

IK-FA, a New Heuristic Inverse Kinematics Solver Using Firefly Algorithm

Nizar Rokbani, Alicia Casals and Adel M. Alimi

Abstract In this paper, a heuristic method based on Firefly Algorithm is proposed for inverse kinematics problems in articulated robotics. The proposal is called, IK-FA. Solving inverse kinematics, IK, consists in finding a set of joint-positions allowing a specific point of the system to achieve a target position. In IK-FA, the Fireflies positions are assumed to be a possible solution for joints elementary motions. For a robotic system with a known forward kinematic model, IK-Fireflies, is used to generate iteratively a set of joint motions, then the forward kinematic model of the system is used to compute the relative Cartesian positions of a specific end-segment, and to compare it to the needed target position. This is a heuristic approach for solving inverse kinematics without computing the inverse model. IK-FA tends to minimize the distance to a target position, the fitness function could be established as the distance between the obtained forward positions and the desired one, it is subject to minimization. In this paper IK-FA is tested over a 3 links articulated planar system, the evaluation is based on statistical analysis of the convergence and the solution quality for 100 tests. The impact of key FA parameters is also investigated with a focus on the impact of the number of fireflies, the impact of the maximum iteration number and also the impact of (α , β , γ , δ) parameters. For a given set of valuable parameters, the heuristic converges to a static fitness value within a fix maximum number of iterations. IK-FA has a fair convergence time, for the tested configuration, the average was about 2.3394×10^{-3} seconds with a position error fitness around

N. Rokbani (✉)

High Institute of Applied Sciences and Technology, University of Sousse,
Cité Taffala (Ibn Khaldoun), Sousse 4003, Tunisia
e-mail: nizar.rokbani@ieee.org

N. Rokbani · A.M. Alimi

REGIM-Lab.: REsearch Groups in Intelligent Machines, University of Sfax, ENIS,
BP 1173, Sfax 3038, Tunisia
e-mail: adel.alimi@ieee.org

A. Casals

Institute for Bioengineering of Catalonia and Universitat Politècnica de Catalunya.
BarcelonaTech., Barcelona, Spain
e-mail: alicia.casals@upc.edu

© Springer International Publishing Switzerland 2015

A.T. Azar and S. Vaidyanathan (eds.), *Computational Intelligence Applications in Modeling and Control*, Studies in Computational Intelligence 575,
DOI 10.1007/978-3-319-11017-2_15

3.116×10^{-8} for 100 tests. The algorithm showed also evidence of robustness over the target position, since for all conducted tests with a random target position IK-FA achieved a solution with a position error lower or equal to 5.4722×10^{-9} .

Keywords Robotics · Inverse kinematics · Heuristics · Computational kinematics · Swarm intelligence

1 Introduction

Inverse Kinematics, IK, is still a challenging problem in robotic systems, essentially needed for path planning, motion generation and control. In robotics, trajectories generation is prior to control in a large panel of applications including wheeled, legged or articulated systems. Path and motion planning are generally expressed in the Cartesian frame, the control is performed relative to the joint-frames. IK produces the motions of the joints in accordance with the needed Cartesian path or trajectory [33, 46]. For this class of application several techniques for solving IK was investigated, due to the complexity of the analytical solutions essentially when the system has a high number of Degrees of Freedom, DOF.

In computer games applications, IK has to tackle a couple of constraints, being real time and producing precise solutions so that the animations can be perceived as natural as possible. The cyclic coordinate descent technique, CCD, was first proposed for that purpose before being applied to robotics, it is one of the earliest heuristic inverse kinematics solvers, providing a good balance between computational cost and stability [17], CCD was not built to find optimal solutions but to find a fast feasible solution in a limited computing time. Since that several heuristics were investigated with more or less success [20, 23, 29]. In heuristic based IK solver we can identify two main classes: Computational based techniques and learning based techniques. The first class is based on nature inspired heuristics such as Particle Swarm Optimization, PSO [25, 26, 40], Ant Colony Optimization, ACO [20], Genetic Algorithms, GA [12, 46]; Buckley et al. [4] or Ant Bee Colony, ABC [6]. For these methods, the IK problem is addressed as an optimization and the heuristics is used as computing alternatives. The second class includes nature inspired techniques and hybrid techniques with learning capacities such as neural networks and neuro-fuzzy systems [13, 22, 32, 45] where the IK solver is observed as system with a set of inputs and output(s), the system needs first to be trained using a set of targets and solutions generally obtained using the forward kinematic model, it is then validated with a separate validation set and finally used. When the target point is far from the training set this systems generate limited precision solutions compared to Jacobian based conventional methods [5, 7]; they are also time consuming and can not satisfy real time constraints [22, 44].

In humanoid robotics, legs, arms and fingers are typical articulated systems, with a similar kinematic chain while dynamics are different. Inverse kinematics is needed at all levels: for leg motion planning in walking, for arms and fingers motions in handling and grasping [2, 36]. Due to these reasons, it is easier to plan motion in the Cartesian frame, since it is the physical frame of the environment. Then an inverse transformation is needed to compute the needed displacements in the joints frames prior to control [24, 27], such a technique is also used in 3D character animation or gaming applications. In humanoid robotics and 3D humanoids simulations, the human motion analysis was a key source of inspirations for walking gaits, arms motions, grasping [37, 39], body postures and biped walking [1, 3]. At this level also inverse kinematics are needed to generate skeleton's joint motions that could fit a robotic design while satisfying a human like motion in the Cartesian frame [16] form a set of marked human motion primitives. Such inverse solvers should satisfy real-time constraints and should not suffer from any singularity which is not the case of classical IK solvers. Classical techniques consists in finding an approximation of the inverse kinematics of a system when the analytical expression of the inverse is complex and difficult to compute; this class of methods include the pseudo-inverse methods, the Jacobian transpose, the quasi Newton and the damped least square methods; classical inverse methods are time consuming essentially in systems with a high DOF [5, 7].

In inverse kinematics based on PSO [25]; or on GA such as [4, 12, 46], a stochastic search is performed, using a population in GA or a set of individuals in PSO, ABC and ACO; each population or individual is a possible solution, here a set of joints positions of the IK problem. Any possible solution is ranked, using a fitness function and the best is returned as the solution of the problem, for a given input, here a target position. The common aspects of these methods is that they try to solve inverse kinematics by evolving iteratively a set of solutions using a limited set of operations that mimic a natural process, swarm behavior and its social organization in the case of PSO, ABC or ACO [9, 10, 14]; GA tends to solve a problem using the natural evolution mechanisms [21]. The design of the optimality criteria is what makes a heuristic render an acceptable result. In IK, a trivial objective could be used, it consists in minimizing the distance to the target position. For applications such as humanoid gait generation, it is possible to add some constraints to obtain a solution which fits better this specific class of robotic systems. In real world applications, the solutions of an articulated arm or an artificial leg should respect the mechanical design of the system. The adaptation of the heuristic IK methods is possible for any kind of robotic system and do not need to be trained.

On the other hand neural network [1] and neuro-fuzzy techniques tend to do the same but after a training process, where a neural or a neuro-fuzzy network is trained by a set of joint motions and their correspondent Cartesian solutions [13, 32, 44, 45]. The training process has a direct impact on the quality of the obtained IK solver, for these intelligent IK solvers designing a good training set is essential. Neuro-fuzzy has the advantage to be interpretable when compared to neural-network IK solutions;

they are accurate but suffer from computing time, and could not be used in real time applications [22, 37].

The remaining of this paper is organized as detailed below. Section 2 reviews the key issues of kinematics modeling with a focus on inverse kinematics challenges; this section reviews the concept of heuristic solvers with a focus on CCD, which is a reference heuristic for IK. Section 3 starts with a review on the FA heuristic, then a new heuristic approach for inverse kinematics based on the Firefly Algorithm, FA, is proposed, the proposal is called IK-FA. IK-FA proposal is detailed for unconstrained and constrained inverse kinematics problems. In Sect. 4 a set of simulation based experiments are detailed, the key aspects of IK-FA were subject to investigation over a classical 3 links articulated system, investigations concerned the impact of IK-FA parameters on convergence and performances. Finally the paper ends with discussions, conclusions and perspectives.

