Comparative Study of Different Metaheuristics for the Trajectory Planning of a Robotic Arm

Poonam Savsani, R. L. Jhala, and V. J. Savsani

***Abstract*—In this paper, seven different metaheuristic optimization algorithms developed between 2005 and 2012 are applied to optimize the robot trajectory for a three-revolute (3R) robotic arm. These algorithms include the artificial bee colony (ABC) algorithm, biogeography-based optimization, gravitational search algorithm, cuckoo search algorithm (CS), firefly algorithm, bat algorithm, and teaching–learning-based optimization (TLBO). This work presents the optimization problem with the objective to plan a trajectory, which can minimize joint travelling time, joint travelling distance, and total joint Cartesian lengths, simultaneously. Nine different design variables are considered for the intermediate joint angles, joint velocities, the end-effector angle for the final configuration, the time from initial to intermediate configuration, and the time from intermediate to final configuration. Two different experiments are conducted to investigate the effect of different considered metaheuristics: 1) for the free workspace and 2) for the workspace with obstacle. All these algorithms are compared based on the algorithm ability to find the best solutions, mean solutions, standard deviation, computational effort, and the convergence of the solutions. Statistical investigation is also performed by using the Friedman rank test and the Holm–Sidak multiple comparison *t*-test to check the significance of the algorithm for its suitability to the robot trajectory optimization problems. The results show the significance of TLBO, ABC, and CS for the robot trajectory optimization problems.**

***Index Terms*—Artificial bee colony (ABC) algorithm, bat algorithm (BA), biogeography-based optimization (BBO), cuckoo search algorithm (CS), firefly algorithm (FA), gravitational search algorithm (GSA), obstacle existence, optimization, robot trajectory, teaching–learning-based optimization (TLBO).**

I. Introduction

ROBOT arms are effectively used in modern production systems, and in order to achieve high-speed operation, optimal trajectory planning is extremely important. Robot motion planning (RMP) has always been an interesting research topic since recent years [1]. Thus, many studies on optimal trajectory planning have been reported in free workspace, as well as in workspace with obstacle. The research started with the application of traditional optimization techniques to such problems [2]–[4]. Traditional optimization techniques have many disadvantages, such as only finding local solutions, requiring gradients, failing for discontinuous functions, etc. Thus, a need arises to apply heuristic and metaheuristic methods, such as simulated annealing (SA), genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization (ACO), etc., to a robot trajectory problem. The first application of SA to RMP was reported in [5], and potential field concept along with SA was presented in [6] and [7]. GA is one of the powerful optimization techniques, and a lot of research is reported for the application of GA to many engineering problems. For the motion planning, GA is used for mobile robot path planning, collision-free path planning, planning multipaths, and dynamic constraint multipath routing [8]–[13]. Application of ACO for the motion planning of a robot can be found for mobile and articulated robots and for multirobot systems [14]–[16]. The other effective and widely used optimization technique is PSO, and it has been also applied to many obstacle avoidance motions planning, which are reported in [17]–[19], for multiobjective optimization, path planning for soccer robots, and smooth path planning for mobile robots. ABC is also reported for the multirobot path planning from the starting to the goal position [20]. Free-space robot trajectory optimization for a three-revolute (3R) planar robotic arm was done by using ABC and teaching–learning-based optimization (TLBO), in which the performance of TLBO was better than ABC [21]. The aforementioned methods also have some disadvantages. The performance of GA depends on parameters such as string length, crossover probability, and mutation rate. GAs have the tendency to get stuck at the local optima and also have poor convergence rate. PSO and ACO also require many parameters, such as cognitive factor, social factor, and inertia weight for PSO and pheromone deposition rate and evaporation rate for ACO, for its effective working. Such parameters are required to be changed with different problems for the effective working of the algorithm. Thus, a need arises to investigate different advance optimization techniques for RMP. In the literature so far, it has been observed that techniques similar to GA, PSO, ABC, ACO, and TLBO are the metaheuristics investigated for the RMP problems, for a robotic arm, so far. There are many other metaheuristics such as biogeography-based optimization (BBO), gravitational search algorithm (GSA), cuckoo search algorithm (CS), firefly algorithm (FA), and bat algorithm (BA), which were developed during the last half a decade and are unexplored for RMP. Hence, for the further investigation of advanced optimization techniques for the RMP problems, seven different recently developed metaheuristics, viz., ABC, BBO, GSA, CS, FFA, BA, and TLBO, are experimented to optimize the RMP of a 3R robotic arm in this work. Two different cases,

Manuscript received July 21, 2013; revised December 23, 2013, May 5, 2014, and June 23, 2014; accepted July 20, 2014. Date of publication August 13, 2014; date of current version May 30, 2016.

P. Savsani is with the Department of Mechanical Engineering, Pacific Academy of Higher Education and Research University, Udaipur 313 024, India (e-mail: poonam.savsani@gmail.com).

R. L. Jhala is with the Department of Mechanical Engineering, Marwadi Education Foundation, Rajkot 360 003, India (e-mail: aryajhala@yahoo.com).

V. J. Savsani is with the Department of Mechanical Engineering, Pandit Deendayal Petroleum University, Gandhinagar 382 007, India (e-mail: vimal.savsani@gmail.com).

Digital Object Identifier 10.1109/JSYST.2014.2342292

i.e., one for the free workspace and the other for the workspace with obstacle, are considered for the experimentation.

The next section describes, in brief, about all the seven metaheuristics. Subsequent sections present the mathematical model for the considered RMP problem and the application of different algorithms for trajectory optimization, along with results and discussions based on the ability of the algorithms to find best, mean, and worst solutions; computational efforts; convergence of the algorithms; and the statistical test for the suitability of the algorithms.

II. Seven Different Metaheuristics

A. TLBO

The TLBO method mimics the effect of the influence of a teacher on the output of learners in a class [22]–[24]. Similar to other nature-inspired algorithms, TLBO is also a population-based method, which uses a population of solutions to proceed for the search. For TLBO, the population of solutions is considered as a group of learners/students and the different design variables as different subjects offered to learners, and the learners' result is analogous to the "fitness." The teacher is considered as the most learned person in the society (best solution). The working of TLBO is divided into two parts; the first part consists of "Teacher Phase," and the second part consists of "Learner Phase." The "Teacher Phase" means learning from the teacher, and the "Learner Phase" means learning through the interaction between learners. TLBO is the latest algorithm used in this paper, and it has gained popularity with its effective applications to many real-life optimization problems, such as multiobjective placement of the automatic regulators in the distribution system [25], data clustering [26], environmental economic problems [27], optimization of planar steel frames [28], dynamic economic dispatch problems [29], and 3-D image registration [30].

B. ABC

The ABC algorithm is an optimization algorithm based on the intelligent foraging behavior of a honeybee swarm. The colony of artificial bees consists of three groups of bees: employed bees, onlookers, and scouts [31]–[34]. The colony of the artificial bees is divided into two groups: first half of the colony consists of the employed artificial bees, and the second half includes the onlooker bees. Scout bees are the employed bee, whose food source has been abandoned. In ABC algorithm, the position of a food source represents a possible solution to the optimization problem (value of design variables), and the nectar amount of a food source corresponds to the quality of the associated solution (value of objective function). The solution is first updated in the employed bee phase, and depending on the quality of the solution of the employed bee phase, the onlooker bees update the solution by choosing the solutions of the employed bee phase, proportionally.

C. BBO

BBO [35]–[38] works on the philosophy of immigration and emigration of species from one island to the other. In BBO, each individual is considered as a "habitat" with a habitat suitability index (HSI). A good solution is analogous to an island with a high HSI, and a poor solution indicates an island with a low HSI. High-HSI solutions tend to share their features with low-HSI solutions. Low-HSI solutions accept a lot of new features from high-HSI solutions. Each individual has its own immigration rate λ and emigration rate μ. A good solution has higher μ and lower λ and vice versa. The immigration rate and the emigration rate are functions of the number of species in the habitat. The working of BBO is divided into two parts: one is migration, in which species move from one island to the other depending on its immigration and emigration rates, and the other is mutation, which means sudden change in the nature due to apparently random events such as large flotsam arriving from a neighboring habitat, disease, natural catastrophes, etc.

D. GSA

The GSA [39]–[41] works on the principal of Newton's gravitational law and law of motion. Any objects that possess mass attract each other, depending on their mass and the distance between them. In GSA, each solution is assumed to have some mass depending on its objective function value. Due to this mass, each solution gets attracted toward other solutions by obeying Newton's law. Moreover, the gravitational constant changes with the age of universe, and thus, the attraction between the solutions changes. This philosophy is the base for the working of GSA for the optimization.

E. CS

The CS [42]–[44] works on the obligate brood parasitic behavior of cuckoo species. A cuckoo is a bird species, which lays its eggs in the nests of other host birds that belong to different species. If the host bird discovers the eggs from the cuckoo, it abandons its nest and builds a new one at some other locations. However, some cuckoo species are specialized in mimicking the egg pattern of the other bird species, which makes it difficult for the host bird to identify. Moreover, the Levy flight concept of the species is incorporated in the CS. Levy flights are the foraging patterns followed by many animals and birds in nature. Based on the aforementioned two concepts, CS is developed and implemented for the optimization.

