Neural Computing and Applications (2021) 33:14119–14131
https://doi.org/10.1007/s00521-021-06050-2

ORIGINAL ARTICLE

A new modified grey wolf optimization algorithm proposal for a fundamental engineering problem in robotics

Serkan Dereli[1]

Received: 29 November 2020 / Accepted: 15 April 2021 / Published online: 29 April 2021

Abstract
In this study, the solution of inverse kinematics, which is the most fundamental problem in the field of robotics, is handled with the grey wolf optimization algorithm. Grey wolves belong to the Canis Lupus species and they act in a number of organization and cooperation while hunting in swarm in nature. Thanks to this heuristic technique, which was created by transferring this collaboration first to the algorithm and then to the code, many engineering problems were quickly solved. Similarly, in this study, inverse kinematics solution, which is an engineering problem, was solved with the grey wolf swarm optimization technique. The results are given in comparison with the traditional grey wolf algorithm and the modified grey wolf algorithm obtained by improving one of the control parameters of this algorithm and with other swarm-based algorithms. According to these results, it has been clearly observed that the grey wolf algorithm produces similar results with other swarm-based algorithms, but the modified grey wolf algorithm produces better values. This shows that the grey wolf optimization algorithm can achieve much better convergence by modifying or strengthening it.

Keywords Heuristic algorithm · Swarm intelligent · Grey wolf optimization · Robotics · Inverse kinematics

1 Introduction

Swarm-based methods, which have an important place in the solution of engineering problems, are becoming widespread day by day, and new techniques are being included in literature [1]. Researchers first met with ant colony optimization based on ant cooperation early 1990 [2]. Over the years, it continued in the form of a particle swarm optimization [3], artificial bee colony [4] and firefly algorithm [5]. It is a fact that in the last thirty years, classical methods have left their place to such heuristic methods in solving engineering problems [6], because researchers for the last fifty years have focused on solving extremely complex nonlinear problems [7]. Nonlinear problems contain complex mathematical equations and transformations in their solutions. Classical methods such as analytic, iterative get one good value in solving these equation [8].

✉ Serkan Dereli
dereli@subu.edu.tr

1 Sakarya University of Applied Sciences, Computer Technologies and Programming, Sakarya, Turkey

Inverse kinematics problem has become a fundamental field of study with the introduction of robot mechanisms into our lives, because it is known as the most basic element of moving a mechanism [9]. Inverse kinematics for a robot manipulator is the process of obtaining joint angles from the coordinates of the point where the end effector is located [10]. Inverse kinematics is a very difficult issue, especially in serial manipulators, and as the number of joints increases, the nonlinearity of the problem increases [11]. Due to these properties of the problem, it is a problem type that is very suitable for the use of heuristic algorithms. Therefore, these two have been studied extensively in recent studies [12]. Raja et al. created a learning interface for the inverse kinematics solution of a 14-joint redundant mobile robot manipulator. In this way, the manipulator and the mobile platform have been constantly connected and carried out the work-flow smoothly [13]. He has solved the inverse kinematics problem of a 6-DOF offset wrist industrial robot arm using the vortex search algorithm based on chaos. The algorithm reaches good values in high iterations in its normal operation. However, he managed to speed up the algorithm by obtaining good values in lower iterations by using that chaos structure [14]. Ram et al.

designed a mobile manipulator and used the bidirectional search algorithm to solve the inverse kinematics problem that will occur according to the changing position of this manipulator [15]. Dereli and Köker solved the inverse kinematics problem of the 7-joint redundant robot manipulator with both the artificial bee colony algorithm [16] and the firefly algorithm. At the same time, they analyzed the effects of the number of fireflies on the solution and the computation time [17]. Zhang and Xiao modified the algorithm by applying a chaotic system to the distribution of bees leaving the hive to search for food in the artificial bee colony. In this way, they managed to reduce the error rate considerably in the inverse kinematics problem of the 7-dof robotic manipulator [18]. They developed the artificial bee colony algorithm by using a different parameter in the process of updating the food sources in this algorithm. With this new technique called K-ABC, they performed the inverse kinematics calculation of a 5-joint robot manipulator [19].

In this study, the inverse kinematics calculation of a 7-DOF redundant robot manipulator was performed with the grey wolf colony algorithm, which is actual swarm algorithm. First of all, DH parameters of the robot manipulator were obtained. Afterward, a fitness function based on the Euclidean distance equation was formed by using position equations. By means of this function, the algorithm optimizes the values. In Chapter 3, the algorithm for the grey wolf colony, both traditional and modified within the scope of this study, is explained in detail. The part where the results and these results are analyzed comprehensively and the algorithm is compared with other swarm algorithms is Chapter 4.

2 Problem definition and fitness function

Control of a robot begins with controlling its movements. Robot motions are modeled in two ways, linear and more easily calculable, forward kinematics and nonlinear and more complex inverse kinematics [20]. As seen in Fig. 1, the position of the end element of the robot manipulator in the working space is obtained with forward kinematics. Here, the joint angles are known, replaced in the relevant equation and results are obtained easily [21]. However, in inverse kinematics the end-effector values are known. Since this situation requires dealing with complex mathematical equations, its calculation is extremely difficult [22]. The problem dealt with in this study is aimed at the solution of the inverse kinematics problem, which is the most fundamental problem of robot control.

2.1 7-DOF robotic manipulator

The robot manipulator used in this study is a redundant industrial manipulator with 7-rotational joints. The most important feature of 7-joint manipulators is that they can avoid obstacles, be positioned flexibly in the workspace and have different number of kinematic solutions [23–25] (Fig. 2).

Different techniques are used in the literature for inverse kinematics calculation. In this study, calculations based on homogeneous transformation matrices are carried out with DH parameters introduced to the literature by Denavit and Hartenberg [26]. These parameters are four, namely “a” indicates the lengths of the limbs connected to each other, “d” indicates the sliding dimension of the center of one limb relative to the center of another limb, “alpha” indicates the sliding angle, and “theta” indicates the joint angles [27].

In Table 1, “i” refers to the joint order. The unit of “a” and “d” is meter, and the unit of “α” and “Θ” is degrees. The general transformation matrix to be used for kinematic calculations appears below (Eq. 1).

$$ {}_{i-1}^{\;\;i}T = \begin{bmatrix} \cos\theta i & -\cos\alpha i\cdot\sin\theta i & \sin\alpha i\cdot\sin\theta i & ai\cdot\cos\theta i \\ \sin\theta i & \cos\alpha i\cdot\cos\theta i & -\cos\theta i\cdot\sin\alpha i & ai\cdot\sin\theta i \\ 0 & \sin\alpha i & \cos\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

The equation to find the position and orientation of the robot manipulator end effector from the above general matrix is as follows (Eq. 2):