2 Inverse Kinematics

In Robotics two aspects are important, kinematics and dynamics [36, 37]; Kinematics deals with how the motions of a mechanism are related to the relative positions of the end effectors of the system in accordance with a reference frame [5, 37]. The motion is studied regardless to what produced it. In robotics kinematics analysis are needed to plan a robot motion with respect to the work space geometric configuration and satisfying angular and geometric constraints that the system could be subject to [24]. Kinematics is forward or inverse: In forward kinematics the mechanism motions are known while the end effectors positions need to be computed. In inverse kinematics the end effectors positions are known and the joint motions involved to achieve them need to be computed [5, 35].

2.1 Forward Kinematics

Assume that $X = (X_1, X_2, \dots, X_l)$ is the position of an articulated body of (n) elements subject to a set of elementary rotations and/or translations, $q = (\theta_1, \theta_2, \dots, \theta_n)$ so the forward kinematics is expressed by Eq. (1).

$$X = f(q) \tag{1}$$

The forward kinematics model can be obtained systematically whatever is the complexity of the mechanism. It is decomposed into a set of primitive Transformations according to a coordinate frame and the forward kinematics function is obtained systematically by composing the elementary transformations T_{i-1}^i ,

$$f(q) = \prod_{i=1}^n T_{i-1}^i. \tag{2}$$

An elementary transformation refers to a rotation towards an axe of the coordinate's frames or a translation on a given direction.

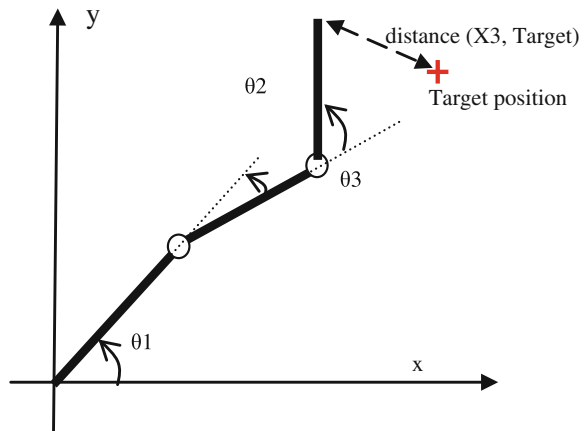
2.2 Inverse Kinematics

Inverse kinematics consists in finding a possible and feasible joints motions solution allowing a robotic system, typically articulated, achieving a pre-defined position, called target position. For an articulated system such in Fig. 1, let's assume the joint rotations needed to produce the motion to $q = (\theta_1, \dots, \theta_n)$; the robot position in a Cartesian reference frame $X = (x_1, \dots, x_i)$, is obtained as the output of the forward kinematics function, $f(q)$ of the system with Q_i as input as in (1). If we assume that the forward kinematics are expressed using a mathematical function $f()$, inverse kinematics, IK, could simply be the inverse of that function meanwhile and considering the nature of $f()$ which is a matrix, its inverse is not for sure defined, retrieving the inverse kinematics function depends on the invertibility of the forward form, the generic formulation of Eq. (3) is in most cases difficult to compute and some approximations are needed to retrieve the IK models [5, 34, 35].

$$q = f^{-1}(X) \tag{3}$$

The main problem in IK is the existence $f^{-1}()$; in the most cases an analytic expression of the IK function is difficult to obtain. Several computational methods are proposed to tackle this problem. The most Known approach is based on the

Fig. 1 Simplified representation of an articulated system composed by 3 links and 3 revolute joints



Jacobian of forward kinematics function $f()$. The Jacobian is the multidimensional expression of the classical differential operator as in Eq. (4), it is also possible to compute $J(q)$ iteratively [2, 5, 36].

$$J(q) = \left(\frac{df}{dq} \right) \quad (4)$$

Using (1) and (4) the linear velocity could be expressed according to the angular velocities of the joints angular positions by (5).

$$\frac{dX}{dt} = J(q) \frac{dq}{dt} \quad (5)$$

For very limited motions, the previous equation could be expressed using differential form instead of the derivatives; It leads to an expression of the elementary position displacement for a given elementary joint displacements as in (6)

$$\delta X = J(q) \delta q \quad (6)$$

The inverse form of this equation is expressed in (7) allowing to compute the small amounts of joints positions changes for a given small relative variation of the end effectors position. Here the problem of IK is simply transformed in computing or finding the inverse of the Jacobian of the forward kinematics function.

$$\delta q = J(q)^{-1} * \delta X \quad (7)$$

Around this concept, several methods are proposed all of them belong to the same class of IK solutions, the Jacobian based IK. Note that if the dimension of the Cartesian position vector is different from the dimension of the joints angular rotations vector $q()$, the $J(q)$ matrix is rectangular and is simply not invertible, it could also suffer from singularities. For systems with high DOF, the analytical solution of inverse kinematics is difficult to express [5]. The Jacobian transpose method used the transpose of $J(q)$ instead of its inverse. The pseudo inverse method replaces the Jacobian by its pseudo inverse, while it is not the exact inverse of J is still a good approximation. The main challenge with Jacobian inverse methods is how to compute the Jacobian directly or iteratively [5, 7].

2.3 Heuristic Inverse Kinematics

For a given target position and knowing the current position, the classical method to solve inverse kinematics consists in retrieving a $q()$ using the inverse kinematics function as in Eq. (3). This solution exists only if the inverse kinematics function is

defined, while in the most of the cases this function is hard to obtain, and suffer from singularities [5].

An Heuristic solution to this problem consists in using an heuristic search method to find iteratively a set of $q()$, by reducing the position error, as in Eq. (8), which is the distance of the end-segment to the needed target position, as shown in Fig. 1. In the case of PSO, particle swarm optimization, the heuristic will guide the search using a set of particles; each one is a potential solution of the problem [10, 11]. The quality of a solution is evaluated using a fitness function, which is naturally quantifying the error of the obtained target versus the needed one.

$$e = \|x_t - f(q_i)\| \quad (8)$$

In inverse kinematics solver using PSO, IK-PSO, the fitness function is the distance of the end-effectors, or specific point of the system to the target [25, 26, 28], its general expression of a dimension (d) is given by (9).

$$f_{\text{fitness}(i)} = \sum_{i=1}^d (x_i - x_t)^2 = \sum_{i=1}^d (f(q)_i - x_t)^2 \quad (9)$$

where (d) denotes the dimension, in the case of Fig. 1, $d = 2$. and the fitness function is expressed by (10).

$$f_{\text{fitness}(i)} = (f(q)_{x_i} - x_t)^2 + (f(q)_{y_i} - y_t)^2 \quad (10)$$

The fitness function is homogenous with the square of a Euclidian distance. In IK-PSO, the inverse kinematics is solved using the forward kinematics function and a heuristic search strategy. By over coming the computing of the inverse forms IK-PSO has no problem of definition, singularities and does no need any matrix inverse computing. The effectiveness of PSO allowed coming to a solution within reasonable time and position error [25].

In the case of the cyclic coordinate descent (CCD), the computational efforts are reduced by varying one joint variable at a time. The goal is to reduce position and orientation errors. At each iteration, a single traversal of the manipulator is performed from the target position, where the final link is, towards the manipulator base; this allowed to CCD to produce an iterative IK solution [2, 4, 11]. For an articulated system composed of (n) links CCD solves a single joint position $q(i)$ using a minimization of the end-segment to the target point.

The link direction is estimated using classical trigonometric approximation of a virtual link joining the target position to the link base, this mean that the joints rotations are produced iteratively and this could lead to non feasible solutions for a robotic system. On the other hand and from a computational point of view, CCD has the advantage to render a solution in a limited time since it computes (n) elementary motions instead of the solution of an (n) joints system. The method handle a single DOF problem relative to a joint $q(i)$, such a problem is simple to solve using classical trigonometry.

3 Inverse Kinematics Using Firefly Algorithm, IK-FA

In this paragraph we first give a brief on the essentials of Firefly Algorithm, FA, metaheuristic, and then a detailed description on how FA was used to solve inverse kinematics is given.

3.1 The Firefly Algorithm (FA)

The Firefly Algorithm (FA) was proposed by Xin-She Yang, as a bio-inspired heuristic from the flashing behaviour of the fireflies. Fireflies use flash light to attract others, the more a light is visible, the more its attraction capacity rises, only this aspect was considered by Xin-She Yang in his algorithm regardless to what is intended behind [43], it was then tested over the classical test functions [18, 41, 42]. FA, assumed all individual to be unisexual, an individual is attracted by a light with a flashing capacity which is higher to its own. The light intensity naturally decreases when the distance to light source increases. The relative displacement between fireflies is given by a position update equation as in (11).

$$x_i = x_i + \beta(x_j - x_i) + \alpha\varepsilon \quad (11)$$

where, x_i, x_j : are respectively the current position of the fireflies (i) and (j). It is important to note here that the position update of any firefly is adjusted with regard to its own current position and also the current positions of all swarm individuals. This is the main difference between FA algorithm and PSO where an individual position depends on a couple of specific particles, the local best and the global best.