F. FA

The FA [46]–[48] is based on the flashing characteristics of the fireflies. This flashing characteristic attracts the female firefly toward the male firefly. The FA works by assuming, following idealized characteristics, that all fireflies are unisex, so that one firefly will be attracted to other fireflies regardless of their gender. The less bright one will get attracted toward the brighter one. Brightness depends on the objective function value. Moreover, the flashing characteristics are combined with the Levy flight behavior, as explained in the CS algorithm.

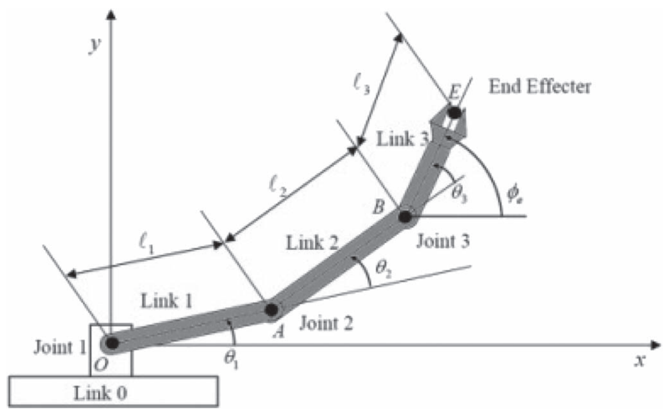

Fig. 1. 3-DOF Robotic arm [21].

G. BA

The BA [49]–[51] works on the echolocation behavior of the bats. The echolocation capability of the bats is used to find their prey and discriminate different types of insects even in the presence of complete darkness. These bats emit a very loud sound pulse and listen for the echo that bounces back from the surrounding objects. Their pulses vary in properties and can be correlated with their hunting strategies, depending on the species. The loudness also varies from the loudest when searching for prey and to a quieter base when homing toward the prey. The BA works by assuming that all bats use echolocation to sense distance, and they also "know" the difference between food/prey and background barriers. In addition, bats fly randomly with velocity v_i at position x_i with a fixed frequency $f_{\min}$, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, depending on the position of their target; loudness varies from a large A_0 to a minimum constant value $A_{\min}$.

III. Problem Formulation and Methodology

In this paper, a three degree of freedom (3-DOF) planar robotic arm is considered, as shown in Fig. 1, where the end-effector is required to move from the initial point to the final point in a free workspace.

For the considered problem, the complete trajectory is divided into two parts, which results in one intermediate configuration in between the initial and the final configurations of the robotic arm (see Fig. 2).

Initial joint angles are obtained by using inverse kinematics, which requires coordinates of the initial point (x_i, y_i) and angle ϕ_e for the end-effector. The redundancy can be avoided by considering the angle of link 2 (θ_2) as positive, i.e., in the counterclockwise direction with respect to link 1. The joint angles for the planar 3R manipulator are calculated by

$$\theta_1 = \tan^{-1}\frac{yy}{xx} \pm \cos^{-1}\left(\frac{xx^2 + yy^2 + l_1^2 - l_2^2}{2l_1\sqrt{xx^2 + yy^2}}\right) \quad (1)$$

$$\theta_2 = \pm\left(\pi - \cos^{-1}\left(\frac{l_1^2 + l_2^2 - xx^2 - yy^2}{2l_1 l_2}\right)\right) \quad (2)$$

$$\theta_3 = \phi_e - \theta_1 - \theta_2 \quad (3)$$

where $xx = x - l_3\cos(\phi_e)$ and $yy = y - l_3\sin(\phi_e)$; l_1, l_2, l_3 represent the length of links 1,2, and 3, respectively; x, y represent the coordinates of the end-effector; andϕ_e represents the angle of the end-effector. A flowchart for the inverse kinematics used in this work is shown in Fig. 3.

Initial and final velocities $(v_{\mathrm{ip}},\ v_{\mathrm{fp}} = 0,\ p = 1, 2, 3)$ and accelerations $(a_{\mathrm{ip}},\ a_{\mathrm{fp}} = 0,\ p = 1, 2, 3)$ are considered "zero" for all the joints. Angles and velocities of all the links at intermediate configuration, angle ϕ_e for the final configuration, time from initial to intermediate configuration, and time from intermediate to final configuration are considered as the design variables, which can vary during the optimization process between the lower and the upper limits specified. As initial acceleration is specified and intermediate acceleration is required to be obtained, this leads to the use of fourth-order trajectory from the initial to the intermediate configuration. Initial acceleration for the trajectory from the intermediate to the final configuration will be equal to the final acceleration of the fourth-order trajectory at the intermediate configuration so as to maintain the continuity of the acceleration for the complete trajectory. Hence, trajectory from the intermediate to the final configuration will have well-defined initial and final accelerations, which requires fifth-order trajectory to be used here.

The fourth-order trajectory to be used from the initial to the intermediate configuration is given by

$$\theta_{i,i+1}(t) = a_{i0} + a_{i1}t_i + a_{i2}t_i^2 + a_{i3}t_i^3 + a_{i4}t_i^4 \quad (4)$$

where $(a_{i0}, \ldots, a_{i4})$ are constants to be determined. The required configurations, velocities, and accelerations can be determined as given in

$$\begin{aligned} \dot{\theta}_i &= a_{i1} \quad \ddot{\theta}_i = 2a_{i2} \quad \theta_i = a_{i0} \\ \theta_{i+1} &= a_{i0} + a_{i1}T_{i,1} + a_{i2}T_{i,1}^2 + a_{i3}T_{i,1}^3 + a_{i4}T_{i,1}^4 \\ \dot{\theta}_{i+1} &= a_{i1} + 2a_{i2}T_{i,1} + 3a_{i3}T_{i,1}^2 + 4a_{i4}T_{i,1}^3 \end{aligned} \quad (5)$$

where T_i is the execution time from point i (*initial configuration*) to point $i+1$ (*intermediate configuration*). The aforementioned (5) can be solved for the required constants $(a_{i0}, \ldots, a_{i4})$ from the given values of initial configuration and the design variables for each link. Five unknowns can be solved using

$$\begin{aligned} a_{i0} &= \theta_i \quad a_{i1} = \dot{\theta}_i \quad a_{i2} = \frac{\ddot{\theta}_i}{2} \\ a_{i3} &= \frac{\left(4\theta_{i+1} - \dot{\theta}_{i+1}T_i - 4\theta_i - 3\dot{\theta}_i T_i^2\right)}{T_i^3} \\ a_{i4} &= \frac{\left(\dot{\theta}_{i+1}T_i - 3\theta_{i+1} + 3\theta_i + 2\dot{\theta}_i T_i + \ddot{\theta}_i T_i^2/2\right)}{T_i^4}. \end{aligned} \quad (6)$$

After obtaining values of constants for all the links, the acceleration of intermediate point $(i+1)$ can be obtained as given in

$$\ddot{\theta}_{i+1} = 2a_{i2} + 6a_{i3}T_{i,1} + 12a_{i4}T_{i,1}^2. \quad (7)$$

The fifth-order trajectory to be used between the intermediate and final configurations is given by

$$\theta_{i,i+1}(t) = b_{i0} + b_{i1}t_i + b_{i2}t_i^2 + b_{i3}t_i^3 + b_{i4}t_i^4 + b_{i5}t_i^5 \quad (8)$$

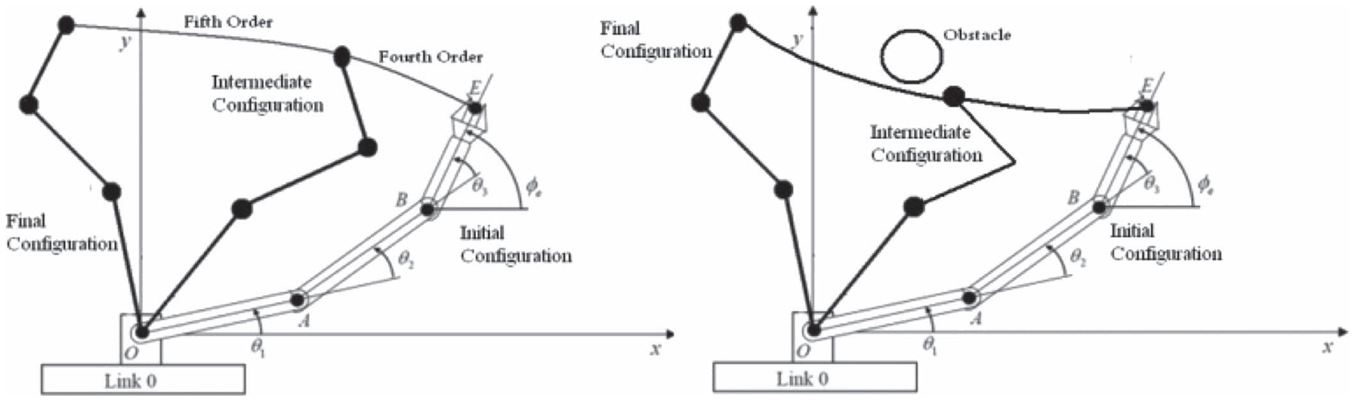

Fig. 2. Intermediate configuration on planned trajectory for the free workspace and obstacle existence workspace.