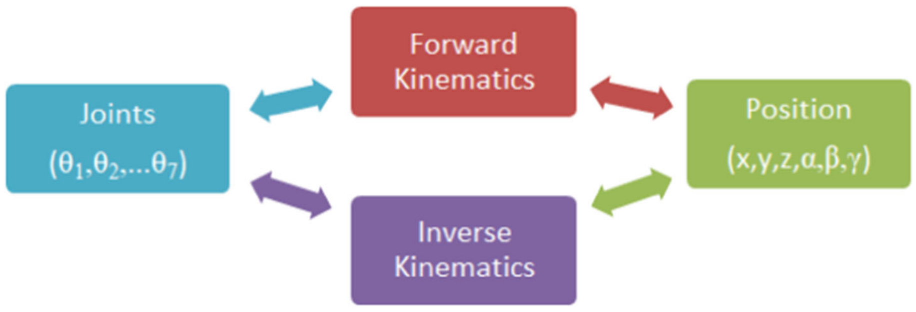


Fig. 1 Kinematics in the robotic

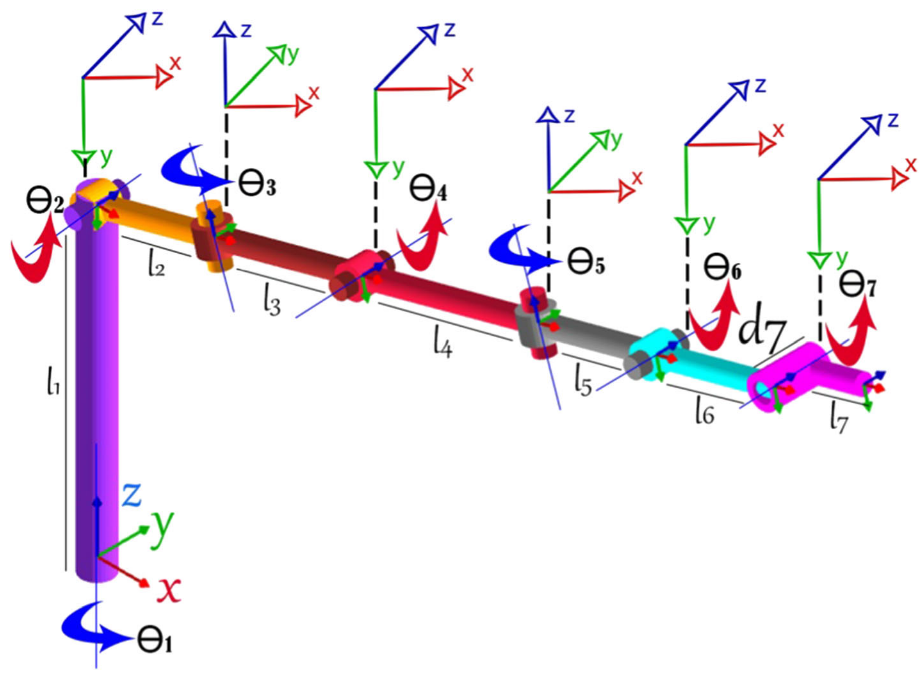


Fig. 2 The structure of 7-DOF robot manipulator

Table 1 DH parameters of the 7-DOF robot manipulator

i	a_i(m)	α_i	d_i(m)	Θ_i (°)(Range)
1	0	− 90	l_1 = 0.5	$-180 < \Theta_1 < 180$
2	l_2 = 0,2	90	0	$-90 < \Theta_2 < 30$
3	l_3 = 0,25	− 90	0	$-90 < \Theta_3 < 120$
4	l_4 = 0,3	90	0	$-90 < \Theta_4 < 90$
5	l_5 = 0,2	− 90	0	$-90 < \Theta_5 < 90$
6	l_6 = 0,2	0	0	$-90 < \Theta_6 < 90$
7	l_7 = 0,1	0	d_7 = 0.05	$-30 < Q_7 < 90$

$$A_{\text{End-Effector}} = {}^{7}_{0}T = {}^{1}_{0}T.{}^{2}_{1}T.{}^{3}_{2}T.{}^{4}_{3}T.{}^{5}_{4}T.{}^{6}_{5}T.{}^{7}_{6}T = \begin{bmatrix} n_x & s_x & a_x & P_x \\ n_y & s_y & a_y & P_y \\ n_z & s_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

${}^{i}T_{i+1}$, i. the transfer matrix of the limb and ${}^{0}T_7$ matrix refers to the transformation matrix between the main frame and the end effector. Each transformation matrix consists of a 3 × 3 rotation matrix (n_x, n_y, n_z, s_x, s_y, s_z, a_x, a_y, a_z) and a 3 × 1 position vector (P_x, P_y ve P_z). The position equations of the robot arm used for this study are given Eq. 3. In these equations, "s" is expressed as sine and "c" is expressed as cosine.

$$\begin{aligned}
P_x = {} & (c\theta_1 c\theta_2 c\theta_3 c\theta_4 - s\theta_1 s\theta_3 s\theta_4 - c\theta_1 s\theta_2 s\theta_4) \\
& (c\theta_5 c\theta_6 l_7 c\theta_7 - c\theta_5 s\theta_6 l_7 s\theta_7 - s\theta_5 d_7 + c\theta_5 l_6 c\theta_6 + l_5 c\theta_5) \\
& + (-c\theta_1 c\theta_2 s\theta_3 - s\theta_1 c\theta_3) \\
& (s\theta_5 c\theta_6 l_7 c\theta_7 - s\theta_5 s\theta_6 l_7 s\theta_7 + c\theta_5 d_7 + s\theta_5 c\theta_6 l_6 + l_5 s\theta_5) \\
& + (c\theta_1 c\theta_2 c\theta_3 s\theta_4 - s\theta_1 s\theta_3 s\theta_4 + c\theta_1 c\theta_4 s\theta_2) \\
& (-s\theta_6 l_7 c\theta_7 - c\theta_6 l_7 s\theta_7 - l_6 s\theta_6) \\
& + c\theta_1 c\theta_2 (c\theta_3 c\theta_4 l_4 + l_3 c\theta_3) - s\theta_1 (s\theta_3 c\theta_4 l_4 + l_3 s\theta_3) \\
& - c\theta_1 s\theta_2 l_4 s\theta_4 + c\theta_1 c\theta_2 l_2 \\
P_y = {} & s\theta_1 c\theta_2 c\theta_3 c\theta_4 + c\theta_1 s\theta_3 c\theta_4 - s\theta_1 s\theta_2 s\theta_4) \\
& (c\theta_5 c\theta_6 l_7 c\theta_7 - c\theta_5 s\theta_6 l_7 s\theta_7 - s\theta_5 d_7 + c\theta_5 c\theta_6 l_6 \\
& + l_5 c\theta_5) + (-s\theta_1 c\theta_2 s\theta_3 + c\theta_1 c\theta_3) \\
& (s\theta_5 c\theta_6 l_7 c\theta_7 - s\theta_5 s\theta_6 l_7 s\theta_7 + c\theta_5 d_7 + s\theta_5 c\theta_6 l_6 + l_5 s\theta_5) \\
& + (s\theta_1 c\theta_2 c\theta_3 s\theta_4 + c\theta_1 s\theta_3 s\theta_4 + s\theta_1 s\theta_2 c\theta_4) \\
& (-s\theta_6 l_7 c\theta_7 - c\theta_6 l_7 s\theta_7 - l_6 s\theta_6) + s\theta_1 c\theta_2 (c\theta_3 c\theta_4 l_4 + l_3 c\theta_3) \\
& + c\theta_1 (s\theta_3 c\theta_4 l_4 + l_3 s\theta_3) - s\theta_1 s\theta_2 s\theta_4 l_4 + s\theta_1 c\theta_2 l_2 \\
P_z = {} & (-s\theta_2 c\theta_3 c\theta_4 - c\theta_2 s\theta_4) \\
& (c\theta_5 c\theta_6 l_7 c\theta_7 - c\theta_5 s\theta_6 l_7 s\theta_7 - s\theta_5 d_7 + c\theta_5 c\theta_6 l_6 + l_5 c\theta_5) \\
& + s\theta_2 s\theta_3 (s\theta_5 c\theta_6 l_7 c\theta_7 - s\theta_5 s\theta_6 l_7 s\theta_7 + c\theta_5 d_7 + s\theta_5 c\theta_6 l_6 + s\theta_5 l_5) \\
& + (-s\theta_2 c\theta_3 s\theta_4 + c\theta_2 c\theta_4)(-s\theta_6 l_7 c\theta_7 - c\theta_6 l_7 s\theta_7 - s\theta_6 l_6) \\
& - s\theta_2 (c\theta_3 c\theta_4 l_4 + l_3 c\theta_3) \\
& - c\theta_2 s\theta_4 l_4 - s\theta_2 l_2 + l_1
\end{aligned} \quad (3)$$

In Eq. 3, the process of writing the angles values and obtaining the x, y and z values is an advanced kinematics problem, which is extremely easy to solve. However, obtaining each angle value by writing instead of x, y and z values is known as the inverse kinematics problem, which is extremely difficult because it requires quite complex mathematical transformations.