An individual of a FA swarm is moved towards any other with higher brightness, this displacement is moderated by the attractiveness coefficient β and a random displacement ($\alpha\varepsilon$). The neighborhood is composed of fireflies within the perception field of the individual. The firefly with a lower brightness is moved towards the one with a higher brightness [43]; a simplified pseudo-code of FA algorithm is given in Fig. 2.

In Eq. (11) the term β represents the attractiveness coefficient which depends also on the distance separating the firefly (i) to individual (i), it is expressed as in Eq. (12).

$$\beta = \beta_0 e^{(\gamma r_{ij}^2)} \quad (12)$$

The final term of Eq. (11) could be observed as a step size with a moderation parameter α and were ε could be derived randomly from a Gaussian distribution. In FA, the brightness of a firefly $I(x)$ could be used as the fitness function of the problem to optimize as expressed in (13).

```

Begin
1) Initialize FA parameters :  $\gamma, \alpha, \beta,$ 
2) Generate an initial population of fireflies  $Q=(0, \dots, x_n)$ 
3)   For  $i\_t=1$  : maximum_iteration
4)     for  $i = 1 : n$  (all n fireflies)
5)       for  $j = 1 : n$  (n fireflies)
6)         Compute  $I_i, I_j$ 
7)         if  $(I_i < I_j)$ 
8)           move firefly i towards j
9)         end if
10)      end for j
11)    end for i
12)    Solution = best current fireflies;
13)    Reduce  $\alpha$ 
14)  End for  $i\_t$ 
15)  Return best firefly (Solution);
End

```

Fig. 2 FA Algorithm pseudo code

$$I_o^i = f_{imess}(x_i) \tag{13}$$

The brightness of an individual (i) is also subject to a natural lost when observed from the position of an individual (j), this lost is expressed as in Eq. (14), where (r) is the distance of firefly (j) to firefly (i) and γ the absorption coefficient.

$$I_i = I_o^i \cdot e^{(-\gamma r)}. \tag{14}$$

3.2 The IK-FA for Inverse Kinematics

This paragraph details how FA is adapted to solve IK for unconstrained inverse kinematics prior to detail how constraints can be handled. The inverse kinematics solver using PSO is called IK-FA, Inverse Kinematics solver using Firefly Algorithm.

3.2.1 Unconstrained IK-FA

To use FA in solving inverse kinematics, the firefly here is assumed to be set motions primitives limited to rotations and translations. The Fireflies positions correspond to robotic motion expressed in the joint frames; then the equivalent position of the end-segment is retrieved in the Cartesian Frame using the Forward kinematics of the system. Here, we have no more to care about singularities or to inverse the forward kinematics function. Only the Forward kinematics, a target

position and a satisfaction condition is needed, a trivial satisfaction condition only could simply be a fixed error position.

Given a system composed of (n) joints, a firefly position at iteration (i) can be expressed as in (15), fireflies evolves in swarms of (n) individuals. For an articulated system such in Fig. 1, where the joints are only subject to rotations, the FA position is expressed by Eq. (15).

$$(q_j)_i = (\theta_1^j, \dots, \theta_n^j)_i \quad (15)$$

The Firefly Algorithm is an iterative computing heuristic and Eq. (15) refers to the position of the firefly (j) at iteration (i). The Cartesian position of end-segment, obtained with the joints solution of firefly (j) at iteration (i) is expressed as in (16).

$$X_{ji} = f(\theta_1^j, \dots, \theta_n^j)_i \quad (16)$$

where f() stands for the forward kinematics function of the system. A trivial fitness function for the system could be expressed by the square of Euclidian distance of the target point to the end-segment position of the system, see Eq. (17).

$$f_{imess}(q_j)_i = (x_t - x_{ji})^2 \quad (17)$$

The brightness of a firefly is maximized as it comes closer to the target position, while instead of using the exponential function, its first order differential equation is used and the brightness of a firefly is expressed by (18).

$$I_j^i = \frac{1}{1 + \gamma(x_t - x_{ji})^2} \quad (18)$$

where (j) is the firefly identifier and (i) the iteration counter, the brightness is designed so that it comes to a maximum as the target point is achieved, $I = 1$. Note that the brightness is related to the distance of a firefly to target position while the firefly himself is an angular position. As is PSO, a stop is observed when the maximum number of iteration is achieved or when the fitness function is satisfied.

3.2.2 Constrained IK-FA

To solve inverse kinematics of systems subject to constraints, such as mechanical arms or bio-inspired systems with biomechanical constraints like artificial limbs, the IK-FA should respect Cartesian constraints and joints constraints. In Constrained IK-FA generated solutions are subject to a couple of tests, the first one consists in checking if joint constraints are respected. The second check is about the Cartesian constraints.

Cartesian constraints verification consists in verifying if a set of end-effectors are within a predefined work space, meanwhile the firefly, by mean of solution, is simply ignored.

Joints constraints are needed to limit the motion into a feasible space. The joints constraints could be imposed by the mechanical design, in this case any motion with violation to those constraints should be simply avoided. Joints constraints could also be useful to produce gaits that are close a predefined biomechanical one: for example knee and ankles rotation limits in the case of human walking or shoulder and elbow limits in human arm. A typical illustration of joints and Cartesian constraints appears in Fig. 3.

Two alternatives are possible to handle those constraints:

- Handle as a specific reward within the fitness function.
- Handle as separate condition with a control mechanism in FA.

It is easier to use the second scenario, since only a set of tests are needed to be added to the original IK-FA in order to make it able to handle the constraints. To use the first scenario the brightness of a solution which do not respects the constraints could be simple waved, decreased, so that no other individual of the swarm is attracted to.

Constraints could be added to the joints search space and also to the solutions search space they could be expressed as in Eq. (19) where (J1) is the expression to limit the angular displacement of a specific joint into a specific interval while (J2) is the complementary constraint.

$$J1: q_i(j) = \theta_j \in [\theta_{jmin}, \theta_{jmax}] \tag{19}$$

$$J2: q_i(j) = \theta_j \notin [\theta_{jmin}, \theta_{jmax}] \tag{20}$$

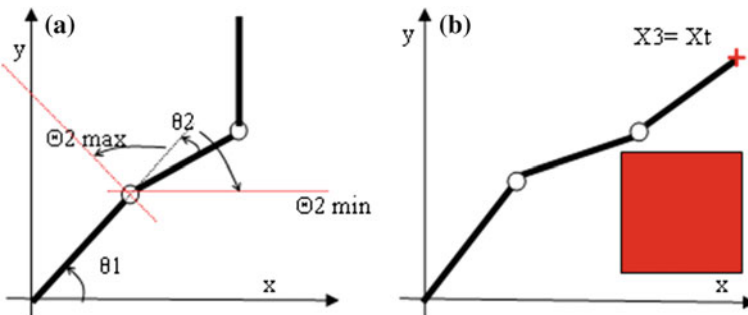


Fig. 3 Typical constraints illustration, a illustration of a joint constraint, b illustration of Cartesian space constraint

Constraints could also be expressed in the Cartesian space, so that a specific (X_i) position is within a specific convex hull. In this case the constraint is similar to J3, given by Eq. (21).

$$J3: X_i(j) = f(q_i) \in C(X) \quad (21)$$

$C(x)$ is the convex hull of the solutions space in the case of Eq. (21). It is the convex hull excluded from the solution space in the case of Eq. (22).

$$J4: X_i(j) = f(q_i) \notin C(X) \quad (22)$$

Figure 3a gives an illustration of a typical joint constraints, and an example of Cartesian space constraint. This could be expressed by (23).

$$J_4: X_i() = f(q_i) \notin [X_{\min}, X_{\max}] \quad (23)$$

In all cases, the faulty firefly is replaced by a random one with respect to needed constraints. The pseudo code of the Constrained IK-FA is presented in Fig. 4.