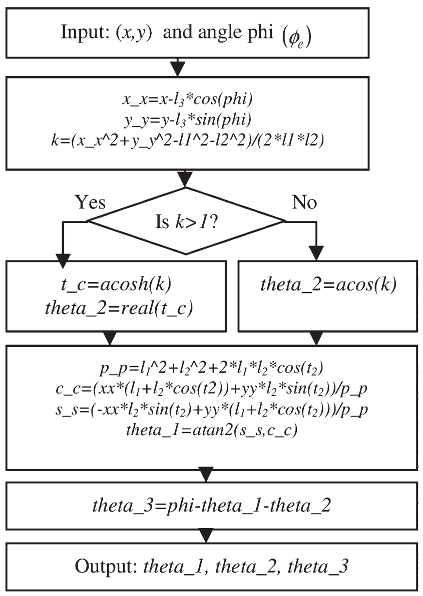
Input: (x,y) and angle phi (ϕ_e)
x_x=x-l3*cos(phi)
y_y=y-l3*sin(phi)
k=(x_x^2+y_y^2-l1^2-l2^2)/(2*l1*l2)
Is k>1?
Yes
No
t_c=acosh(k)
theta_2=real(t_c)
theta_2=acos(k)
p_p=l1^2+l2^2+2*l1*l2*cos(t2)
c_c=(xx*(l1+l2*cos(t2))+yy*l2*sin(t2))/p_p
s_s=(-xx*l2*sin(t2)+yy*(l1+l2*cos(t2)))/p_p
theta_1=atan2(s_s,c_c)
theta_3=phi-theta_1-theta_2
Output: theta_1, theta_2, theta_3

Fig. 3. Flowchart for the procedure of inverse kinematics.

where $(b_{i0}, \ldots, b_{i5})$ are constants to be determined. The required configurations, velocities, and accelerations can be determined as given in

$$\begin{aligned}\theta_i &= b_{i0} \quad \dot{\theta}_i = b_{i1} \quad \ddot{\theta}_i = 2b_{i2}\\ \theta_{i+1} &= b_{i0} + b_{i1}T_i + b_{i2}T_i^2 + b_{i3}T_i^3 + b_{i4}T_i^4 + b_{i5}T_i^5\\ \dot{\theta}_{i+1} &= b_{i1} + 2b_{i2}T_i + 3b_{i3}T_i^2 + 4b_{i4}T_i^3 + 5b_{i5}T_i^4\\ \ddot{\theta}_{i+1} &= 2b_{i2} + 6b_{i3}T_i + 12b_{i4}T_i^2 + 20b_{i5}T_i^3. \end{aligned} \tag{9}$$

Equation (9) is required to be solved for the six unknowns from the given final point and design variables using

$$\begin{aligned} b_{i0} &= \theta_i \quad b_{i1} = \dot{\theta}_i \quad b_{i2} = \frac{\ddot{\theta}_i}{2}\\ b_{i3} &= \frac{\left(20\theta_{i+1} - 20\theta_i - (8\dot{\theta}_{i+1} + 12\dot{\theta}_i)T_i - (3\ddot{\theta}_i - \ddot{\theta}_{i+1})T_i^2\right)}{2T_i^3}\\ b_{i5} &= \frac{\left(12\theta_{i+1} - 12\theta_i - (6\dot{\theta}_{i+1} + 6\dot{\theta}_i)T_i - (\ddot{\theta}_i - \ddot{\theta}_{i+1})T_i^2\right)}{2T_i^5}. \end{aligned} \tag{10}$$

For the optimization, three different objectives are considered, viz., total joint travelling distance (f_{joint}), total Cartesian trajectory length (f_{clength}), and the total time consumed (f_{time}) from the initial to the final configuration simultaneously. Total joint travelling distance given by (11) represents the energy consumed by the robotic arm; therefore, less joint travelling distance results in less energy consumed. That is

$$f_{\text{joint}} = \sum_{i=1}^{a}\sum_{j=2}^{b} |\theta_{ij} - \theta_{ij-1}| \tag{11}$$

where a is the number of robot links, and b is the number of joint configurations from the initial to the final configuration. Total Cartesian trajectory length given by (12), represents the workspace; therefore, minimizing Cartesian length will result in minimum work volume. That is

$$f_{\text{clength}} = \sum_{j=2}^{b} d\left((x,y)_j, (x,y)_{j-1}\right) \tag{12}$$

where $(x,y)_j$ represents the Cartesian coordinates of the jth configuration, and $d((x,y)_j, (x,y)_{j-1})$ calculates the distance between j and $j-1$ configurations.

Similarly, minimizing the total time given by (13) will operate the robot fast between the configurations, which results in the quicker operations and higher productivity. That is

$$f_{\text{time}} = T_{i,i+1} + T_{i+1,f} \tag{13}$$

where $T_{i,i+1}$ represents the time taken by the robotic arm to reach from the initial to the intermediate configuration, and $T_{i+1,f}$ represents the total time required from the intermediate to the final configuration. Hence, the objective function can be represented as given in

$$f = w_1 f_{\text{joint}} + w_2 f_{\text{clength}} + w_3 f_{\text{time}} \tag{14}$$

where w_1, w_2, and w_3 represent the constant assigned to each objective function to maintain the consistency of the function value so as to reduce the variation in one objective function compared to the other. The maximum value of f_{time}, which can be achieved, is 16 (8 + 8), as it depends on the design variable range. On the other hand, the maximum value achieved by f_{joint} and f_{clength} can be about 6.3 ($-\pi$ to π) and 7.5 (πR, where $R = l_1 + l_2 + l_3 = 2.5$ for the considered work). Hence, it is logical to have values of $w_1 = w_2 = 2$ and $w_3 = 1$ [8].

The other experiment is conducted using the workspace with the obstacle. For the current study, an obstacle with a circular shape is considered, as shown in Fig. 2. The change in the objective function due to the existence of the obstacle is given in

$$f_{\text{with\_obs}} = f / f_{\text{obs}} \tag{15}$$

where f_{obs} is the factor depending on the collision (intersection) of the robotic arm with the obstacle. If there is a collision of the robotic arm with the obstacle, then the objective function is penalized by considering $f_{\text{obs}} = 0$, which considers the objective function as infinite. On the other hand, if there is no collision of the robotic arm with the obstacle, then $f_{\text{obs}} = 1$, which leads to $f_{\text{with\_obs}} = f$. The intersection of the robotic arm with the obstacle can be obtained by using the "inpolygon" command of MATLAB, which returns value "1" if any point is inside the polygon and "0" if there is no point inside the polygon. The introduction of the obstacle in the workspace increases the complexity of the problem and offers a real challenge for the metaheuristics to find the optimal solutions.

All the joints are assigned a particular value of maximum torque (torque supplied by motor), which should not be exceeded. This leads to three constraints for three joints as given by

$$g_i(X) : Tor_i \leq Tor_{i\max} \tag{16}$$

where $i = 1, 2, 3$ represents three joints; X represents the design variables. Moreover, the constraints are imposed to ensure the robotic arm to attain the final configuration. The solution is considered as infeasible, if the final configuration is not reached with the given value of $\phi_{f,e}$ for the end-effector. This results in equality constraints, which are given in

$$g_4(X) : X_{\text{calculated}} = X_{\text{final}} \tag{17}$$

$$g_5(X) : Y_{\text{calculated}} = Y_{\text{final}}. \tag{18}$$

The values of Tor_i are calculated using the Lagrange–Euler dynamic algorithm [52], [54], which is given by (19)–(24). That is

$$Tor_i = \sum_{j=1}^{n} D_{ij}\ddot{\theta}_j + I_i\ddot{\theta}_i + \sum_{j=1}^{n}\sum_{k=1}^{n} D_{ijk}\dot{\theta}_j\dot{\theta}_k + D_i \tag{19}$$

where

$$D_{ij} = \sum_{p=\max(i,j)}^{n} Trace\left(U_{pj} J_p U_{pi}^T\right) \tag{20}$$

$$D_{ijk} = \sum_{p=\max(i,j,k)}^{n} Trace\left(U_{pjk} J_p U_{pi}^T\right) \tag{21}$$

$$D_i = \sum_{p=i}^{n} -m_p g^T U_{pi} {}^oT_i r_i \tag{22}$$

$$U_{ij} = \frac{\partial {}^oT_i}{\partial \theta_j}, \qquad j \leq i \tag{23}$$

$${}^oT_i = {}^oT_1 {}^1T_2 {}^2T_3 \ldots {}^{i-1}T_i. \tag{24}$$

T is the Denavit–Hartenberg transformation matrix, g^T is the gravitational matrix, r_i is the location of center of mass of link relative to the frame of the link.