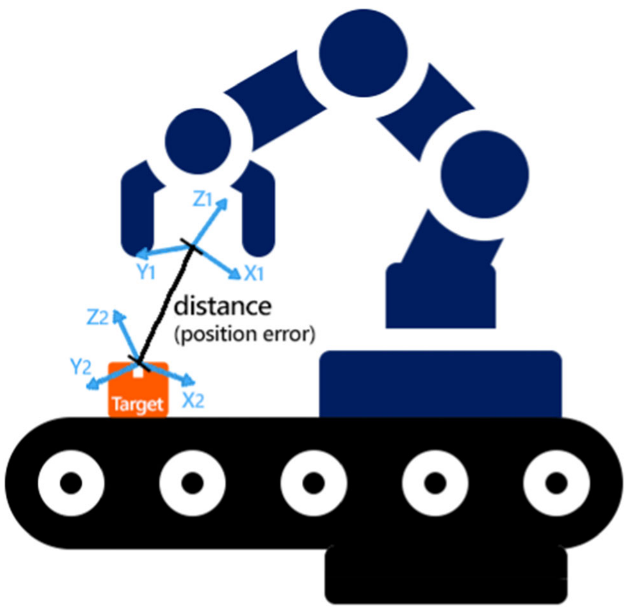


Fig. 3 Illustration of the position error (distance between target and end effector)

2.2 Fitness function

The fitness function consists of an equation that heuristic optimization algorithms use to compare the values they obtain, and it is one of the most important phases of the algorithm. In this way, the algorithm can continuously update the best value [28]. There are many fitness functions used for this purpose by researchers in literature [29–31]. In this study, Euclidean distance, which is frequently used to find the shortest distance between two points [32], is used as a fitness function.

In this study, the closest position to the predetermined target point has been obtained with the grey wolf colony algorithm. Figure 3 clearly illustrates this situation. The 3-dimensional Euclidean distance is used to find the smallest position error value (Eq. 4).

$$error = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{4}$$

As can be seen in Fig. 3, (x_1, y_1, z_1) is the position of the end effector obtained by the heuristic algorithm and (x_2, y_2, z_2) is the position of the target point.

3 Methods

The robot manipulator used in this study has a complex structure since it has an excessive number of joints and its inverse kinematic equations are nonlinear. For this reason, this problem in robotics can be easily solved by heuristic methods aiming to obtain the best values by making a random search in a certain solution space. In this study, with the grey wolf swarm algorithm developed for this purpose, modified grey wolf colony algorithms developed by improving the parameters that prevent it from being stuck to the local minimum were used and the results were compared comprehensively in the next section.

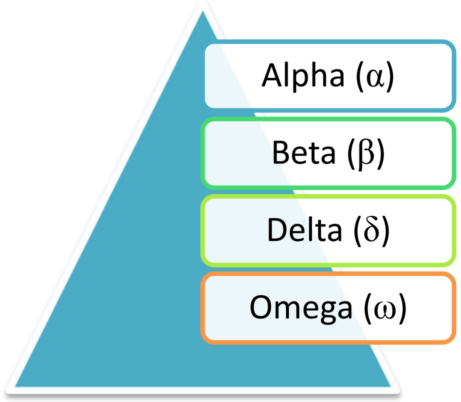


Fig. 4 Hierarchy of grey wolfs

3.1 An overview GWO algorithm

This algorithm, proposed by Mirjalili in 2014, imitates the hunting and survival skills of grey wolves in nature. The grey wolves mentioned in this algorithm belong to the Canis Lupus species and are included in the Canidae family. Grey wolves prefer to live in swarms of 5–12 individuals [33]. In addition, a different social hierarchy including alpha, beta, delta and omega is followed in the swarm. Figure 4 shows the swarm hierarchy according to the dominance of grey wolves. The most dominant feature is the alpha and then beta wolf, while the omega wolves do not have any dominant feature [34].

The leader of the swarm is alpha, and its assistant is beta. So, if the alpha wolf dies or ages, it is the closest beta wolf to replace it. Decisions such as the progression, hunting and resting of the swarm are determined by the leader wolf alpha. The omega wolves below the social hierarchy are wolves whose executing orders follow the swarm leader and are led by other wolves such as alpha and beta [35]. The other wolf group that can dominate the wolves in this group is the delta. Delta wolves track the prey, serve as a guard to protect the swarm, assist the alpha and beta wolf in hunting, provide food for the herd, and take care of sick, wounded and weak wolves [36].

The hunting and catching of grey wolves is a complete optimization process, and for this purpose, they follow the following algorithm:

As can be seen in Fig. 5, the hunting of grey wolves takes place in three stages from the moment they decide to hunt. These are expressed as observation, surrounding and

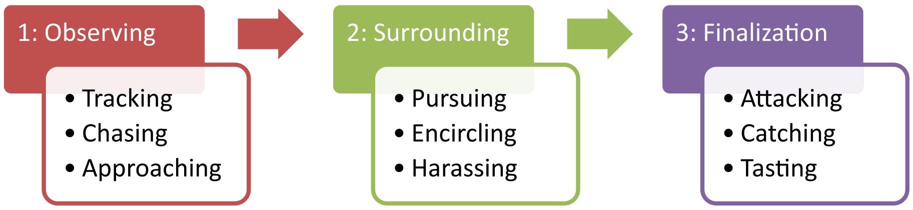


Fig. 5 Hunting algorithm of grey wolfs

finalization. In the first stage, the prey is followed from a distance and then closely. Then, when the prey comes to a suitable environment, it is quite approached. In the second stage, the prey is surrounded so that it can move nowhere and have nowhere to run. In the last stage, the hunt is close to the end. At this stage, the grey wolf is looking for the best time to hunt and the weakest moment of the hunt. When he has this moment, he completes the process by catching the prey with other wolves [37, 38].

The grey wolf algorithm is a frequently recommended herd-based heuristic optimization method recently. Because, compared to other classical techniques, this algorithm has very important advantages such as having fewer parameters, reaching the solution with high accuracy, powerful exploration and fast convergence [39]. For this reason, the grey wolf algorithm has been used frequently in various fields in the last five years and its advantages and deficiencies compared to other algorithms have been revealed. Sharma et al. diagnosing Parkinson's disease [40], Pradhan et al. in the field of power electronics [41], Natesan et al. in their work on task sharing [42], Khandelwal et al. work on the expansion of transmission networks [43], Kalemci et al. in a new retaining wall design on the construction site [44]; they used this algorithm and obtained effective results.

It is witnessed that the grey wolf swarm algorithm is used in the robotic field on the basis of robot control. Rahmani et al. used the grey wolf algorithm to control a 2-joint robot manipulator [45], while others used it for mobile robot control [46]. In this study, again, an optimization process based on robot control was carried out. For all these operations, grey wolf swarm behaviors were modeled mathematically as depicted in Fig. 6.

According to the algorithm, the most appropriate solutions to the predicted location of the prey are firstly the alpha (α), then the beta (β) and finally the delta (δ) wolves. Omega (ω) wolves consist of grey wolves that are candidates to be alpha, beta or delta. Therefore, in the whole algorithm process, the position of the omega wolf is updated gradually until it reaches the optimal value. Each current value of the omega wolf is compared with the value of alpha, beta and delta wolves, respectively. When the comparison result is positive, the wolves change places and the new value is updated. As a result, the first best value obtained according to the algorithm belongs to alpha, the second best value belongs to beta, and the third best value belongs to delta [47].