For the pseudo-code of Fig. 4, the fitness function is subject to minimization, as defined in (18) the brightness is maximal as the fitness approaches zero, this mean that the best firefly, will be the brightest one and will be the one with fitness as close to zero as possible. Note that it is possible to code IK-FA with a minimization

```

Begin
1) Initialize IK-FA parameters
2) Generate an initial population of fireflies Q =(0, ... . xn)
3) Xt = Input (target position)
4) for i_t=1 to max_iteration number
5)     for i = 1 : n (all n fireflies)
6)         for j = 1 : n (n fireflies)
7)             if (Ii < Ij )
8)                 move firefly i towards j
9)             end if
10)        end for j
11)    end for i
12) Solution = best current fireflies;
13) For k= 1 to max_joint number
14) If J1 = True
15)     Then Q ← random ([qmin, qmax])
16) If J4 = True [Xmin, Xmax]
17)     Then Q ← random ([qmin, qmax])
18) end for k
19) if (error < fixed) break
20) end for i_t
21)     Return best firefly
End

```

Fig. 4 IK-FA with Cartesian and joint constraints

formulation directly by imposing to the brightness expression to be equal to the fitness as in (24).

$$I_j^i = fitness(q_j)_i \quad (24)$$

where I_j^i the brightness of firefly (j) by iteration (i); in this case the FA procedure should be slightly adjusted so that the firefly with the higher brightness is moved toward the one with a lower brightness, since lower brightness indicates a better solution. This modification should be made on line (8) of the pseudo-code of Fig. 4.

4 Experimental Results

The experimental simulation process consisted in implementing the IK-FA solver for a 3 links planar articulated system, in a humanoid, such a system could serve as a simplified kinematic model of an arm, a leg or a finger, depending on the segments sizes and parameters. The simulations were dedicated to the performance evaluation of IK-FA, it covers the following aspects:

- The convergence capacities.
- The Impact of FA parameters on the convergence and on the quality of results.
- The impact the swarm size on the quality of the solutions and on the processing time.
- The robustness of the algorithm over the target position.

Experimental results are based on simulations using matlab software, version 7.6 (R2008a), the software runs on a personal computer with 4 Go of DRAM and T4200 processor cadenced at 2 GHz. All results are presented for 100 tests, and an estimation of the density of probability of the fitness function which is the square of the position error between the end-effector position and the target position.

4.1 The Experimental Protocol

The first test bench is a generic articulated system composed by 3 links and a 3 revolute joints, similar to what appears in Fig. 1, it represents a 3 DOF articulated system that could be used for a leg of for an arm planar model. In the case of a leg the links (11), (12) and (13) represents respectively the thigh and the tibia and the foot. To apply the IK-FA, we first write the forward kinematics of that system, as in is the system of equations given by (25).

$$\begin{cases} x_1 = l_1 * \cos(\theta_1) \\ y_1 = l_1 * \sin(\theta_1) \\ x_2 = l_1 * \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \\ y_2 = l_1 * \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \\ x_3 = l_1 * \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \\ y_3 = l_1 * \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \end{cases} \quad (25)$$

For IK-FA, The inverse kinematics problem, relative to a given target position $X_t = (x_t, y_t)$ for the terminal end-segment position, could be written as follows:

$$\text{Find: } Q = [\theta_1, \theta_2, \theta_3] \text{ Satisfying } f_{p3}(q) = X_t \quad (26)$$

This formulation corresponds to a non constrained IK problem, where $f_{p3}()$ is the forward kinematics function of system limited to the end-segment of link (3). The fitness function used for both heuristics is given by (27):

$$f_{imess}(q_j) = \|f_{p3}(q_j) - X_t\|^2 \quad (27)$$

All tests were performed with target position (0.700, -0.500). The impact of parameters is estimated based on the mean results obtained over 100 tests for each variant. A simulation of the 3 links articulated system is also produced for the best solution, such in Fig. 5a.

The couple of parameters that are used to evaluate the performances are the fitness function and the computing time which is related to the iteration number needed to converge. The fitness function used here is the square of the distance error, this consideration allowed, for some tests, to fix the position error by controlling its square instead of computing the square rough.

All tests results were visualized using a Cartesian frame such in Fig. 5a, which shows the best solution found by the end of the processing. We also systematically plot the evolution of the fitness function in order to see if a convergence behavior is observed and to evaluate the precision of the obtained solutions. A typical plot of the fitness function for a solution appears in Fig. 5b, where the fitness is plotted iteratively.

For general conclusions, the mean and the standard deviation for a set of 100 tests are used to evaluate the impact of IK-FA parameters on the results. The mean is the average of the fitness function computed for a given configuration test using the distributions fitting tool of Matlab. This tool allowed also plotting an approximation of the density of probability using a normal distribution.

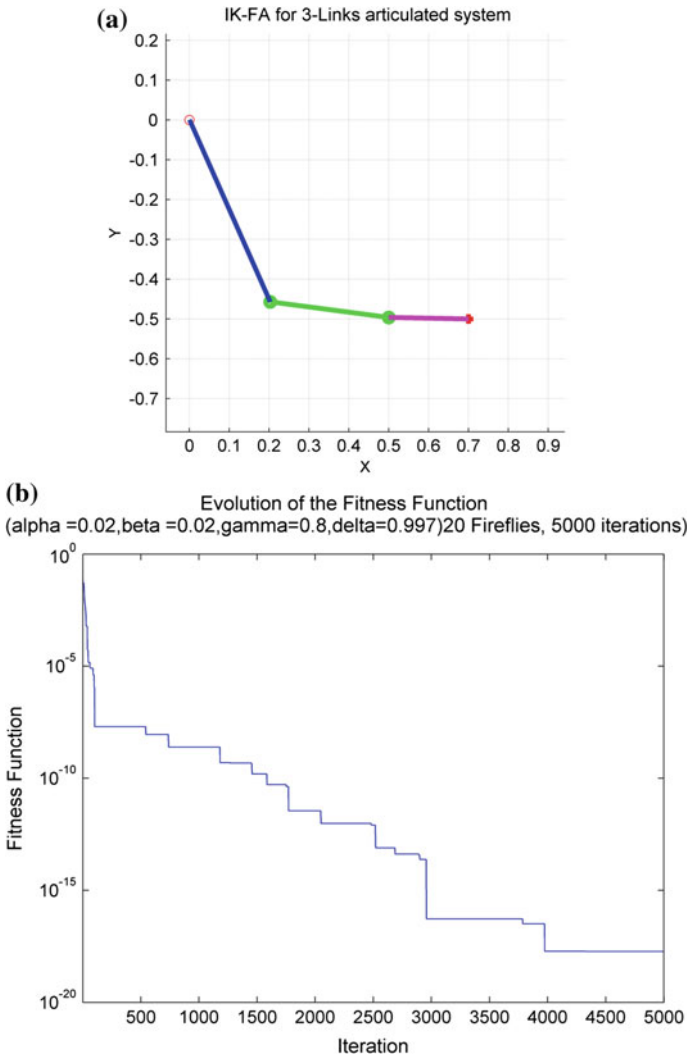


Fig. 5 IK-FA Typical solution. **a** 3 links arm with a target position (0.700, -0.500), **b** fitness function evolution for ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$) and 20 fireflies.

4.2 Performances Analysis of IK-FA

Performances analyses are based on the evaluation of the fitness function of the obtained solutions as well as the convergence time. The test protocol consist in a statistical results over 100 tests. All tests were performed with the same target position (0.700, -0.500), a section is dedicated to the effect of the target position on a specific set of good parameters of IK-FA. Discussions are conducted on the effect

of the key FA algorithm parameters on convergence and performance when used in IK-FA. Statics and comparisons are made using the statistical Matlab Toolbox [19].

4.2.1 Investigation on the Possible Convergence of IK-FA

A typical set of solutions of the 3 Links system appears in Fig. 5a for ten tests. The target position is tagged with a red cross, the relative link sizes are respectively ($l_1 = 0.5$, $l_2 = 0.3$, $l_3 = 0.2$); they are obtained by dividing the real length of the link by the length of the articulated system when all links are aligned. The target position is fixed to (0.700, -0.500). Note that IK-FA returns the best solution found by the end of its processing, the simulation results of Fig. 6, corresponds to 10 results obtained from 10 different executions of the solver with the same set of parameters. Result evaluations are based on the fitness function see Fig. 6b.

IK-FA was tested first using a set of FA parameters: ($\alpha = 0.2$, $\beta = 0.2$, $\gamma = 0.8$, $\delta = 0.9$), with this set parameters, a convergence attitude is observed with a fitness mean of about 1.14735×10^{-3} , this average was observed over a statistical test of 100 runs of IK-FA. This number of tests is necessary to measure the quality of the provided solutions. The analysis of the evolution of the fitness function for several runs show that in all cases a solution is provided within 100 iterations for this set of parameters see Fig. 6b. Some results have a high quality with fitness about 1×10^{-18} , meaning that the distance error is about 1×10^{-9} ; meanwhile this kind of solutions are far from the mean performances since the worst result, obtained for this test, is a distance error of about 10^{-1} .