The aforementioned problem requires nine design variables given in (25), which represent the intermediate joint angles and velocities, the end-effector angle, the time to reach the intermediate configuration from the initial configuration, and the time to reach the final configuration from the intermediate configuration. That is

$$\{\theta_{\text{int}1}, \theta_{\text{int}2}, \theta_{\text{int}3}, \phi_{f,e}, \dot{\theta}_{\text{int}1}, \dot{\theta}_{\text{int}2}, \dot{\theta}_{\text{int}3}, T_1, T_2\} \tag{25}$$

where $\theta_{\text{int}i,(1,2,3)}$ represents the intermediate joint angles; $\phi_{f,e}$ represents the end-effector for the final configuration; $\dot{\theta}_{\text{int}i,(1,2,3)}$ represents the intermediate velocity; and $T_{1,2}$ represents time from initial to intermediate configuration and from intermediate to final configuration, respectively. Upper and lower limits for the design variable are considered, such that it can cover the whole workspace generated by the robotic arm, and it is given by

$$\begin{aligned} -\pi &\leq \theta_{\text{int}i,(1,2,3)} \leq +\pi, \\ -\pi/4 &\leq \dot{\theta}_{\text{int}i,(1,2,3)} \leq +\pi/4,\ 0.1 \leq T_{1,2} \leq 8, \\ -\pi &\leq \phi \leq +\pi. \end{aligned} \tag{26}$$

Values for $Tor_{i\max} = 45,\ 20,$ and $5\ \text{N} \cdot \text{m}$ for the first, second, and third joints, respectively. The mass and length of the first (base link) and second links (intermediate link) are equal to 1 kg and 1 m, respectively, whereas it is taken as 0.5 kg and 0.5 m for the third link (end-effector link).

The flowchart for solving the aforementioned problem is given in Fig. 4. From the flowchart, it is observed that the procedure starts with the initialization of the robot and optimization algorithm parameters. The initial point is checked for the feasibility as per the link lengths, and then, the initial configuration is obtained using inverse kinematics. The optimization algorithm will start with the initial set of solutions (population), which produces intermediate configurations equal to the population size. The coordinate of the end-effector for the intermediate configuration is obtained using forward kinematics from the values of intermediate angles. The final configuration of the robotic arm is obtained from the value of $\phi_{f,e}$, which is checked for the required final configuration. Penalty is added to the objective function f ($f_{\text{with\_obs}}$ in case of obstacle existence workspace), if the final configuration is not attained. Tor_i is calculated using time T_1 and T_2, which is checked for $Tor_{i\max}$, and penalty is added in the objective function if torque constraints are violated. The objective function is calculated considering free workspace and workspace with obstacle. The procedure is repeated until the termination criterion is reached.

IV. Application of Metaheuristics

The aforementioned problem is investigated by using seven different metaheuristics developed from the year 2005 to 2012. These algorithms include the ABC algorithm, BBO, GSA, CS, FA, BA, and TLBO. Programming is done using MATLAB, and the code is run in an Intel core i3 laptop with M350 at 2.27 GHz. The investigation is divided into two experiments: 1) for the free

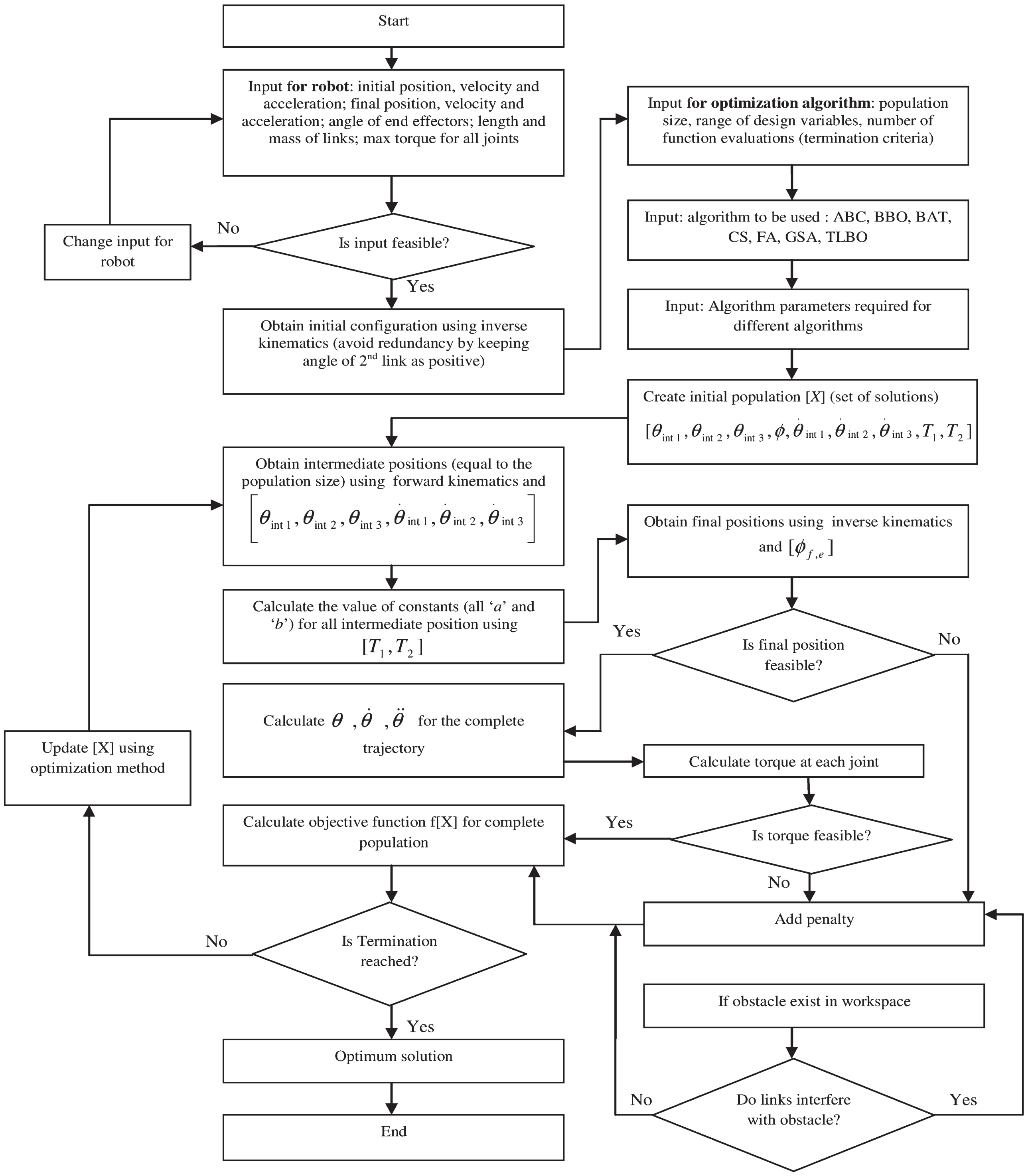
Start
Input for robot: initial position, velocity and acceleration; final position, velocity and acceleration; angle of end effectors; length and mass of links; max torque for all joints
Is input feasible?
No
Change input for robot
Yes
Obtain initial configuration using inverse kinematics (avoid redundancy by keeping angle of 2nd link as positive)
Input for optimization algorithm: population size, range of design variables, number of function evaluations (termination criteria)
Input: algorithm to be used : ABC, BBO, BAT, CS, FA, GSA, TLBO
Input: Algorithm parameters required for different algorithms
Create initial population [X] (set of solutions)
$[\theta_{int 1}, \theta_{int 2}, \theta_{int 3}, \phi, \dot{\theta}_{int 1}, \dot{\theta}_{int 2}, \dot{\theta}_{int 3}, T_1, T_2]$
Obtain intermediate positions (equal to the population size) using forward kinematics and $[\theta_{int 1}, \theta_{int 2}, \theta_{int 3}, \dot{\theta}_{int 1}, \dot{\theta}_{int 2}, \dot{\theta}_{int 3}]$
Calculate the value of constants (all 'a' and 'b') for all intermediate position using $[T_1, T_2]$
Obtain final positions using inverse kinematics and $[\phi_{f,e}]$
Is final position feasible?
Yes
No
Calculate θ, $\dot{\theta}$, $\ddot{\theta}$ for the complete trajectory
Calculate torque at each joint
Is torque feasible?
Yes
No
Update [X] using optimization method
Calculate objective function f[X] for complete population
Add penalty
Is Termination reached?
No
Yes
Optimum solution
End
If obstacle exist in workspace
Do links interfere with obstacle?
No
Yes

Fig. 4. Flowchart for the optimization of the trajectory for a robotic arm.

workspace and 2) for the obstacle existence workspace. The first experiment is also divided into two subparts, as follows: a) the RMP problem is solved using different metaheuristics with the same initial and final configurations, as given in [8], and the results obtained are compared with the results of GA [8], ABC, and TLBO [21]; b) effectiveness of algorithms are checked for different initial and final configurations across the workspace. The initial and final points are considered as (2.3, 0) and $(-2, 0)$, respectively, with end-effector angle at 80°.

Algorithms require specific algorithm-parameters for its effective working, and these parameters affect the quality of the result and searching ability of the algorithm. These parameters are problem specific, and it may happen that particular set of parameters may work effectively in a particular problem but may fail in some other problems. Hence, the parameters are decided after conducting experiments with different parameters, and the set of parameters, which has shown better performance in the results, is considered for the further experimentation.