When prey is found, omega wolves pursue the prey by means of alpha, beta and delta wolves and encircle it at the appropriate time. This situation is modeled in Eq. 5.

$$\begin{aligned} D_\alpha &= |C_1 * X_\alpha - X_t| \\ D_\beta &= |C_2 * X_\beta - X_t| \\ D_\delta &= |C_3 * X_\delta - X_t| \end{aligned} \tag{5}$$

In Eq. 5, "t" iteration and X_t denote the omega wolf in the "i" iteration. "D" coefficient is the vector that shows the distance of wolves (alpha, beta and delta) from prey. X_α, X_β and X_δ show the current positions of alpha, beta and delta wolves, respectively. "C" is a random coefficient value calculated in Eq. 8.

$$\begin{aligned} X_1 &= |X_\alpha - A_1 * D_\alpha| \\ X_2 &= |X_\beta - A_2 * D_\beta| \\ X_3 &= |X_\delta - A_3 * D_\delta| \end{aligned} \tag{6}$$

In Eq. 6, the parameters X_1, X_2 and X_3 show the positional relationship between alpha, beta and delta wolves and other grey wolves, respectively. This means that grey wolves update their positions based on alpha, beta and delta wolves.

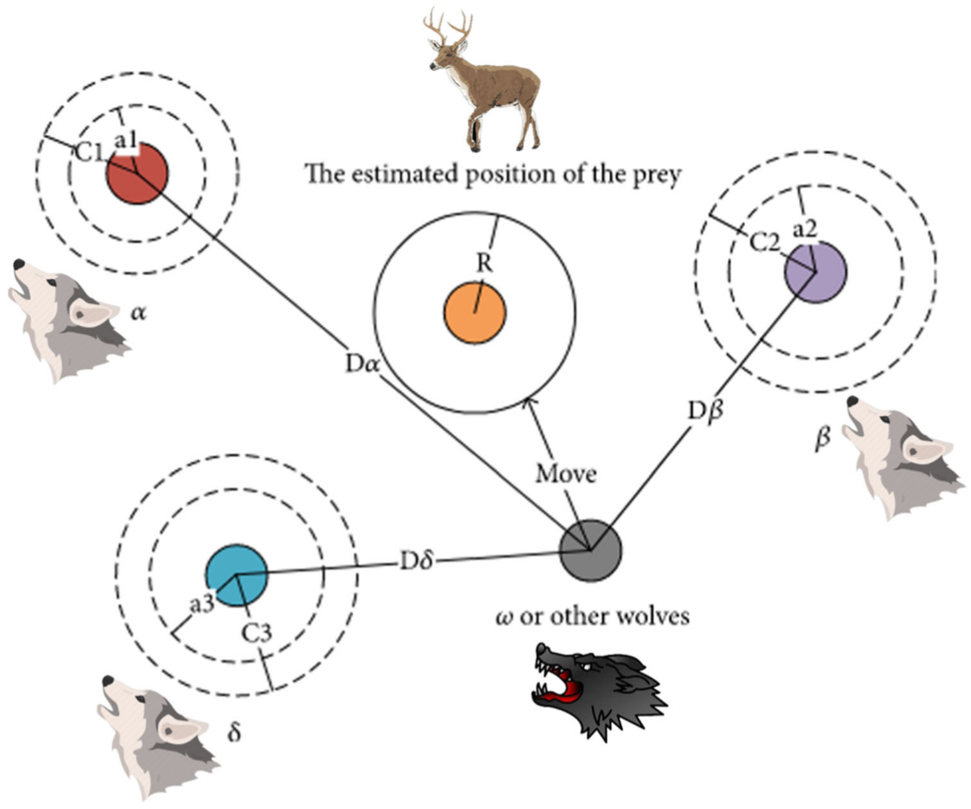
$$X_{t+1} = \frac{X_1 + X_2 + X_3}{3} \tag{7}$$

X_{t+1} calculated in Eq. 7 now represents the new position of the grey wolf. This process is the last process to update the positions of grey wolves except for alpha, beta and delta wolves. After that, the best values for alpha, beta and delta wolves will be determined from these positions.

$$\begin{aligned} A &= 2 * a * r_1 - a \\ C &= 2 * r_2 \\ a &= 2 * \left(1 - \frac{t}{t_{\max}}\right) \end{aligned} \tag{8}$$

The coefficients in this equation are known as the control parameters of the algorithm and are only three. In every heuristic algorithm, there are random situations as in real life. In this algorithm, the coefficients A and C shown in Eq. 8 are the values calculated in this way. "*a*" takes the value "2" at the beginning and approaches "0" step by step in each iteration. This decrease takes a linear course.

Fig. 6 Position updating in GWO [39]



Depending on the values of “a,” the “A” parameter takes random values in the range of [-a, a]. The value taken by A can decide whether the grey wolf will take the attack position or not. As a result of the calculation, if the $|A < 1|$ status is present, the grey wolf is extremely close to the hunt and can attack at any time. In the case of $|A > 1|$, the grey wolf moves away from prey in hopes of finding new prey [48].

C, another important control parameter, is accepted as the exploration component of the algorithm and can take random values in the range [0, 2]. This parameter contributes to the algorithm to exhibit a more random behavior and thus prevents the optimization from stuck to local optimum values. This situation occurs if $|C > 1|$, otherwise, that is, in case of $|C < 1|$, the state of displaying random behaviors decreases [49].

The flowchart of the algorithm is shown in Fig. 7 and, like other heuristic techniques, it consists of three basic steps: initializing, computation of fitness function, position updates of all swarm members and obtaining the best value. The optimization process starts with assigning all control parameters to the initial value, and all grey wolves receive random values at certain intervals. In the next step, the fitness function is calculated based on the initial values and the best solutions are determined as alpha, beta and delta wolves, respectively. Now the next stage is the position updates for all grey wolves except alpha, beta and delta wolves. Now the next stage is the position updates of all grey wolves and then alpha, beta and delta wolves, and finally, the values of the control parameters are updated. Finally, the alpha wolf’s best position value is returned. In terms of solving the inverse kinematics problem, the best joint angles are obtained.

3.2 Modified grey wolf optimization algorithm

Due to the small number of control parameters, Mirjalili particularly found that the GWO algorithm tends to stay stuck at local optimal values when used in its simplest form [33]. Therefore, the researchers modified the GWO by adding new ones to the control parameters or by changing the value range of the control parameters. They changed Eq. 7 based on the fact that the alpha wolf was dominant than the beta and delta wolf in the search process. In this way, it really gets better results in tests [50]. For this purpose, it is possible to find many studies in the literature in which the grey wolf algorithm has been modified and developed in different fields. They changed Eq. 7 based on the fact that the alpha wolf was dominant than the beta and delta wolf in the search process. In this way, it really gets better results in tests [50]. They developed this algorithm by adding machine learning feature to the grey wolf algorithm in order to diagnose Parkinson’s disease [40]. They have changed the GWO to produce better results by