What could be also underlined here is the fast convergence time, since all results were reported in less then 100 iterations. Meanwhile we could not speak about a stable inverse kinematics solver due to the big range in fitness variations limits which is the square of the position error. This first test confirms the possible convergence of IK-FA to high quality solutions even if solutions with fitness under 1×10^{-16} , were only 31 over 100 tests. Solutions with fitness less than 1×10^{-6} , were 59 over 100. Weak fitness's solutions, equal or higher than 1×10^{-5} , were 41 over 100 tests. This first investigation allowed confirming that it is possible to achieve a convergence using IK-FA, meanwhile deep investigations are needed to define a good set of parameters.

4.2.2 Impact of FA Parameters on Convergence

In Firefly Algorithm $\alpha \in [0, 1]$ and $\gamma \in [0, \infty)$ in theory, but in practice, it typically varies from 0.01 to 100 [10]. In this investigation a couple of IK-parameters sets are compared respectively ($\alpha = 0.2$, $\beta = 0.2$, $\gamma = 0.8$, $\delta = 0.9$) and ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$), 10 fireflies, a maximum iteration number of (1,000), for the same target point (0.700, -0.500). This simulation showed that for the first set of parameters as discussed in the previous paragraph only 59 % of the solutions had an error lower than 1×10^{-6} and the position error ranged from 10^{-1} to 10^{-8} , see Fig. 6b.

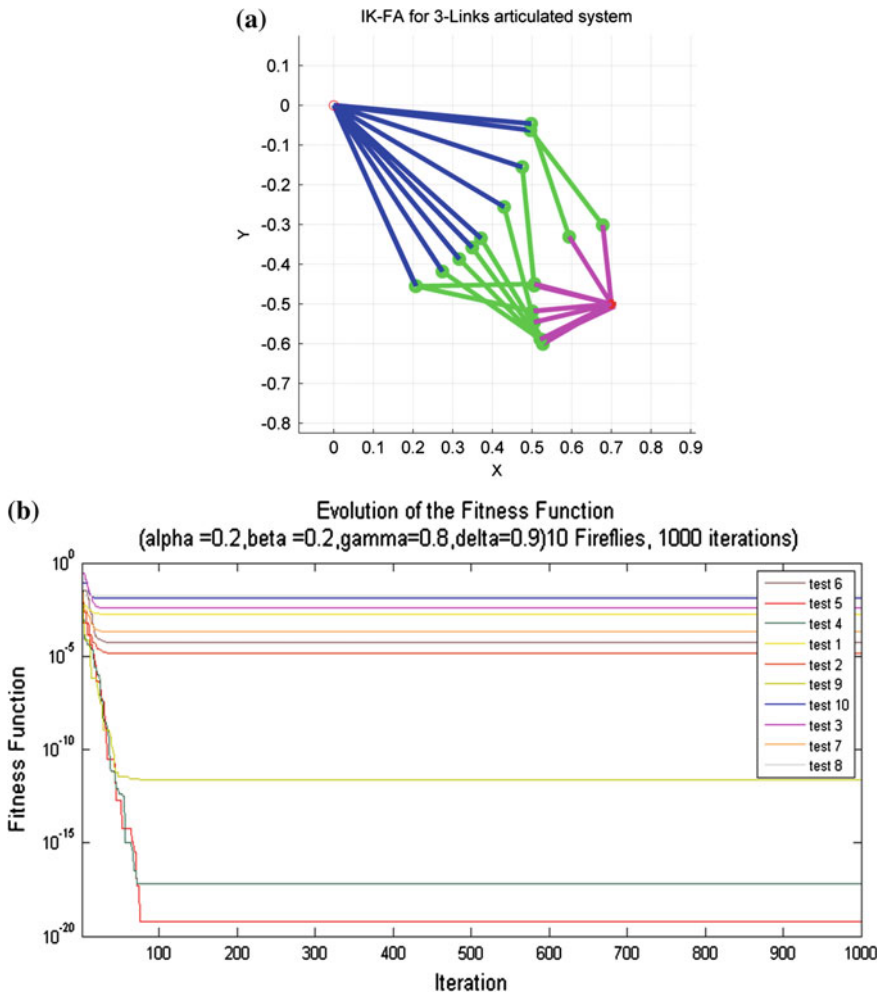


Fig. 6 IK-FA solver for a 3 links system. **a** possible solutions for a target position (0.700, -0.500), **b** evaluation of the fitness functions using ($\alpha = 0.2, \beta = 0.2, \gamma = 0.8, \delta = 0.9$) and 10 fireflies.

By using the set of parameters ($\alpha = 0.02, \beta = 0.02, \gamma = 0.8, \delta = 0.997$) we observed similar results for all the 10 tests and 100 % of the solutions where with a fitness ranging around 10^{-9} with an iteration number limited to 1,000. The evolution of the fitness function for ten tests of this configuration are displayed in Fig. 7, it showed that globally the IK-FA algorithm evolves toward decreasing its fitness function, meaning that it is evolving toward decreasing the distance to the target point. Note that this set of parameters was experimentally adjusted. For the remaining of the investigations, this set will be used.

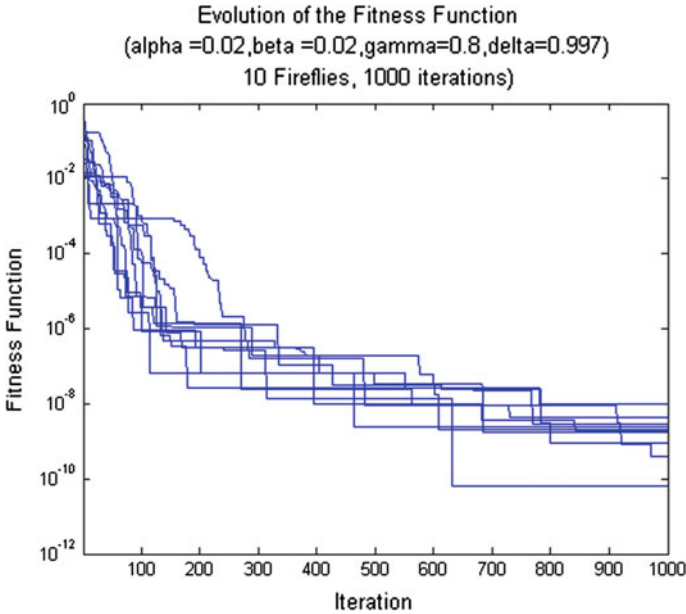


Fig. 7 Evolution of the fitness function for 10 tests, ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$) and 10 fireflies

4.2.3 Impact of the Maximum Iteration Number

A first investigation of the impact of the maximum iteration for the set of parameters, ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$), 10 fireflies, and a target position (0.7, 0.5) was done over 500, 1,000, 2,000, 3,000, 5,000 and 10,000 iterations. This experiment showed that the best fitness value decreases as the maximum iteration number increases; the best fitness for 10,000 iterations is about 1.27102×10^{-18} , and the worst result, at iteration 10,000, is 1×10^{-17} .

The fitness values observed around 500, 1,000 and 2,000 were decreasing but did not show a static fitness value, this was only achieved for 5,000 iterations, and clearly confirmed by the test of 10,000 iterations, see Fig. 8.

Using the distribution fitting tool of Matlab, the fitness is approximated with a normal distribution with an average, mean, about 1.50×10^{-17} . For 10,000 iterations a static convergence compartment is observed around 4,500, (4,489.44) iterations. These results are confirmed by 100 tests, see Fig. 9. This experiment confirm that a valuable balance consist in fixing the maximum iteration number to 5,000. The only conclusion that could be taken at this level is that for this specific set of parameters, IK-Firefly convergence is ensured with fitness around 5×10^{-17} or lower by a maximum iteration of 5,000, see Fig. 9b.

