. Kashan AH (2011) An efficient algorithm for constrained global optimization and application to mechanical engineering design: league championship algorithm (LCA). Comput Aided Des 43(12):1769–1792. https://doi.org/10.1016/j.cad.2011.07.003
31. Moosavian N, Roodsari BK (2014) Soccer league competition algorithm: a novel meta-heuristic algorithm for optimal design of water distribution networks. Swarm Evol Comput 17:14–24
32. Ghorbani N, Babaei E (2016) Exchange market algorithm for economic load dispatch. Int J Electr Power Energy Syst 75:19–27
33. Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput 13(5):2592–2612
34. Ramezani F, Lotfi S (2013) Social-based algorithm (SBA). Appl Soft Comput 13(5):2837–2856
35. Tan Y, Zhu Y (2010) Fireworks algorithm for optimization. Adv Swarm Intell. https://doi.org/10.1007/978-3-642-13495-1_44

36. Dai C, Zhu Y, Chen W (2007) Seeker optimization algorithm. Comput Intell Sec. https://doi.org/10.1007/978-3-540-74377-4_18
37. Blum C, Roli A (2008) Hybrid meta-heuristics: an introduction. Hybrid Metaheuristics. https://doi.org/10.1007/978-3-540-78295-7_1
38. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
39. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
40. Crepinsek M, Liu SH, Mernik M (2013) Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput Surv. https://doi.org/10.1145/2480741.2480752
41. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102. https://doi.org/10.1109/4235.771163
42. Jamil M, Yang XS (2013) A literature survey of benchmark functions for global optimisation problems. Int J Math Model Numer Optim 4(2):150–194. https://doi.org/10.1504/IJMMNO.2013.055204
43. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18. https://doi.org/10.1016/j.swevo.2011.02.002
44. Yang H, Shao L, Zheng F, Wang L, Song Z (2011) Recent advances and trends in visual tracking: a review. Neurocomputing 74:3823–3831. https://doi.org/10.1016/j.neucom.2011.07.024
45. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. ACM Comput Surv 38(4):1–45. https://doi.org/10.1145/1177352.1177355
46. Sokhandan A, Monadjemi A (2016) A novel biologically inspired computational framework for visual tracking task. Biol Inspired Cogn Archit 18:68–79
47. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. IEEE Trans Pattern Anal Mach Intell 25(5):564–577. https://doi.org/10.1109/TPAMI.2003.1195991
48. Hare S et al (2016) Struck: structured output tracking with kernels. IEEE Trans Pattern Anal Mach Intell 38(10):2096–2109. https://doi.org/10.1109/TPAMI.2015.2509974
49. Yi S, Jiang N, Feng B, Wang X, Liu W (2016) Online similarity learning for visual tracking. Inf Sci 364–365:33–50
50. Chen W, Zhang K, Liu Q (2016) Robust visual tracking via patch based kernel correlation filters with adaptive multiple feature ensemble. Neurocomputing 214:607–617
51. Gao M-L, Yin L-J, Zou G-F, Li H-T, Liu W (2015) Visual tracking method based on cuckoo search algorithm. Opt Eng 54(7):073105
52. Gao M-L, Shen J, Yin L-J, Liu W, Zou G-F, Li H-T, Gui-Xia Fu (2016) A novel visual tracking method using bat algorithm. Neurocomputing 177:612–619
53. Crouse DF (2015) A general solution to optimal fixed-gain (α–β–γ etc) filters. IEEE Signal Process Lett 22(7):901–904
54. Simon D (2010) Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. IET Control Theory Appl 4(8):1303–1318
55. Khan ZH, Gu IYH, Backhouse AG (2011) Robust visual object tracking using multi-mode anisotropic mean shift and particle filters. IEEE Trans Circuits Syst Video Technol 21(1):74–87
56. Zhou H, Yuan Y, Shi C (2009) Object tracking using SIFT features and mean shift. Comput Vis Image Underst 113:345–352
57. Thida M, Eng H-L, Monekosso DN, Remagnino P (2013) A particle swarm optimisation algorithm with interactive swarms for tracking multiple targets. Appl Soft Comput 13:3106–3117
58. Wu Y, Lim JW, Yang MH (2015) Object tracking benchmark. IEEE Trans Pattern Anal 37(9):1834–1848