TABLE I
COMPARISON OF "BEST," "MEAN," AND "STANDARD DEVIATION" (SD) USING DIFFERENT METAHEURISTICS FOR THE FREE WORKSPACE

Algorithms	Best	Mean	SD
GA [8]	13.14	-	-
TLBO [21]	12.03	12.82	1.58
ABC [21]	12.10	12.53	1.16
BA	12.66	15.46	2.59
FA	12.68	15.29	1.56
CS	12.20	13.41	1.77
GSA	14.81	16.32	0.68
BBO	12.97	13.93	0.91

TABLE II
COMPARISON OF GA, ABC, AND TLBO

Function value	GA [8]	ABC	TLBO
Total joint travelling distance (f_{joint})	1.91	1.87	1.88
Total Cartesian trajectory length ($f_{clength}$)	3.28	3.32	3.32
Total time (f_{time})	2.76	1.70	1.62
Final function (f)	13.14	12.10	12.03
Function Evaluations	16000	4000	4000

For ABC, number of employed bees (N_b) = onlooker bees; for BBO, maximum immigration rate $(I_{\max}) = 1$, minimum immigration rate $(I_{\min}) = 0$, and mutation coefficient $(m) = 0.05$; for GSA, gravitation constant $(G_0) = 100$, gravitation reduction factor $(\alpha) = 20$, and normalizing factor $(F_{\text{norm}}) = 2$; for CS, probability factor $(pa) = 0.25$, and Levy flight factor $(\beta) = 1.5$; for FA, attractiveness factor $(\beta_o) = 0.5$, absorption coefficient $(\gamma) = 1$, and randomization parameter $(\alpha) = 0.2$; for BA, maximum frequency $(f_{\max}) = 100$, minimum frequency $(f_{\min}) = 0$, loudness factor $(A_o) = 1$, pulse rate factor $(r_o) = 0.5$, pulse rate change factor $(\gamma) = 0.9$, and loudness change factor $(\alpha) = 0.9$. TLBO does not require any algorithm parameter for its working, and this is one of the effective advantages of TLBO. All the algorithms are investigated using 4000 function evaluations with a population size of 20. All the algorithms are heuristics methods; therefore, it is required to obtain the results for different runs. All the results are obtained for 25 independent runs for all the algorithms.

The best solutions, mean solutions, and standard deviations obtained in 25 independent runs are summarized in Table I, considering same initial and final configurations given in [8] with free workspace. The results are compared with the results obtained by using GA [8] and by using ABC and TLBO [21]. It is observed from the results that ABC and TLBO have produced better results compared to all the considered algorithms. In addition, all the considered algorithms have produced better results than GA [8].

Moreover, in [8], results of GA were obtained using the population size of 200 and maximum generation of 80, resulting in 16 000 function evaluations. However, ABC and TLBO have produced better results than GA using only 4000 function evaluations, which is four times less than GA. Values of all the objective functions are given in Table II for GA, ABC, and TLBO. It is further observed from the results that there is not much change in the results of total joint travelling distance and total Cartesian trajectory length, but ABC and TLBO have produced better results, in terms of total time consumed from the initial to the final configuration.

TABLE III
COMPARISON OF "BEST," "MEAN," AND "STANDARD DEVIATION" (SD) OBTAINED USING DIFFERENT METAHEURISTICS FOR THE OBSTACLE EXISTENCE WORKSPACE

Algorithms	Best	Mean	SD
GA [8]	25.63	-	-
BA	19.93	30.82	7.79
FA	19.34	25.13	6.15
TLBO	17.15	18.54	0.49
ABC	18.55	18.66	0.11
CS	18.57	19.92	1.31
GSA	45.57	66.82	20.25
BBO	19.30	23.22	5.93

TABLE IV
RESULT OF THE HOLM–SIDAK MULTIPLE COMPARISON TEST FOR THE FREE WORKSPACE AND WORKSPACE WITH OBSTACLE

Free-Workspace		Workspace with obstacle	
Algorithms	p-value	Algorithms	p-value
TLBO--GSA	0	ABC--GSA	0
ABC--GSA	0	TLBO--GSA	0
CS--GSA	0	BA--ABC	0.0001
GSA--BBO	0	CS--GSA	0.0001
FA--GSA	0	FA--ABC	0.0002
BA--GSA	0	BA--TLBO	0.0004
BA--TLBO	0.0028	FA--TLBO	0.0009
BA--ABC	0.0031	GSA--BBO	0.0013
BA--CS	0.0075	BA--CS	0.0052
BA--BBO	0.0589	FA--CS	0.0097
FA--TLBO	0.1002	BA--BBO	0.035
FA--ABC	0.1068	ABC--BBO	0.0524
BA--FA	0.1545	FA--BBO	0.0589
FA--CS	0.1918	TLBO--BBO	0.1207
TLBO--BBO	0.2401	FA--GSA	0.151
ABC--BBO	0.253	ABC--CS	0.2212
CS--BBO	0.4058	BA--GSA	0.2261
FA--BBO	0.631	TLBO--CS	0.4089
TLBO--CS	0.7282	CS--BBO	0.4609
ABC--CS	0.7525	TLBO--ABC	0.6875
TLBO--ABC	0.9743	BA--FA	0.8181

Results for the workspace with the existence of obstacle are given in Table III, in which results for the best solutions, mean solutions, and standard deviations are compared. It is observed from the results that all the algorithms, except GSA, have produced better results than GA. Results produced by TLBO is better than all the other algorithms. There is marginal difference in the results produced by ABC and CS.

From the results, it can be interpreted that TLBO is the better algorithm than all the considered algorithms, but is it really statistically significant than other algorithms? Hence, it is required to test the results obtained by 25 runs for the statistical significance for all the algorithms. Student's t-test can be used to compare only two groups. When there are more than two groups, e.g., k groups, then using a t-test will mislead the statistical significance because if a t-test is performed at fixed $\alpha = 0.05$, there is a probability of $k * 0.05$ to find a difference when it is not. The Holm–Sidak test is a step-down "recursive reject" because it applies an accept/reject criterion on the sorted set of null hypothesis. This starts from the lower p-value and going up to the acceptance of the null hypothesis. For each comparison, the alpha value is set according to Sidak correction of Bonferroni inequality [53]. The Holm–Sidak multiple test is performed on both the cases, i.e., without and with obstacle.

TABLE V
BEST SOLUTIONS OBTAINED BY DIFFERENT METAHEURISTICS FOR THE DIFFERENT CASES IN WORKSPACE [21]

$\phi_e \rightarrow$	0	-135	-45	0	45	-180	45
Algorithms	2---4	9--11	5--6	2--7	9—6	4--10	6--9
BA	25.861	16.337	10.648	19.331	15.592	29.706	16.265
FA	25.845	16.312	10.937	19.103	15.023	29.495	16.184
TLBO	25.716	16.063	10.604	18.895	15.200	29.371	15.383
ABC	25.736	16.076	10.605	18.928	14.909	29.355	16.138
CS	25.764	16.211	10.669	18.931	14.957	29.369	15.728
GSA	26.071	16.475	10.909	19.17	14.9961	29.386	16.803
BBO	26.287	16.340	11.134	19.406	15.307	29.612	16.790

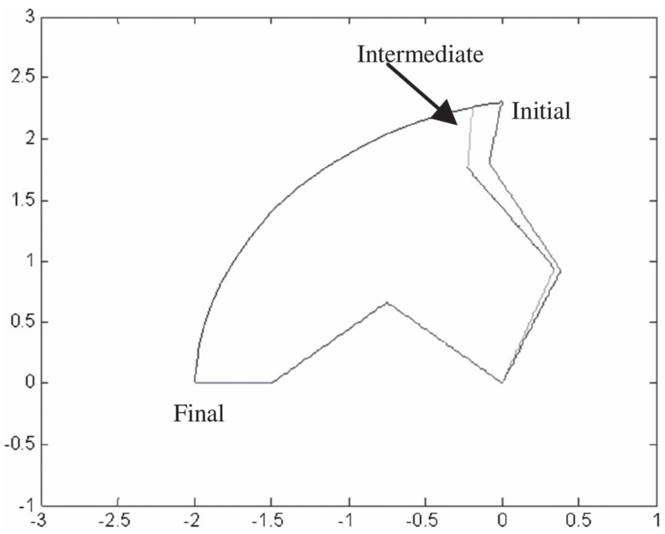

Fig. 5. Configurations for free workspace.

The results are given in Table IV for free workspace and in Table V for the obstacle existence workspace.

It is observed for the results given in Table IV that the p-value of TLBO–ABC is 0.9743, which indicates that there is no statistical difference between the results of ABC and TLBO for the free workspace. A lower p-value indicates large statistical difference in the obtained results. It depends on the user to select the appropriate algorithm, depending on other requirements such as the quality of solution, convergence, etc. In addition, it is observed that the results produced by GSA, BA, and FA have large statistical difference compared to ABC, TLBO, and CS for both free and obstacle existence workspace. For obstacle existence workspace, the results of TLBO, ABC, and CS are statistically different than the free workspace. Configurations, variations of angles, velocities, and torques are shown in Figs. 5–12, for free workspace and obstacle existence workspace.

It is observed from the figures that, for free workspace, torque constraints are active constraints as all the joint torque values are reached to the maximum allowable value of torque, and this is the reason for consuming less time (nearly three times) in the free workspace motion compared to obstacle existence workspace. Furthermore, it is observed that the velocity graphs are smooth in both the cases.