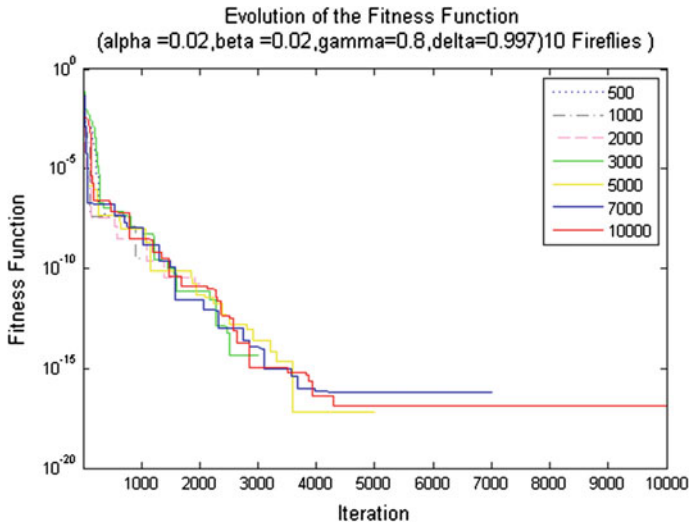


Fig. 8 Impact of the iteration number on the IK-FA convergence; Investigation for several maximum iteration numbers ranging from 500 to 10,000

4.2.4 Impact of the Swarm Size

The swarm size is the number of individuals composing the swarm. The impact of the number of fireflies is an important parameter in any swarm based heuristic; it has a direct impact on the procession time and also on the quality of the solutions. In swarm based techniques, such as PSO or GA, population sizes of 10 to 60 are commonly used [8, 38]. The investigation on the effect of the FA swarm size conducted here is specific to the IK-FA algorithm.

The number of fireflies are waved from 10 to 60 for a fixed target point (0.700, -0.500), the set of ($\alpha = 0.02, \beta = 0.02, \gamma = 0.8, \delta = 0.997$) parameters and a fixed maximum iterations number = 5,000. For any given set of firefly’s numbers the tests are repeated 100 times prior to any interpretations. The fitness functions are then subject to a statistical investigation using the distribution fitting tool of Matlab statistics Toolbox [19]. Interpretations are based on the mean and standard deviation values of the fitness’s on the tests. The fitness corresponding to a given swarm size is approximated by a normal distribution of the density of probability function, DPF, and the mean is used to compare the impact of the swarm size, see Fig. 10.

For all swarm sizes ranging from 10 to 60 the fitness mean ranges respectively from 1.27×10^{-17} to 1.79×10^{-18} , as in Table 1, allowing to conclude that as the swarm size increases the fitness decreases and the position error which is the square root of the fitness decreases, the obtained solutions are more precise.

For a swarm size of 60 individuals, the probability to obtain a result with a fitness of $1.5e-18$ is 99.8 %. For a swarm size of 10 fireflies, the mean of the normal distribution used to approximate the results is 1.27148×10^{-17} , with a

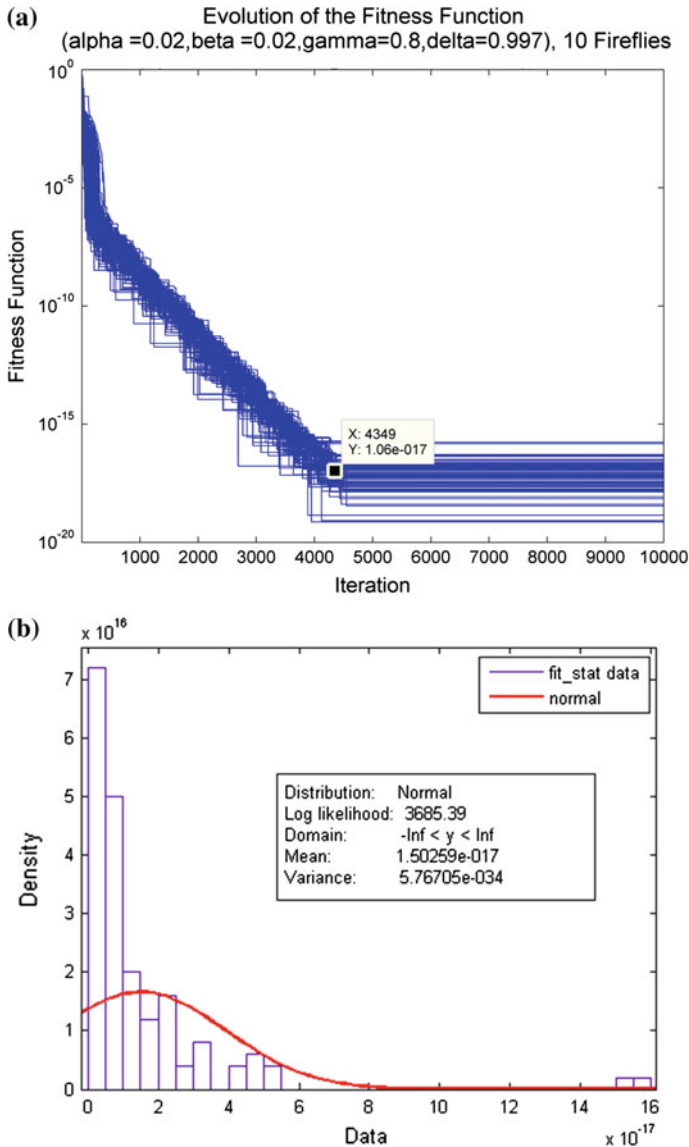


Fig. 9 Impact of the iteration number on the IK-FA convergence. **a** the evolution of the fitness function over 100 tests, **b** approximation of the fitness density of probability by a normal distribution

variance of 1.38555×10^{-34} , the probability to obtain a result with a fitness of 10^{-16} is 100 %, which could be considered as a proof of convergence of the IK-FA algorithm. Note also that results for 50 fireflies are very close to those of 60 fireflies, see Fig. 7, where the yellow distribution represents the results for 50 fireflies its

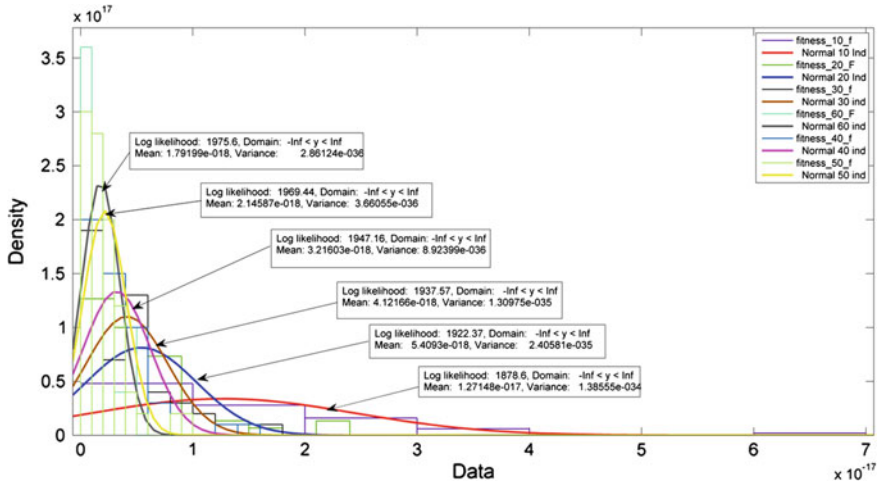


Fig. 10 Impact of the swarm size

Table 1 Computing time with maximum iteration stop condition at 5,000

Swarm size	Mean fitness function	Variance of fitness function
10	1.2714×10^{-17}	1.3855×10^{-34}
20	5.4093×10^{-18}	2.4058×10^{-35}
30	4.1216×10^{-18}	1.3097×10^{-35}
40	3.2146×10^{-18}	8.9239×10^{-36}
50	2.1458×10^{-18}	3.6605×10^{-36}
60	1.7891×10^{-18}	2.8612×10^{-36}

mean is 2.145×10^{-18} with a variance of 3.660×10^{-36} . Results for 40, 30 and 20 fireflies are also close with respective fitness means of 3.2146×10^{-18} , 4.1216×10^{-18} and 5.4093×10^{-18} . Results are resumed in Table 1.

Globally we can deduce that for a swarm size of 40–60 the fitness mean is ranged from 3.21×10^{-18} to 1.79×10^{-18} respectively with a variance ranging from 8.92×10^{-36} to 2.86×10^{-36} , for swarm’s size of 10 fireflies the fitness is 10 times lower, with a mean of 1.27×10^{-17} and a variance of 1.34×10^{-34} .

Results comparison based on normalized distributions on the fitness functions’ over 100 tests, showed that as the swarm size increased the variance of the fitness functions decreased, best results are obtained with 60 fireflies, see Fig. 8, meanwhile results with 50 and 40 fireflies are very close, see Fig. 10. Known the impact of the swarm size on the processing a good balance between fitness, swarm size and processing time is the next investigation issue.