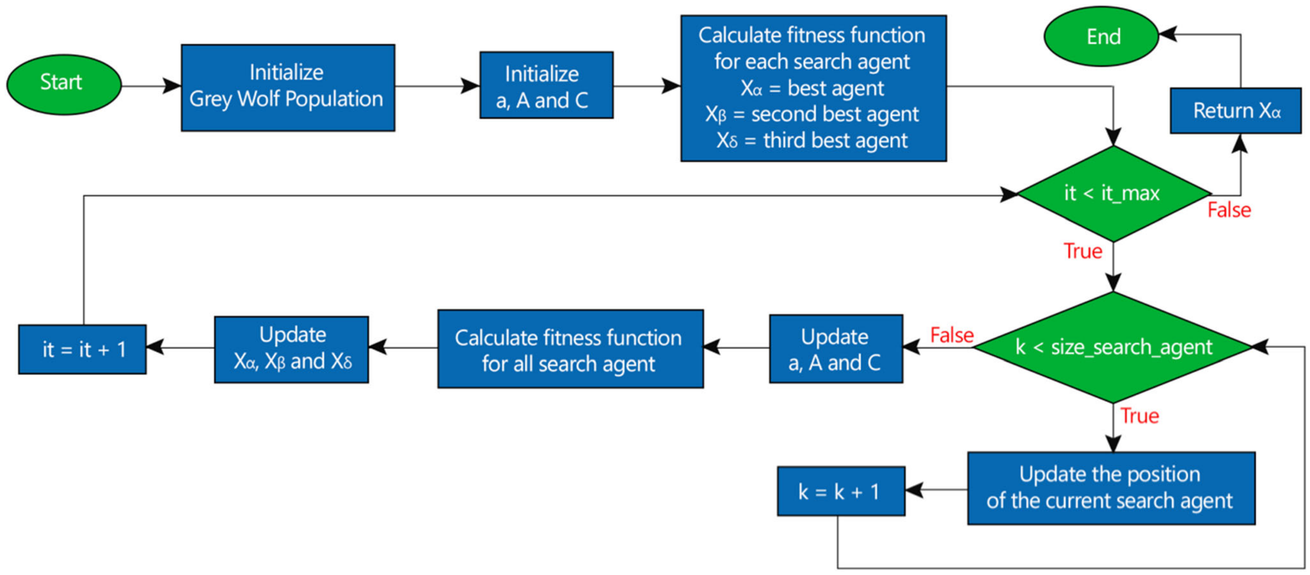

Start
Initialize Grey Wolf Population
Initialize a, A and C
Calculate fitness function for each search agent Xα = best agent Xβ = second best agent Xδ = third best agent
End
Return Xα
it < it_max
False
True
k < size_search_agent
False
True
Update a, A and C
Calculate fitness function for all search agent
Update Xα, Xβ and Xδ
it = it + 1
Update the position of the current search agent
k = k + 1

Fig. 7 Flow chart of the GWO

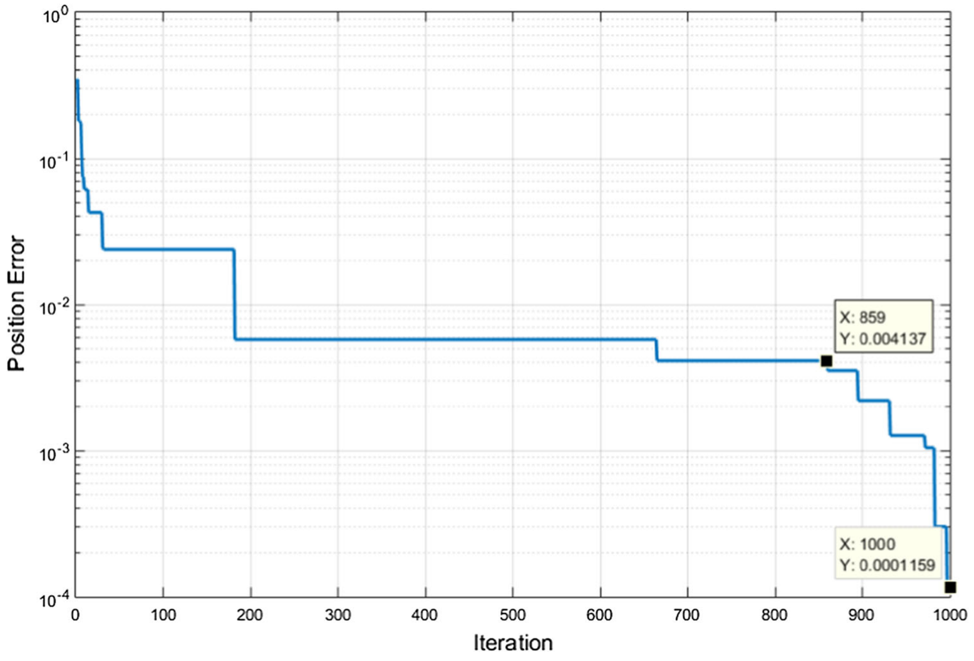


Fig. 8 An example figure for inverse kinematics solution with conventional GWO

including a Levy flight-based operator in the algorithm's calculation processes and tested the results in some benchmarks [51].

Figure 8 shows the values obtained in each iteration of an example calculation made within the scope of this study, and similar results were obtained in other simulations. It is clearly seen in this graph as in other graphs; attempts to find better value toward the end of the algorithm are quite high. For example, in this graph, many new values were obtained since the 860th iteration and this situation continued until the last iteration.

Starting from the idea that such a result is directly related to the "a" parameter in the GWO algorithm, the calculation method of the "a" parameter in the algorithm has been changed in this study. Because, the parameter "a" decreases from 2 to 0 linearly and as this parameter approaches zero, the values obtained with the search process improve. Therefore, the more searches the "a" parameter performs in a position close to zero, the more it improves its values in the algorithm.

In this study, the algorithm has been incredibly strengthened by changing the way in which the "a"

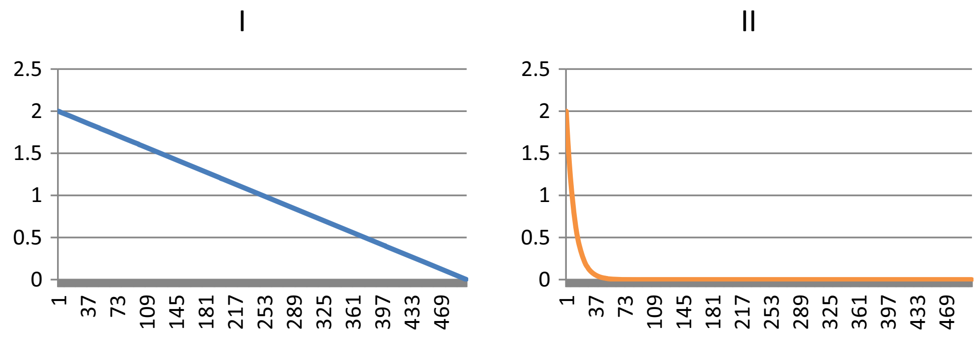


Fig. 9 "a" parameter in traditional GWO (I) and modified GWO (FPD-GWO) (II)

parameter is calculated without including an additional parameter to the algorithm. While the parameter "a" is calculated in the traditional GWO as seen in Eq. 8, it is modified in this study and calculated as in Eq. 9.

$$a = 2 * \left(e^{-\frac{t*s}{t_{max}}}\right) \tag{9}$$

As seen in Eq. 9, only "s" value is added to the "a" parameter and it represents the number of agents in the swarm. As is known, in traditional GWO, the "a" parameter decreases linearly, preventing the algorithm from sticking to local minimum values. In the analysis made in this study, it was observed that as the parameter "a" approaches zero, it not only saves the algorithm from the local minimum, but also contributes greatly to the strengthening of the algorithm. For this reason, the faster this parameter goes from 2 to 0, the more the algorithm converges to the best values. For this reason, the algorithm has approached from 2 to 0 very quickly and parabolically as in Fig. 9. This led to the name of the new technique being called Fast Parabolic Descending GWO (FPD-GWO).

Figure 9 shows the values that parameter "a" achieves at each iteration in traditional GWO and modified GWO. In this figure, the "a" parameter has decreased linearly in the graph no I and approached zero toward the end of the iteration. However, as seen in the graph no II, it is very close to zero at the beginning of the iteration.

4 Results

In this study, the traditional grey wolf algorithm has been modified for the inverse kinematics calculation of a 7-joint serial robot manipulator to achieve better results. For this purpose, the calculation method of the "a" parameter, which is also used in the algorithm to prevent stuck in local minimum values, has been changed. The "a" parameter, which decreases linearly from 2 to 0 in traditional GWO, is rapidly reduced parabolically within the scope of this study. In this way, the results obtained with traditional GWO are improved between 10^9 and 10^{12} times.