It is an obvious fact that the increase in the DOF of the robotic arm increases the number of design variables for the optimization problems. Hence, the increase in the number of DOF leads to the increase of computational efforts of the

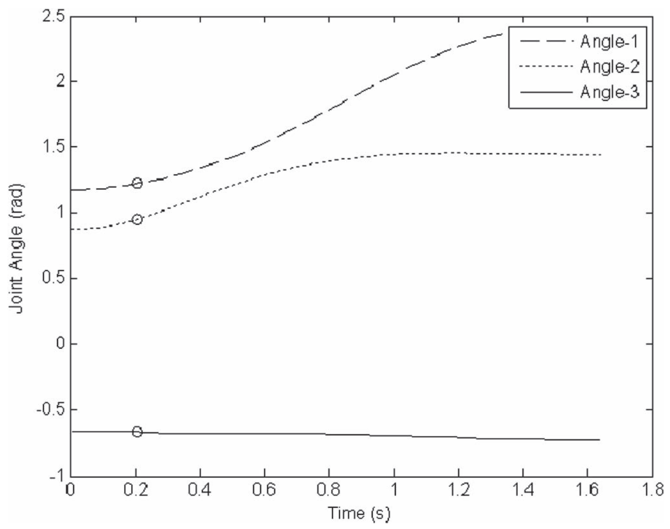

Fig. 6. Variations of angles for free workspace.

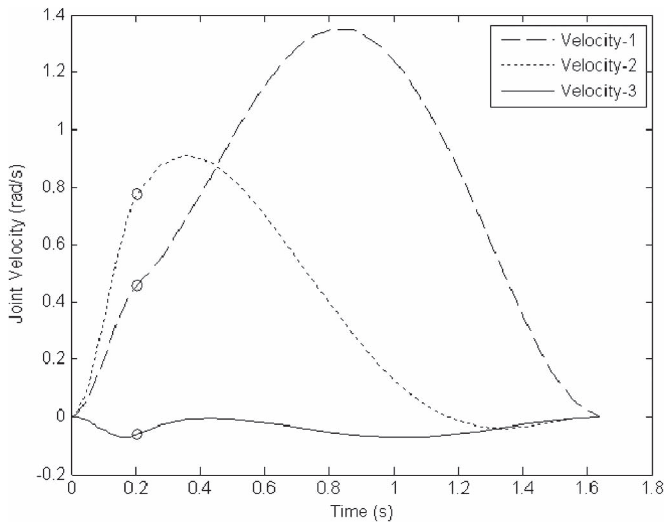

Fig. 7. Variations of velocities for free workspace.

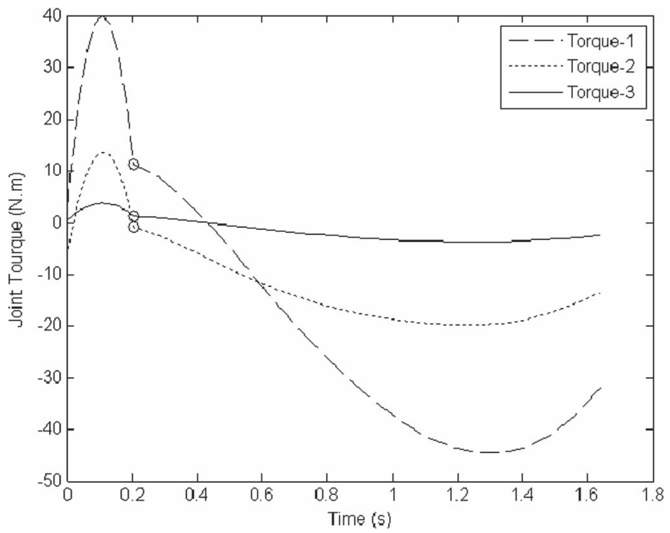

Fig. 8. Variations of torque for free workspace.

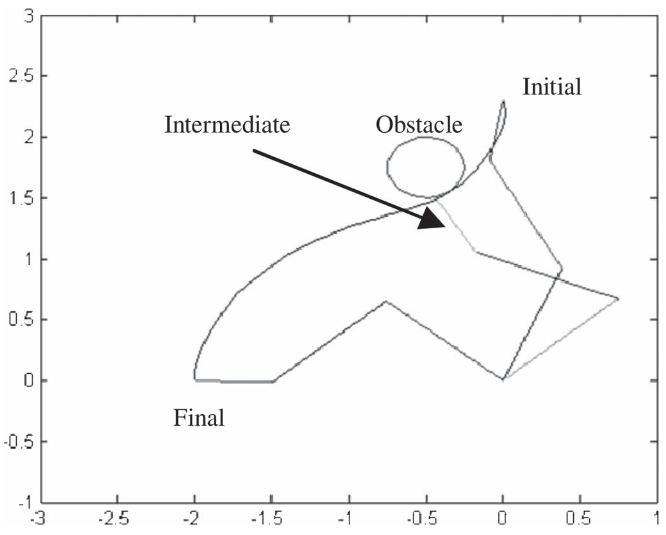

Fig. 9. Configurations for obstacle existence workspace.

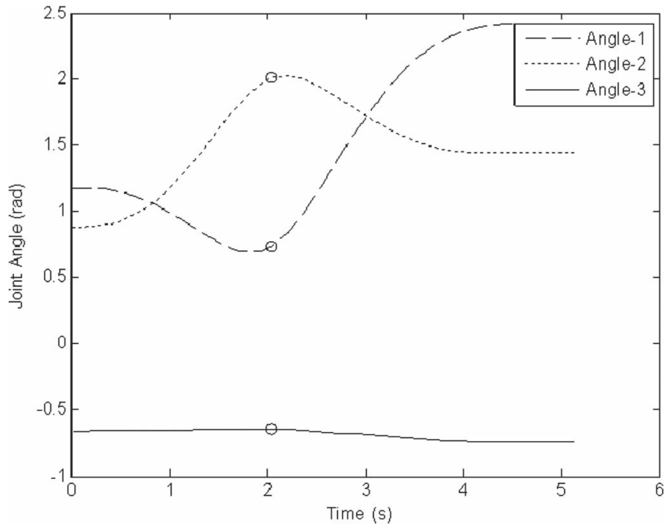

Fig. 10. Variations of angles for obstacle existence workspace.

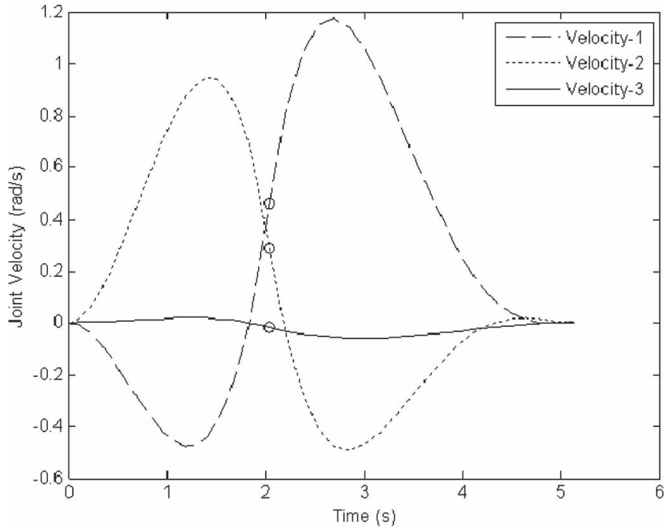

Fig. 11. Variations of velocities for obstacle existence workspace.

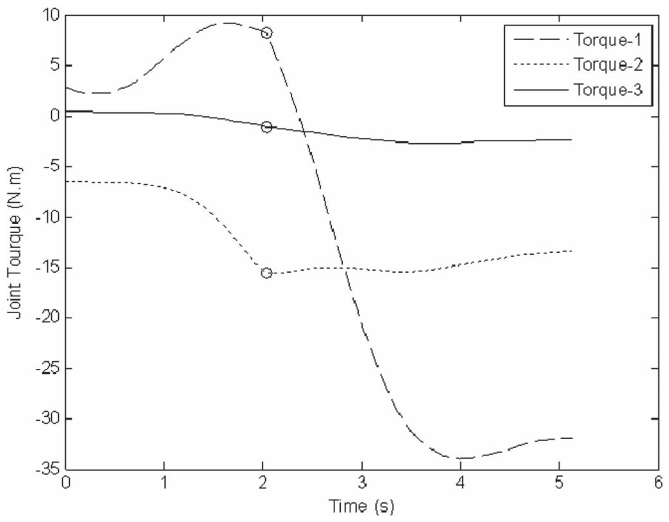

Fig. 12. Variations of torque for obstacle existence workspace.

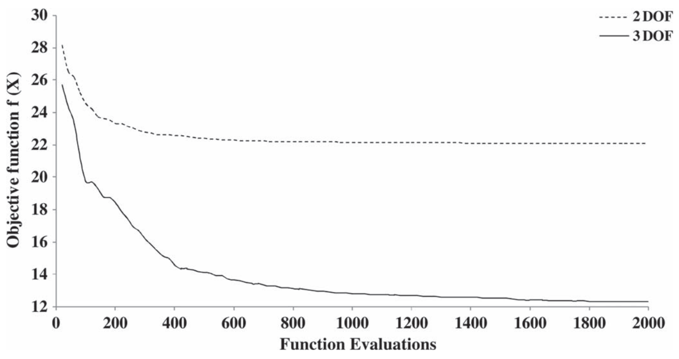

Fig. 13. Computational effort for 2-DOF and 3-DOF robotic arms.

optimization algorithm. Computational efforts are compared for 2-DOF and 3-DOF robotic arms by using TLBO. Variations of the objective function for five different runs are shown in Fig. 13, for the 2-DOF and 3-DOF robotic arms. It is observed from the figure that convergence is achieved in nearly 800 function evaluations for the 2-DOF robotic arm compared to nearly 1800 function evaluations for the 3-DOF robotic arm. The reason for the increase in computational effort is the increase in the number of design variables, as 2 DOF requires only six design variables compared to nine variables for 3 DOF.