Table 2 Computing time with maximum iteration stop condition at 5,000

Computing time (s)	Swarm size
$1.215262075163430 \times 10^{-3}$	10
$1.755538920139096 \times 10^{-3}$	20
$2.733618017054081 \times 10^{-3}$	30
$4.241389728678068 \times 10^{-3}$	40
$5.328678837890005 \times 10^{-3}$	50
$7.154939660265453 \times 10^{-3}$	60

4.2.5 IK-FA Computing Time

The next investigation concerns the impact of swarm size on the computing time; results reported on Table 2, concern the time needed for a fixed maximum iteration number of 5,000 iterations. In Table 3, the impact of the population size on computing time for a given error position, The IK-FA stop condition is modified so that it ends treatments when the position error is less or equal to 10^{-6} , meaning that the fitness function is less or equal to 10^{-12} . If the error position is not achieved the algorithm will stop at its maximum iteration fixed to 5,000 as in the previous test. The time values presented in Table 2 are average time observed over 100 tests.

Crossing the impact of the number of the population size and the computing time, it appeared that a swarm size of 20 individual is a valuable choice, since it allowed achieving a fitness function of 5.48×10^{-18} in a computing time relatively close to what we can obtain with a limited swarm size of 10 individuals. This choice is confirmed when the stop condition is modified so that the swarm stops when it achieved a desired fitness, by mean of error, details of this experimentation appears in Table 3.

4.2.6 Robustness Over the Target Point Position

In order the check the robustness of the results over the target position, 100 tests were performed with a randomly generated target position at each attempt. The test configuration parameters are: ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$), a swarm size of 20 individual's and a maximum iteration number of 5,000. The random target positions are generated within a circle of radius (1), as in Fig. 11a. The fitness of each solution is returned and subject to a statistical analysis using the distribution fitting Matlab tool.

Statistics analysis showed that the probability to obtain a solution with a fitness lower that 3×10^{-17} is 100 %, this means that for any random target position IK-FA will generate at 100 % a solution with a fitness lower that 3×10^{-17} . We can conclude that that for any target position within the definition space of the system, here a circle of radius (1), we are sure that an inverse kinematics solution exists and we are also sure at 100 % that this solution has a position error of 5.4722×10^{-9} as in Fig. 11b.

Table 3 Computing time with a stop condition (fitness = or <1e-12)

Computing time (s)	Swarm size	Iteration of convergence
6.7354192×10^{-4}	10	2,226 (min 1,711, max 2,569)
$7.63017534 \times 10^{-4}$	20	2,399 (min 2,300, max 2,661)
$1.46698068 \times 10^{-3}$	30	2,400 (min 2,110, max 2,605)
$2.10200931 \times 10^{-3}$	40	2,377 (min 2,100, max 2,535)
$2.54710426 \times 10^{-3}$	50	2,390 (min 2,230, max 2,546)
$3.50012342 \times 10^{-3}$	60	2,453 (min 2,175, max 2,569)

Fig. 11 IK-FA with random target positions. **a** Plot of solutions with random target. **b** Density of probability distribution of the fitness function

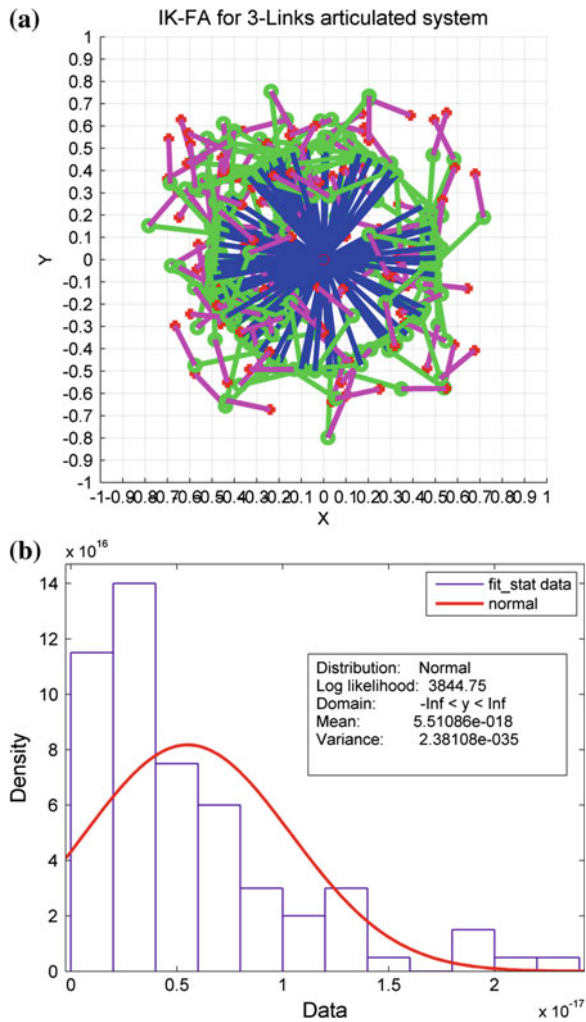


Table 4 Computing time of IK-FA compared to CCD for similar conditions

IK method	Position error	Computing time (s)
IK-FA	10^{-4}	2.1918973×10^{-3}
CCD	10^{-4}	0.4329001
IK-FA	10^{-8}	2.3394578×10^{-3}
CCD	10^{-8}	0.7112541

4.3 Comparison with CCD Method

After investigating the key aspects of IK-PSO, a comparative test with CCD inverse kinematics method is conducted for a 3 links articulated system with three revolute joints as in Fig. 1. The test is conducted under the same conditions using IK-FA and the CCD method. CCD was selected because it is a reference method for inverse kinematics solver and is assumed to be a real time IK method, real time methods are time stressed and have a large panel of application [37]. For this test both algorithms were asked to solve the inverse kinematics problem for the target point x_t (0.700, -0.500), for a given position error.

Results showed that for both configurations the IK-FA is faster than CCD, these results were established for 10^{-4} and 10^{-8} error positions, meaning that IK-FA fitness were respectively of 10^{-8} and 10^{-16} . Test were conducted using unconstrained IK-FA variant. Results showed that IK-FA clearly rendered solutions in a limited time compared to CCD, see Table 4 for details.

5 Conclusions and Future Directions

In this paper a new heuristic method for inverse kinematics based on Firefly Algorithm is proposed, IK-FA. It is based on the firefly algorithm and the forward kinematics model of a robotic system. The method is proposed for a human like articulated system, HLAS, while it could be generalized to any kind of robotics. The paper focuses on the IK-FA convergence capacities as well as the impact of FA parameters on the quality of the solutions. A set of good parameters for IK-FA was also established.

As conclusions: IK-FA, is a valuable solver for inverse kinematics, while parameter fitting is still a challenging problem. For a Given set of parameters, the heuristic converges to a static fitness value within a fix maximum number of iterations, in this work about 4,500 for ($\alpha = 0.02$, $\beta = 0.02$, $\gamma = 0.8$, $\delta = 0.997$). IK-FA has a fair convergence time, for the tested configurations, the average was about (2.3394×10^{-3}) s with a position error around ($3.116e-08$) over 100 tests. The algorithm showed also evidence of robustness over the target position, since for all conducted tests with a randomly generated target positions, IK-FA achieved a solution with a position error lower or equal to 5.4722×10^{-9} .

The investigation of the impact of the swarm size, showed that whatever is the swarm size, from 10 to 60, IK-FA convergences. Meanwhile it has been established in this work that as the swarm size increases the variance of the obtained solutions decreases. This means that the probability of finding a solution closer to the mean is higher. When the swarm size increases the computing time does so. A balance, between swarm size and computing time, need to be defined, in this work 20 FA individuals is an interesting choice.

Further developments are needed to deeply investigate the impact of FA variants on IK-FA. The implementation of IK-FA as an inverse kinematics solver of a robotic system such in [29, 31] should be introduced soon.

In This paper IK-FA was introduced as new heuristic inverse kinematics solver for constrained and unconstrained problems. The experimental investigations were limited to the unconstrained variant with an application to an articulated system composed by 3 links and 3 revolute joints. The impact of constraints on performances and computing time are under developments.

Acknowledgment The authors would like to acknowledge the financial support of this work by grants from General Direction of Scientific Research (DGRST), Tunisia, under the ARUB program.