Figure 10 shows the behavior of both the traditional GWO and the FPD-GWO developed within the scope of this study at different iteration values. In this figure, it is observed that the traditional GWO intensifies the search process toward the end of the iteration, while FPD-GWO finishes the search process in the middle of the iteration and achieves the best value in both. Therefore, it is obvious that the method developed within the scope of this study works much more stable than the traditional GWO and produces much better results.

In order to demonstrate the accuracy of the algorithms, both methods were run 50 times consecutively and the results obtained are shown in Fig. 11. Although both methods have a stable structure, the big difference in terms of the results obtained is clearly seen in this figure.

The figure showing the comparison of the best, worst and average values among 50 data obtained in Fig. 11 is given in Fig. 12. These values are very close to each other in both traditional GWO and FPD-GWO. Even this situation shows that the methods available are extremely stable.

$$F_1 = \left(x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10 \tag{10}$$

$$F_2 = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \left[30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right] \tag{11}$$

$$F_3 = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right) \tag{12}$$

$$F_4 = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right) \tag{13}$$

After the proposed new method shows successful results in solving the inverse kinematics problem, benchmark functions are used for verification. The functions used are shown in Eqs. 10, 11, 12 and 13. Although they are not as

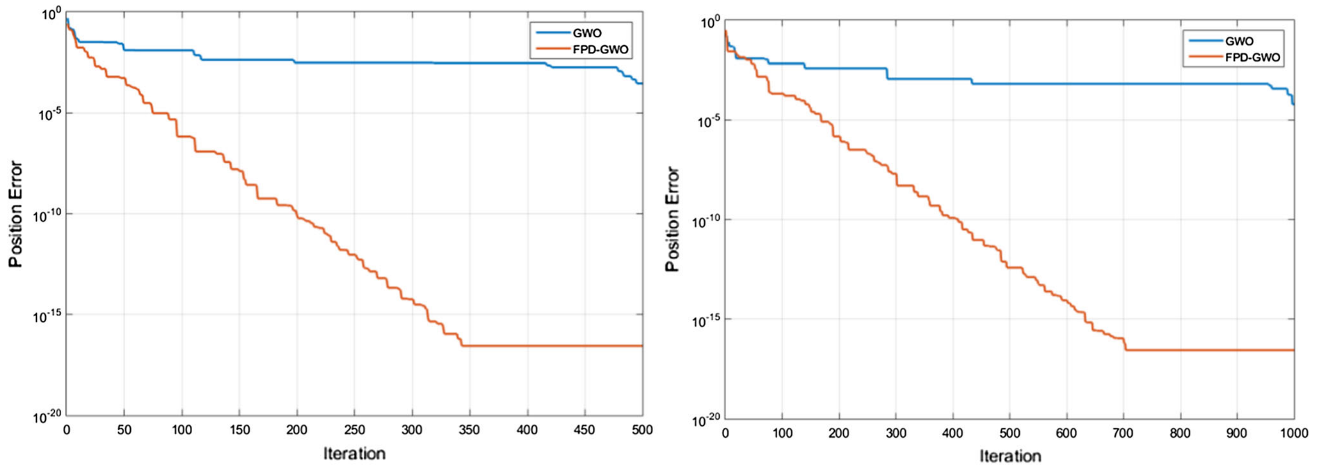


Fig. 10 Algorithm behavior with different iteration numbers

Fig. 11 Data obtained from 50 consecutive runs

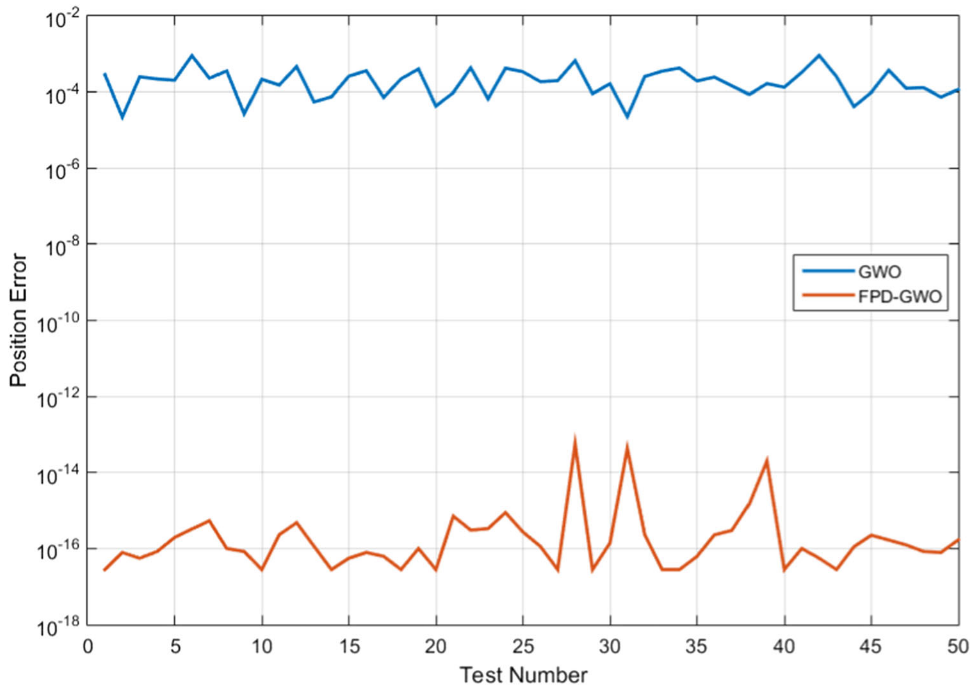


Fig. 12 Comparison of both calculation methods in terms of best, worst and average value

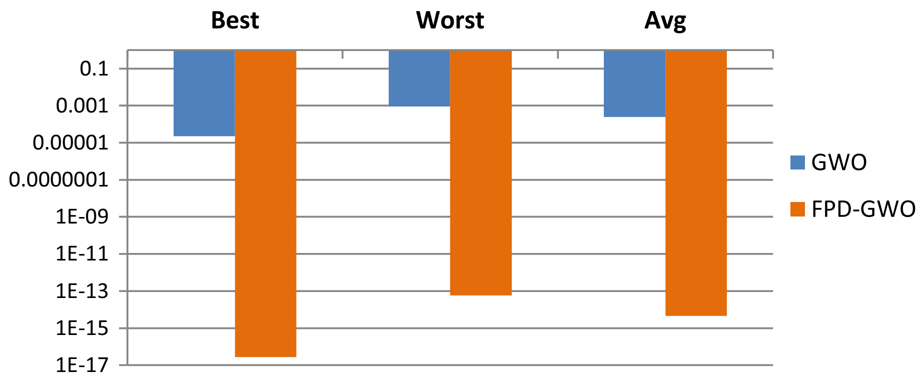


Table 2 Comparative results of all the benchmark functions

Benchmark	Dim	Range	fmin	GWO	FPD-GWO
F_1	2	[− 5, 5]	0.398	0.39789	0.39789
F_2	2	[− 2, 2]	3	3	3
F_3	3	[1, 3]	− 3.86	− 3.8549	3.8614
F_4	6	[0, 1]	− 3.32	− 3.201	− 3.321

complex as the problem addressed in this study, they are sufficient to verify the proposed technique.

Table 2 shows the dimension, limit values, the best value and the minimum values obtained by GWO and FPD-GWO techniques related to the benchmark functions used. Considering that as the number of dimensions increases in functions, the complexity of the function increases, it is clear that the success of the proposed technique stands out more.

5 Discussion

With the new technique proposed in this study, the search capability in the grey wolf colony algorithm has been strengthened and the probability of being stuck to local minimum values is considerably reduced. In fact, the most important handicap of all heuristic algorithms available in the literature is the situation of very frequent fall in to local minimum values. Therefore, the less the heuristic algorithm falls to local minimum values, the higher its ability to produce successful results. When evaluated from this point of view, this new technique proposed in terms of the results it obtained gets ahead of many heuristic algorithms in this sense (Fig. 13).