In the second part of the experiment, different points are identified in the whole workspace, as shown in Fig. 14. These points includes extreme points (EP) indicated by 1, 2, 3, and 4, which lie at periphery of the workspace; intermediate points (IP) indicated by 9, 10, 11, and 12, which lie in between the base and the extreme positions on the horizontal and vertical axis; and space points (SP) indicated by 5, 6, 7, and 8, which lie at any location in the space. To avoid the infeasibility of locations, EPs lie at the distance of 2.49 from the base, while IPs lie at a distance of 1.25 from the base and SPs at 1.25 from each IP. It is cumbersome to consider all the movements, such as EP–EP, IP–IP, SP–SP, EP–SP, IP–SP, EP–IP, and SP–IP, for each point.

Hence, for the experimentation, only one movement is considered randomly for each case. Seven different movements are considered with different angle ϕ_e, for which the results are summarized in Tables V and VI and obtained in

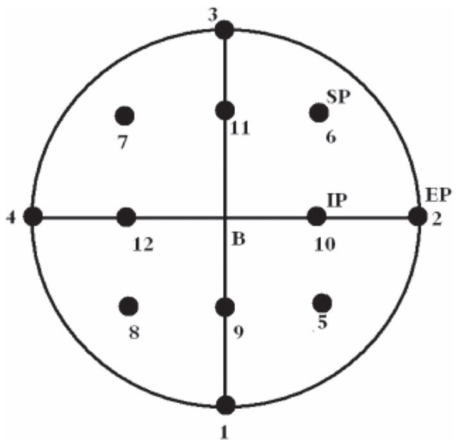

Fig. 14. Different locations in free workspace.

TABLE VI
MEAN SOLUTIONS OBTAINED BY DIFFERENT METAHEURISTICS FOR THE DIFFERENT CASES IN WORKSPACE

$\phi_e \rightarrow$	0	-135	-45	0	45	-180	45
Algorithms	2---4	9--11	5--6	2--7	9—6	4--10	6--9
BA	27.917	18.000	12.698	22.720	16.507	30.398	18.890
FA	26.928	16.488	12.738	20.756	15.622	31.459	18.018
TLBO	25.736	16.206	10.706	19.330	15.520	29.493	16.306
ABC	25.747	16.236	10.696	19.320	15.498	29.455	16.359
CS	26.008	16.297	10.72	19.021	15.523	29.506	17.059
GSA	26.842	17.465	11.903	20.519	16.104	30.002	17.778
BBO	27.296	16.668	11.600	20.113	15.992	29.927	17.804

TABLE VII
FRIEDMAN RANK FOR DIFFERENT ALGORITHMS

Rank	BA	FA	TLBO	ABC	CS	GSA	BBO
Friedman Rank	47	39	12	11	19	35	33
Rank	7	6	2	1	3	5	4
Normalized rank	4.27	3.55	1.09	1.00	1.73	3.18	3.00

100 independent runs, which show the best solutions and mean solutions, respectively.

It is further observed from the results that TLBO, ABC, and CS have performed well compared to other algorithms. The tendency to find the best solutions for all the cases is nearly same for all the considered algorithms. There is slight difference in the tendency of the algorithms to find the mean solutions and worst solutions. Hence, it is required to quantify the performance of the algorithms by providing ranks. The well-known method to rank metaheuristics is the Friedman rank [55]. The results of the Friedman rank test for the mean solutions are shown in Table VII. It is observed from the Friedman rank test that ABC has ranked 1 followed by TLBO and CS. In addition, there is a marginal difference in the Friedman rank (1 for ABC and 12 for TLBO) and normalized rank (1 for ABC and 1.09 for TLBO). The next best algorithm after ABC and TLBO is CS, and the worst performing algorithm is BA.

Convergence of the algorithm also plays an important role for the optimization algorithms. Convergence graphs for all the considered algorithms are shown in Fig. 15. The plot is for the best solutions averaged for ten runs.

It is observed from the graph that TLBO and ABC have good convergence tendency compared to other algorithms. CS and

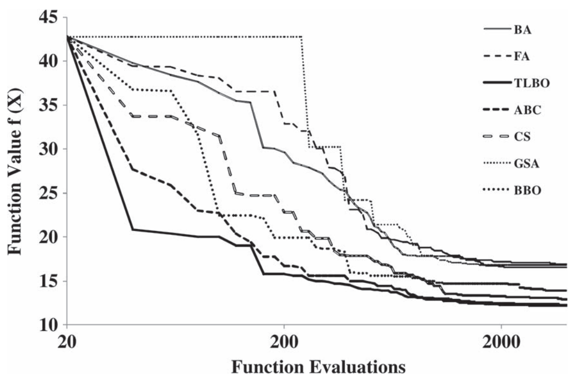

Fig. 15. Convergence graphs for the considered metaheuristics.

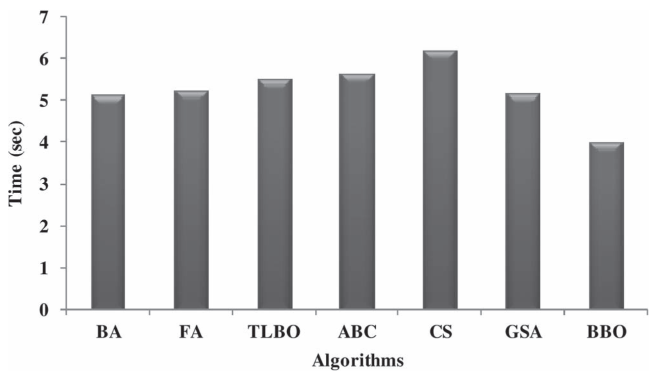

Fig. 16. Comparison for the computational time of the free workspace.

BBO have nearly same convergence characteristics, whereas FA, GSA, and BA have poor convergence tendency. Computational time is also an important characteristic of any algorithms. Hence, a comparison for the algorithms is also made for the computational effort required in terms of time, and it is shown in Fig. 16. It is observed from the result that computational time for the CS is more, as compared to the other algorithms, and BBO is having the least computational effort. TLBO and ABC requires slightly more time than the other algorithms.

V. CONCLUSION

Seven different metaheuristics developed between 2005 and 2012 have been experimented on the RMP problem for the trajectory optimization of a robotic arm. These algorithms include the ABC algorithm, BBO, GSA, CS, FA, BA, and TLBO. Three different objective functions, to minimize joint travelling time, joint travelling distance, and total joint Cartesian lengths, are considered simultaneously. TLBO, ABC, and CS have performed well for the considered trajectory optimization problem for the searching of best solutions, convergence, and statistical significance for the free workspace and the workspace with obstacle existence. BA, FA, and GSA performed poorly for the considered problem. The Friedman rank test has shown the better performance of ABC followed by TLBO and CS. However, the difference between ABC and TLBO is marginal. TLBO does not require any algorithm parameter for its working. BA, FA, and GSA require many algorithm parameters for its performance; therefore, further experiments can be conducted to tune these parameters for the effective working of the algorithms.

REFERENCES

[1] E. Masehian and D. Sedighizadeh, "Classic and heuristic approaches in robot motion planning—A chronological review," in *Proc. World Academy Sci., Eng. Technol.*, 2007, pp. 101–106.

[2] J. F. Canny, *The Complexity of Robot Motion Planning*. Cambridge, CA, USA: MIT Press, 1988.

[3] Y. K. Hwang and N. Ahuja, "Gross motion planning—A survey," *ACM Comp. Surv.*, vol. 24, no. 3, pp. 219–291, Sep. 1992.

[4] T. Lozano-Perez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," in *Proc. Commun. ACM*, 1979, pp. 560–570.

[5] W. F. Carriker, P. K. Khosla, and B. H. Krogh, "The use of simulated annealing to solve the mobile manipulator path planning problem," in *Proc. IEEE ICRA*, 1990, pp. 204–209.

[6] T. Makino, H. Yokoi, and Y. Kakazu, "Development of a motion planning system for an agricultural mobile robot," in *Proc. SICE Annu.*, 1999, pp. 959–962.

[7] Z. Qidan, Y. Yongjie, and X. Zhuoyi, "Robot path planning based on artificial potential field approach with simulated annealing," in *Proc. ISDA*, 2006, pp. 622–627.

[8] I. B. Kazem, A. I. Mahdi, and A. T. Oudha, "Motion planning for a robot arm by using genetic algorithm," *Jordan J. Mech. Ind. Eng.*, vol. 2, pp. 131–136, 2008.

[9] Y. Davidor, "Robot programming with a genetic algorithm," in *Proc. IEEE Int. Conf. Comput. Sys. & Soft. Eng.*, 1990, pp. 186–191.

[10] M. Hamdan and M. E. El-Hawary, "A novel genetic algorithm searching approach for dynamic constrained multicast routing," in *Proc. IEEE/CCECE*, 2003, pp. 1127–1130.

[11] J. K. Parker, A. R. Khoogar, and D. E. Goldberg, "Inverse kinematics of redundant robots using genetic algorithms," in *Proc. IEEE ICRA*, 1989, vol. 1, pp. 271–276.