References

1. Ammar, B., Chouikhi, N., Alimi, A.M., Chérif, F., Rezzoug, N., Gorce, P.: Learning to walk using a recurrent neural network with time delay. In: *Artificial Neural Networks and Machine Learning–ICANN*, pp. 511–518. Springer, Heidelberg (2013)
2. Asfour, T., Dillmann, R.: Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. In: *Intelligent Robots and Systems (IROS 2003)*, vol. 2, pp. 1407–1412 (2003)
3. Azevedo, C., Andreff, N., Arias, S.: BiPedal walking: from gait design to experimental analysis. *Mechatronics* **14**(6), 639–665 (2004)
4. Buckley, K.A., Simon H., Brian C.H.T.: Solution of inverse kinematics problems of a highly kinematically redundant manipulator using genetic algorithms. *IET*, pp. 264–269 (1997)
5. Buss, S.R.: Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE J. Robot. Autom.* **17** (2004)
6. Çavdar, T., Mohammad, M., Milani, R.A.: A new heuristic approach for inverse kinematics of robot arms. *Adv. Sci. Lett.* **19**(1), 329–333 (2013)
7. Chiaverini, S., Siciliano, B., Egeland, O.: Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Trans. Control Syst. Technol.* **2**(2), 123–134 (1994)
8. De Jong, K.A., Spears, W.M.: An analysis of the interacting roles of population size and crossover in genetic algorithms. In: *Parallel Problem Solving from Nature*, pp. 38–47. Springer, Heidelberg (1991)
9. Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. *IEEE Comput. Intell. Mag.* **1** (4), 28–39 (2006)
10. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, vol. 1, pp. 39–43 (1995)
11. Eberhart, R.C., Shi, Y.: Particle swarm optimization: developments, applications and resources. In: *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 1, pp. 81–86 (2001)

12. Edison, E., Shima, T.: Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms. *Comput. Oper. Res.* **38**(1), 340–356 (2011)
13. Juang, J.G.: Fuzzy neural network approaches for robotic gait synthesis. *IEEE Trans. Syst. Man Cybern. B Cybern.* **30**(4), 594–601 (2000)
14. Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N.: A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artif. Intell. Rev.* 1–37 (2012)
15. Kuffner, J., Nishiwaki, K., Kagami, S., Inaba, M., Inoue, H.: Motion planning for humanoid robots. In: *Robotics Research*, pp. 365–374. Springer, Heidelberg (2005)
16. Kulpa, R., Multon, F.: Fast inverse kinematics and kinetics solver for human-like figures. In: *Proceedings of Humanoids*, pp. 38–43 (2005)
17. Lander, J., CONTENT, G.: Making kine more flexible. *Game Developer Mag.* **1**, 15–22 (1998)
18. Łukasik, S., Żak, S.: Firefly algorithm for continuous constrained optimization tasks. *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*, pp. 97–106. Springer, Heidelberg (2009)
19. MATLAB Statistics Toolbox User's Guide (2014). The MathWorks Inc. http://www.mathworks.com/help/pdf_doc/stats/stats.pdf
20. Mohamad, M.M., Taylor, N.K., Dunnigan, M.W.: Articulated robot motion planning using ant colony optimisation. In: *3rd International IEEE Conference on Intelligent Systems*, pp. 690–695 (2006)
21. Pant, M., Gupta, H., Narayan, G.: Genetic algorithms: a review. In: *National conference on frontiers in applied and computational mathematics (FACM-2005)*, Allied Publishers, p. 225, 04–05 Mar 2005
22. Pérez-Rodríguez, R., Marcano-Cedeño, A., Costa, Ú., Solana, J., Cáceres, C., Opisso, E., Gómez, E.J.: Inverse kinematics of a 6 DoF human upper limb using ANFIS and ANN for anticipatory actuation in ADL-based physical neurorehabilitation. *Expert Syst. Appl.* **39**(10), 9612–9622 (2012)
23. Pham, D.T., Castellani, M., and Le Thi, H.A.: Nature-inspired intelligent optimisation using the bees algorithm. In: *Transactions on Computational Intelligence XIII*, pp. 38–69. Springer, Heidelberg (2014)
24. Pollard, N.S., Hodgins, J.K., Riley, M.J., Atkeson, C.G.: Adapting human motion for the control of a humanoid robot. In *Proceedings of IEEE International Conference on Robotics and Automation, ICRA'02*, vol. 2, pp. 1390–1397 (2002)
25. Rokbani, N., Alimi, A.M.: Inverse kinematics using particle swarm optimization, a statistical analysis. *Procedia Eng.* **64**, 1602–1611 (2013)
26. Rokbani, N., Alimi, A.M.: IK-PSO, PSO inverse kinematics solver with application to biped gait generation. *Int. J. Comput. Appl.* **58**(22), 33–39 (2012)
27. Rokbani, N., Alimi, A.M., Cherif, B.A.: Architectural proposal for an intelligent humanoid. In: *Proceedings of IEEE Conference on Automation and Logistics* (2007)
28. Rokbani, N., Benbousaada, E., Ammar, B., Alimi, A.M.: Biped robot control using particle swarm optimization. In: *IEEE International Conference Systems on Man and Cybernetics (SMC)*, pp. 506–512 (2010)
29. Rokbani, N., Boussada, E.B., Cherif, B.A., Alimi, A.M.: From gaits to ROBOT, A Hybrid methodology for A biped Walker. *Mobile Robotics: Solutions and Challenges*. In: *Proceedings of Clawar*, vol. 12, pp. 685–692 (2009)
30. Rokbani, N., Cherif B.A., Alimi, A.M.: Toward intelligent biped-humanoids gaits generation. In: Choi, B. (eds.) *Humanoids*. Chap 14, InTech (2009)
31. Rokbani, N., Zaidi, A., Alimi, A.M.: Prototyping a biped robot using an educational robotics kit. In: *IEEE International Conference on Education and E-learning Innovations*. Sousse, Tunisia (2012)
32. Rutkowski, L., Przybyl, A., Cpalka, K.: Novel online speed profile generation for industrial machine tool based on flexible neuro-fuzzy approximation. *IEEE Trans. Industr. Electron.* **59**(2), 1238–1247 (2012)

33. Schmidt, V., Müller, B., Pott, A. Solving the forward kinematics of cable-driven parallel robots with neural networks and interval arithmetic. In: *Computational Kinematics*, pp. 103–110. Springer, Netherlands (2014)
34. Tchoń, K., Jakubiak, J.: Endogenous configuration space approach to mobile manipulators: a derivation and performance assessment of Jacobian inverse kinematics algorithms. *Int. J. Control* **76**(14), 1387–1419 (2003)
35. Tchon, K., Jakubiak, J.: Jacobian inverse kinematics. In: *Advances in Robot Kinematics: Mechanisms and Motion*, p. 465 (2006)
36. Tevatia, G., Schaal, S.: Inverse kinematics for humanoid robots. In: *Proceedings of IEEE International Conference on Robotics and Automation, (ICRA'00)*, pp. 294–299 (2000)
37. Tolani, D., Goswami, A., Badler, N.I.: Real-time inverse kinematics techniques for anthropomorphic limbs. *Graph. Models* **62**(5), 353–388 (2000)
38. Van den Bergh, F., Engelbrecht, A.P.: Effects of swarm size on cooperative particle swarm optimizers (2001)
39. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. In: *ACM Transactions on Graphics (TOG)*, ACM, vol. 28, No. 3, p. 63 (2009)
40. Xu, Q., Li, Y.: Error analysis and optimal design of a class of translational parallel kinematic machine using particle swarm optimization. *Robotica* **27**(1), 67–78 (2009)
41. Yang, X.S. (2010). Firefly algorithm, Levy flights and global optimization. In: *Research and Development in Intelligent Systems XXVI*. Springer, London, 209–218
42. Yang, X.S.: Firefly algorithm, stochastic test functions and design optimisation. *Int. J. Bio-Inspired Comput.* **2**(2), 78–84 (2010)
43. Yang, X.S.: Firefly algorithms for multimodal optimization. In: *Stochastic algorithms: foundations and applications*, pp. 169–178, Springer, Heidelberg (2009)
44. Zaidi, A., Rokbani, N., Alimi, A.M.: A hierarchical fuzzy controller for a biped robot. In: *Proceedings of ICBR 2013. Sousse, Tunisia* (2013)
45. Zaidi, A., Rokbani, N., Alimi, A.M.: Neuro-Fuzzy gait generator for a biped robot. *J. Electron. Syst.* **2**(2), 48–54 (2012)
46. Zhang, X., Nelson, C.A.: Multiple-criteria kinematic optimization for the design of spherical serial mechanisms using genetic algorithms. *J. Mech. Des.* **133**(1) (2011)