Figure 13 shows the comparative results of some heuristic algorithms used in inverse kinematics calculation, which is one of the most fundamental problems in robotics. These heuristic algorithms are known as techniques that are used extensively to solve frequently encountered problems in the literature. In 2004 Quantum PSO (QPSO) [52], in 2008 firefly (FA) [5], in 2016 whale optimization algorithm (WOA) [53] and in 2005 artificial bee colony (ABC) [54] are introduced and compared with the new technique suggested in Fig. 13. In the comparison, all algorithms were run with 50 population sizes and 500 iterations. The results clearly demonstrate the extraordinary success of the proposed new method compared to other heuristic techniques. Similarly, one of the important conclusions to be drawn from this figure is that algorithms with improved search capability can update their values with a better value in each iteration. It is clear that QPSO and FPD-GWO have better search capabilities in this respect. Algorithms such as ABC, FA, GWO and WOA whose curves resemble a relatively straight line could not obtain a better up-to-date value because their search capabilities were weak.

Figure 14 shows the number of exceeding the limit values of joint angles obtained through both algorithms. During the optimization process performed with GWO, it is obvious that limit exceeds are extraordinarily high, and on the contrary, the situation remains at an extremely low level in FPD-GWO. This situation gives the impression that the variables are kept under more control in the FPD-

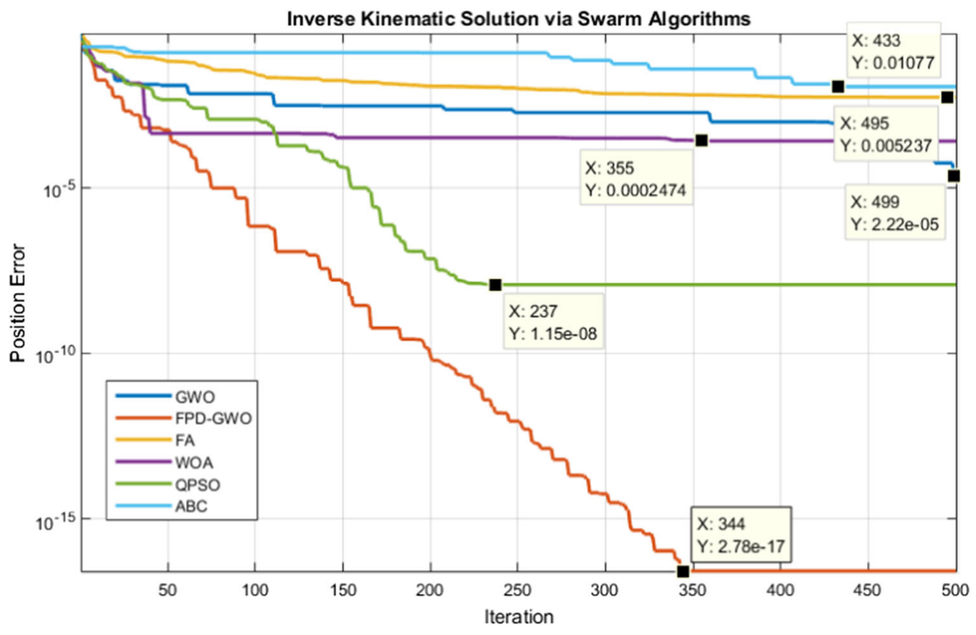
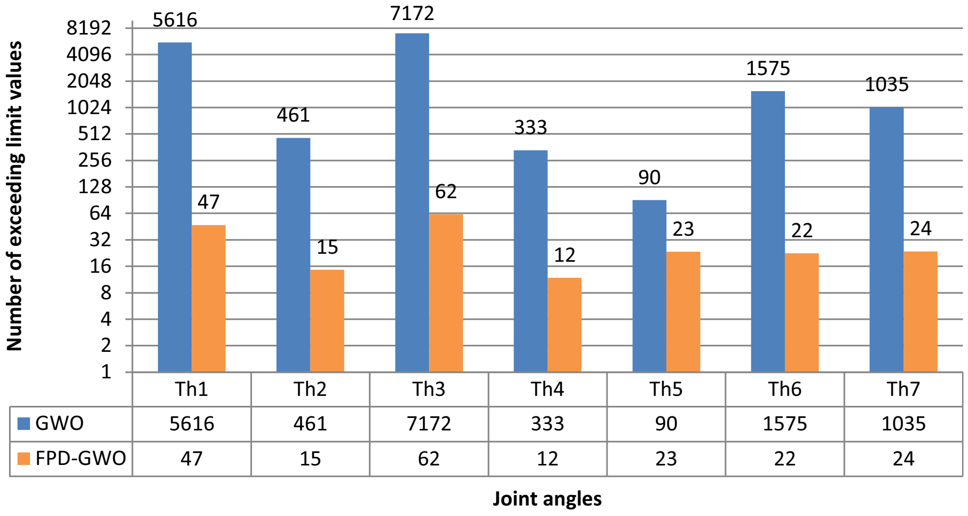


Fig. 13 Comparison of the proposed technique with other swarm algorithms (population size = 50)



	Th1	Th2	Th3	Th4	Th5	Th6	Th7
GWO	5616	461	7172	333	90	1575	1035
FPD-GWO	47	15	62	12	23	22	24

Fig. 14 Limit value of joint angles throughout the algorithm

Table 3 Comparative results of all the algorithms

Algorithms	Swarm size	MSE (Position error)
PSO	300	2.1162e−04
ABC	100	1.1105e−06
Firefly	50	1.4547e−05
Quantum PSO	150	6.9347e−13
GWO	50	9.4745e−08
FPD-GWO	50	1.1169e−28

GWO technique and the convergence to the target occurs more consistently.

Table 3 shows the comparison of the results obtained by GWO and FPD-GWO in the inverse kinematic solution of a 7-joint robot manipulator with some other swarm algorithms. It is clear from the results that the FPD-GWO technique developed within the scope of this study gives much better results than other swarm algorithms [10].

Since the technique proposed in this study is based on developing a swarm algorithm, it can be used to solve any engineering problem suitable for this technique. Because there are two important evidences that it can be used in other engineering or industry problems. The first of these has shown an extraordinary success in solving a very complex problem with a large number of variables. The other has been verified by four benchmark functions with different number of variables.

6 Conclusions

In this study, the inverse kinematics problem of a 7-joint robot manipulator has been solved using GWO that is one the current swarm algorithms and a new technique consisting of modified GWO. The results obtained clearly show that the grey wolf optimization algorithm produces effective results in the solution of nonlinear problems, and is stable and consistent in terms of the results produced. Nevertheless, the large differences between the results obtained with both techniques clearly indicate that the grey wolf optimization algorithm is frequently stuck at local minimum values. The new technique obtained by modifying the GWO and referred to as FPD-GWO has a stable structure just like GWO, but has produced much better results than GWO. In addition, this new technique is very easy to use because it is obtained by changing the calculation method of a single parameter in GWO.