[12] L. Qing, T. Xinhai, X. Sijiang, and Z. Yingchun, "Optimum path planning for mobile robots based on a hybrid genetic algorithm," in *Proc. HIS*, 2006, pp. 53–58.

[13] J. Solano and D. I. Jones, "Generation of collision-free paths, a genetic approach," in *IEEE Colloq. Gen. Algorithms Control Sys. Eng.*, 1993, pp. 51–56.

[14] J. Deneubourg, P. Clip, and S. Camazine, "Ants, buses and robots-self-organization of transportation systems," in *Proc. Perception Action*, 1994, pp. 12–23.

[15] M. M. Mohamad, N. K. Taylor, and M. W. Dunnigan, "Articulated robot motion planning using ant colony," in *Proc. IEEE Int. Conf. Optim., Intel. Sys.*, 2006, pp. 690–695.

[16] H. Ying-Tung, C. Cheng-Long, and C. Cheng-Chih, "Ant colony optimization for best path planning," in *Proc. IEEE/ISCIT*, 2004, pp. 109–113.

[17] Q. Yuan-Qing, S. De-Bao, L. Ning, and C. Yi-Gang, "Path planning for mobile robot using the particle swarm optimization with mutation operator," in *Proc. Int. Conf. Mach. Learn. Cybern.*, 2004, pp. 2473–2478.

[18] M. Hua-Qing, Z. Jin-Hui, and Z. Xi-Jing, "Obstacle avoidance with multiobjective optimization by PSO in dynamic environment," in *Proc. Int. Conf. Mach. Learn. Cybern.*, 2005, vol. 5, pp. 2950–2956.

[19] W. Li, L. Yushu, D. Hongbin, and X. Yuanqing, "Obstacle-avoidance path planning for soccer robots using particle swarm optimization," in *Proc. IEEE Int. Conf. ROBIO*, 2006, pp. 1233–1238.

[20] P. Bhattacharjee, P. Rakshit, I. Goswami, A. Konar, and A. K. Nagar, "Multi-robot path-planning using artificial bee colony optimization algorithm," in *Proc. 3rd World Congr. NaBIC*, 2011, pp. 219–224.

[21] P. Savsani, R. L. Jhala, and V. J. Sasvani, "Optimized trajectory planning of a robotic arm using teaching learning based optimization (TLBO) and artificial bee colony (ABC) optimization techniques," in *Proc. 7th IEEE Syst. Conf.*, Orlando, FL, USA, Apr. 15–18, 2013, pp. 381–386.

[22] R. V. Rao, V. J. Savsani, and D. P. Vkharia, "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303–315, Mar. 2011.

[23] R. V. Rao, V. J. Savsani, and D. P. Vkharia, "Teaching-learning-based optimization: A novel optimization method for continuous nonlinear large scale problems," *Inf. Sci.*, vol. 183, no. 1, pp. 1–15, Jan. 2012.

[24] R. V. Rao and V. J. Savsani, *Mechanical Design Optimization Using Advanced Optimization Technique*. New York, NY, USA: Springer-Verlag, 2012.

[25] H. Hosseinpour, T. Niknam, and S. I. Taheri, "A modified TLBO algorithm for placement of AVRs considering DGs," in *Proc. 26th Int. Power Syst. Conf.*, Tehran, Iran, 31st Oct.–2nd Nov. 2011, pp. 16–23.

[26] S. C. Satapathy and A. Naik, "Data clustering based on teaching-learning-based optimization," in *Swarm, Evolutionary, Memetic Computing. Lecture Notes in Computer Science*, vol. 7077. Berlin, Germany: Springer-Verlag, 2011, pp. 148–156.

[27] K. R. Krishnanand, B. K. Panigrahi, P. K. Rout, and A. Mohapatra, "Application of multi-objective teaching-learning-based algorithm to an economic load dispatch problem with incommensurable objectives," in *Swarm, Evolutionary, Memetic Computing, Lecture Notes in Computer Science*, vol. 7076. Berlin, Germany: Springer-Verlag, 2011, pp. 697–705.

[28] V. Toğan, "Design of planar steel frames using teaching-learning based optimization," in *Eng. Struct.*, 2012, pp. 225–232.

[29] T. Niknam, F. Golestaneh, and M. S. Sadeghi, " θ-multiobjective teaching-learning-based optimization for dynamic economic emission dispatch," *IEEE Syst. J.*, vol. 6, no. 2, pp. 341–352, Jun. 2012.

[30] A. Jani, V. J. Savsani, and A. Pandya, "3D affine registration using Teaching-Learning Based Optimization," *3D Res.*, vol. 4, no. 3, Sep. 2013.

[31] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Engineering Faculty, Computer Engineering Department, Erciyes University, Kayseri, Turkey, Tech. Rep.-TR06, 2005.

[32] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *IEEE Swarm Intell. Symp.*, Indianapolis, IN, USA, 2006, pp. 49–53.

[33] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Proc. IFSA LNAI*, P. Melin, O. Castillo, L. T. Aguilar, J. Kacprzyk, and W. Pedrycz, Eds., Berlin, Germany, 2007, vol. 4529, pp. 789–798, Springer-Verlag.

[34] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 687–697, Jan. 2008.

[35] D. Simon, "Biogeography based optimization, evolutionary computation," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[36] H. Ma, D. Simon, M. Fei, and Z. Xie, "Variations of biogeography-based optimization and Markov analysis," *Inf. Sci.*, vol. 220, pp. 492–506, Jan. 2013.

[37] D. Simon, "A dynamic system model of biogeography-based optimization," *Applied Soft Computing*, vol. 11, no. 8, pp. 5652–5661, Dec. 2011.

[38] H. Ma and D. Simon, "Analysis of migration models of biogeography-based optimization using Markov theory," *Eng. Appl. Artif. Intell.*, vol. 24, no. 6, pp. 1052–1060, Sep. 2011.

[39] E. Rashedi, H. Nezamabadi, and S. Saryazdi, "GSA: A Gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, Jun. 2009.

[40] E. Rashedi, H. Nezamabadi, and S. Saryazdi, "BGSA: Binary gravitational search algorithm," *Natural Comput.*, vol. 9, no. 3, pp. 727–745, Sep. 2010.

[41] E. Rashedi, H. Nezamabadi, and S. Saryazdi, "Filter modeling using gravitational search algorithm, Engineering Applications of Artificial Intelligence," *Eng. Appl. Artif. Intell.*, vol. 24, no. 1, pp. 117–122, Feb. 2011.

[42] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proc. IEEE World Congr. NaBIC*, Coimbatore, India, Dec. 2009, pp. 210–214.

[43] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *Int. J. Math. Model. Numerical Optim.*, vol. 1, no. 4, pp. 330–343, 2010.

[44] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems," *Eng. Comput.*, vol. 29, no. 1, pp. 17–35, Jan. 2013.

[45] X.-S. Yang, *Nature Inspired Metaheuristic Algorithms*. Bristol, U.K.: Luniver press, 2010.

[46] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Proc. SAGA Lecture Notes Comput. Sci.*, 2009, vol. 5792, pp. 169–178.

[47] X.-S. Yang, "Firefly algorithm, Lévy flights and global optimization," in *Proc. Res. Develop. Intell. Syst. 26th*, M. Bramer, R. Ellis, and M. Petridis, Eds., London, U.K., 2010, pp. 209–218, Springer-Verlag.

[48] X.-S. Yang, S. S. S. Hosseini, and A. H. Gandomi, "Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect," *Appl. Soft Comput.*, vol. 12, pp. 1180–1186, 2012.

[49] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Proc. NISCO*, vol. 284, *Studies in Computational Intelligence*, J.R. Gonzalez, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., Berlin, Germany, 2010, pp. 65–74, Springer-Verlag.

[50] X. S. Yang, "Bat algorithm for multiobjective optimization," *Int. J. Bio-Inspired Comput.*, vol. 3, no. 5, pp. 267–274, 2011.

[51] X.-S. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, 2012.

[52] J. Craig, *Introduction to Robotics: Mechanics and Control*. Reading, MA, USA: Addison-Wesley, 1989.

[53] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian J. Statist.*, vol. 6, no. 2, pp. 65–70, 1979.

[54] S. B. Niku, *Introduction to Robotics, Analysis, Systems, Applications*. Upper Saddle River, NJ, USA: Pearson Education, Inc., 2003.

[55] D. Joaquín, G. Salvador, M. Daniel, and H. Francisco, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, 2011.

Poonam Savsani She is currently a Research Scholar with the Pacific Academy of Higher Education and Research University, Uidapur, India. Her research interests include synthesis of mechanisms, evolutionary robotics, trajectory optimization of redundant robots, and investigation of advanced metaheuristics for the robot path planning of different robots.

R. L. Jhala He is currently a Professor with Marwadi Education Foundation, which is affiliated to Gujarat Technological University, Gujarat, India. He has 15 years of experience in the academia and industry. He is the author of more than 25 journal and conference papers. His research interest includes automobile design, finite-element methods, and robotics.

V. J. Savsani He is currently an Assistant Professor with Pandit Deendayal Petroleum University, Gandhinagar, India. He has ten years of academic experience. He is the author of more than 30 papers in reputed journals and conferences and one book published by Springer London. His research interests include advanced metaheuristics and its investigation to different engineering problems, automobile suspension optimization, wind farm layout, and structure and topology optimization.