Declarations

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

References

1. Yang XS (2014) Swarm intelligence based algorithms: a critical analysis. Evol Intel 7:17–28
2. Dorigo M, Maniezzo V, Colorni A (1991) Distributed optimization by ant colonies. In: European conference on artificial life, vol 134, p 142
3. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: International conference on neural networks, pp 1942–1948
4. Karaboga D, Basturk B (2008) On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput 8:687–697
5. Yang XS (2013) Multiobjective firefly algorithm for continuous optimization. Eng Comput 29:175–184
6. Ribeiro JM, et al (2017) Comparison of PID controller tuning methods: analytical/classical techniques versus optimization algorithms. In: IEEE 18th international Carpathian control conference (ICCC), pp 533–538

7. Doma MI (2013) Particle swarm optimization in comparison with classical optimization for GPS network design. J Geod Sci 3:250–257
8. Yang XS, Deb S (2014) Cuckoo search: recent advances and applications. Neural Comput Appl 24(1):169–174
9. Iakovlev R, Denisov A, Prakapovich R (2020) Iterative method for solving the inverse kinematics problem of multi-link robotic systems with rotational joints. In: International conference on electromechanics and robotics, pp 237–251
10. Dereli S, Köker R (2020) A meta-heuristic proposal for inverse kinematics solution of 7-DOF serial robotic manipulator: quantum behaved particle swarm algorithm. Artif Intell Rev 53:949–964
11. Ren H, Ben-Tzvi P (2020) Learning inverse kinematics and dynamics of a robotic manipulator using generative adversarial networks. Robot Autonom Syst 124:1
12. Bai L et al (2019) Appl Sci 9:546
13. Rajaa R, Dutta A, Dasgupta B (2019) Learning framework for inverse kinematics of a highly redundant mobile manipulator. Robot Autonom Syst 120:103245. https://doi.org/10.1016/j.robot.2019.07.015
14. Toz M (2020) Chaos-based Vortex Search algorithm for solving inverse kinematics problem of serial robot manipulators with offset wrist. Appl Soft Comput 89:106074. https://doi.org/10.1016/j.asoc.2020.106074
15. Ram RV, Pathak PM, Junco SJ (2019) Inverse kinematics of mobile manipulator using bidirectional particle swarm optimization by manipulator decoupling. Mech Mach Theory 131:385–405
16. Dereli S, Köker R (2020) Simulation based calculation of the inverse kinematics solution of 7-DOF robot manipulator using artificial bee colony algorithm. SN Appl Sci 2(1):1–11
17. Dereli S, Köker R (2020) Calculation of the inverse kinematics solution of the 7-DOF redundant robot manipulator by the firefly algorithm and statistical analysis of the results in terms of speed and accuracy. Inverse Problem Sci Eng 28:601–613
18. Zhang L, Xiao N (2019) A novel artificial bee colony algorithm for inverse kinematics calculation of 7-DOF serial manipulators. Soft Comput 23:3269–3277
19. El-Sherbiny A, Elhosseini MA, Haikal AY (2018) A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm. Appl Soft Comput 73:24–38
20. Pérez L et al (2019) Industrial robot control and operator training using virtual reality interfaces. Comput Ind 109:114–120
21. Liu H, Wang L (2020) Remote human–robot collaboration: a CYBEr–physical system application for hazard manufacturing environment. J Manuf Syst 54:24–34
22. Iliukhin VN et al (2017) The modeling of inverse kinematics for 5 DOF manipulator. Proc Eng 176:498–505
23. Su H et al (2018) Safety-enhanced collaborative framework for tele-operated minimally invasive surgery using a 7-DoF torque-controlled robot. Int J Control Autom Syst 16:2915–2923
24. Chen X, Zhao B, Wang Y, Xu S, Gao X (2018) Control of a 7-DOF robotic arm system with an SSVEP-based BCI. Int Neural J Syst 28(08):1850018. https://doi.org/10.1142/S0129065718500181
25. Köker R, Çakar T (2016) A neuro-genetic-simulated annealing approach to the inverse kinematics solution of robots: a simulation based study. Eng Comput 32:553–565
26. Patil A, Kulkarni M, Aswale A (2017) Analysis of the inverse kinematics for 5 DOF robot arm using DH parameters. In: IEEE international conference on real-time computing and robotics (RCAR), pp 688–693
27. Singh S, Singla E (2016) Realization of task-based designs involving DH parameters: a modular approach. Intel Serv Robot 9:289–296
28. Dereli S, Köker R (2018) IW-PSO approach to the inverse kinematics problem solution of a 7-DOF serial robot manipulator. Sigma J Eng Nat Sci 36:77–85
29. Edla DR, Kongara MC, Cheruku R (2019) A PSO based routing with novel fitness function for improving lifetime of WSNs. Wireless Pers Commun 104:73–89
30. Malhotra R, Khanna M (2019) Dynamic selection of fitness function for software change prediction using particle swarm optimization. Inf Softw Technol 112:51–67
31. de Souza EF, Goues CL, Camilo-Junior CG (2018) A novel fitness function for automated program repair based on source code checkpoints. Genet Evolution Comput Confer 2018:1443–1450
32. Greche L, et al (2017) Comparison between Euclidean and Manhattan distance measure for facial expressions classification. In: International conference on wireless technologies, embedded and intelligent systems (WITS), pp 1–4
33. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
34. Ozsoydan FB (2019) Effects of dominant wolves in grey wolf optimization algorithm. Appl Soft Comput 83:105658. https://doi.org/10.1016/j.asoc.2019.105658
35. Al-Tashi Q, Kadir SJA, Rais HM, Mirjalili S, Alhussian H (2019) Binary optimization using hybrid grey wolf optimization for feature selection. IEEE Access 7:39496–39508. https://doi.org/10.1109/ACCESS.2019.2906757
36. Ibrahim RA, Elaziz MA, Lu S (2018) Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization. Expert Syst Appl 108:1–27
37. Niu P, Niu S, Chang L (2019) The defect of the Grey Wolf optimization algorithm and its verification method. Knowl-Based Syst 171:37–43
38. Faris H et al (2018) Grey wolf optimizer: a review of recent variants and applications. Neural Comput Appl 30:413–435
39. Gu Q, Li X, Jiang S (2019) Hybrid genetic grey wolf algorithm for large-scale global optimization. Complexity
40. Sharma P et al (2019) Diagnosis of Parkinson's disease using modified grey wolf optimization. Cogn Syst Res 54:100–115
41. Pradhan M, Roy PK, Pal T (2018) Oppositional based grey wolf optimization algorithm for economic dispatch problem of power system. Ain Shams Eng J 9:2015–2025
42. Natesan G, Chokkalingam A (2019) Task scheduling in heterogeneous cloud environment using mean grey wolf optimization algorithm. ICT Express 5:110–114
43. Khandelwal A et al (2018) Modified grey wolf optimization algorithm for transmission network expansion planning problem. Arab J Sci Eng 43:2899–2908
44. Kalemci EN et al (2020) Design of reinforced concrete cantilever retaining wall using Grey wolf optimization algorithm. Structures 23:245–253
45. Rahmani M, Komijani H, Rahman MH (2020) New sliding mode control of 2-DOF robot manipulator based on extended grey wolf optimizer. Int J Control Autom Syst 2020:1–9
46. Rao AM, Ramji K, Kumar TN (2018) Intelligent navigation of mobile robot using grey wolf colony optimization. Mater Today: Proc 5:19116–19125
47. Zhou Z et al (2018) Color difference classification based on optimization support vector machine of improved grey wolf algorithm. Optik 170:17–29
48. Long W et al (2017) A modified augmented Lagrangian with improved grey wolf optimization to constrained optimization problems. Neural Comput Appl 28:421–438
49. Nadimi-Shahraki MH, Taghian S, Mirjalili S (2021) An improved grey wolf optimizer for solving engineering problems. Expert Systems with Applications 166:113917. https://doi.org/10.1016/j.eswa.2020.113917

50. Gao ZM, Zhao J (2019) An improved grey wolf optimization algorithm with variable weights. Comput Intell Neurosci 2019:2981282. https://doi.org/10.1155/2019/2981282
51. Heidari AA, Pahlavani P (2017) An efficient modified grey wolf optimizer with Lévy flight for optimization tasks. Appl Soft Comput 60:115–134
52. Sun J, Feng B, Xu W (2004) Particle swarm optimization with particles having quantum behavior. In: Proceedings of the congress on evolutionary computation
53. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 62:51–67
54. Karaboga D, Basturk B (2008) On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput 5:687–697