An Adaptive Fuzzy Self-Tuning Inverse Kinematics Approach for Robot Manipulators

Ahmed Elmogy *,, Yassine Bouteraa *,***, Wael Elawady ****

** Computer Engineering Department, Prince Sattam Bin Abdelaziz University, KSA (email:a.elmogy@psau.edu.sa)*
*** Department of Computers and Control Engineering, Tanta University, Egypt*
(email: wael_ahmed@f-eng.tanta.edu.eg*)*
**** University of Sfax, Tunisia (email:y.bouteraa@psau.edu.sa)*

Abstract: In order for a robot manipulator to reach a desired position, an accurate knowledge of kinematics is required. Also, the Jacobian matrix of the robot manipulator should be nonsingular. However, when the robot deals with objects of unknown parameters, the overall kinematics becomes uncertain and changing. Furthermore, the non-singularity of the Jacobian matrix cannot be guaranteed. Fuzzy logic control is a good candidate technique to deal with uncertain kinematics, and Jacobian matrix. Nevertheless, the conventional fuzzy logic control is not adequate to develop a robust and efficient solution for the inverse kinematic problem. In this paper, a new adaptive fuzzy self-tuning control system for robot manipulators is developed. The developed system proposes two methods for reducing the numbers of rules and number of fuzzy inputs which significantly reduce the computational complexity. The developed simulations conducted on 2 and 3 DOFs robot manipulators show the effectiveness of the proposed approach.

Keywords: Robot manipulator, Inverse kinematics, Adaptive Control, Fuzzy logic, Jacobian Matrix.

1. INTRODUCTION

A robot manipulator is like a human arm. It consists of a series of links connected together by joints (Adelhedi et al., 2015; Zhou et al., 2019). When it comes to control or motion planning of robot manipulators, two main problems must be considered; forward and inverse kinematics (Elawady et al., 2020; Kutuk et al., 2017; Batista et al., 2020). In the forward kinematics, the position and orientation of the robot end effector is determined given the joint variables; namely joint angles for revolute joints, and link extensions for prismatic joints. While in inverse kinematics, the joint variables are computed to position the manipulator end effector at a desired location. Thus, the forward and inverse kinematics tackles the mapping between joint space and the Cartesian space. The forward kinematic solution is simple and straightforward, while the solution of inverse kinematics is nonlinear, uncertain and impossible to find unique solution in many cases. Thus, many efforts have been seen in the literature working towards finding the approximate inverse kinematics model (Tarokh et al., 2007; Raheem et al., 2016; Dinh, 2009). However, this is neither efficient nor sufficient for many robotic applications. When it comes to deal with certain type of applications that require tracking complex paths, finding an accurate inverse kinematics model of robot manipulators becomes a must (Xanthidis et al., 2018)

The complexity of inverse kinematics problem (IKP) depends mainly on the robot manipulator geometry and its nonlinear model. The nonlinear model of the robot manipulator represents the mapping relationship between Cartesian space and joint space. In spite of the complexity of the IKP, many robotic applications need a very accurate solution in order to achieve the required tasks efficiently and reliably (Hasan et al., 2011). Many methods have been developed in the literature working towards the solution of the IKP. These methods may be mainly categorized as algebraic methods (Perez et al., 2005; Selig, 2013), numerical methods (Xu et al., 2010; Olsen et al., 2011), and iterative methods (Manan et al., 2018; Ignacy et al., 2013). The algebraic methods work on finding closed form solution for the IKP which is considered the most effective one. However, the closed form solution can be obtained only for non-redundant robots with special geometry. For general serial manipulators, the closed form solutions are very difficult or impossible to guarantee (El-Sherbiny et al., 2017). Also, the numerical methods are very slow especially for high DOFs robot manipulators. On other hand, the iterative methods work on finding an approximate mapping between the robot Cartesian and joint spaces. The most important advantage of iterative methods over closed form and numerical methods is that they can provide the joint-Cartesian mapping fast and thus are best suited for many applications.

Soft computing techniques are among the popular iterative approaches. Much attention has been paid to solve IKP for robot manipulators using soft computing techniques (Hasan et al., 2011; Koker, 2013; Alavandar et al., 2008) as they provide good solutions without the need for complex calculations. This includes fuzzy logic, neural networks, and genetic algorithms approaches.

Generally, the ability of neural networks to provide the nonlinear Input and output system model makes it a good candidate to solve the IKP. Thus, the neural networks can be used to provide a good mapping between the joint space and the Cartesian space. Several neural networks structures have been used to deal with the IKP (Hasan et al., 2011; Koker et al., 2013; Duka, 2014). However, the main disadvantage of using neural networks to solve the inverse kinematics problem is that the training of the designed network is done based on the data collected from the robot forward kinematics. Collecting the forward kinematics data is done offline before incorporating the robot in the required task. This puts some constrains on using such type of approaches in the real time applications or in applications that require fast adaptation of the robot controller to the environment changes.

Also, Fuzzy logic has been extensively used in many robotic applications including the robot manipulators (Ching et al., 2017). This mainly comes from the powerfulness of fuzzy logic in dealing with uncertain and inaccurate data. The main core of the fuzzy controller is the rule database which should be built very carefully as it produces the required robotic behaviors. In order for the fuzzy controller to generate decisions, it goes through main processes; fuzzification, rules inference and aggregation, and defuzzification. For the inverse kinematics problem, some fuzzy like controllers have been developed (Mahmoodabadi et al., 2019; Mary et al., 2016). Although fuzzy controllers are effective in modeling nonlinear and complex systems, they depend mainly on developing a list of rules which is very challenging as it depends on human experiences. Choosing the inputs, outputs, and the rule base of the fuzzy system is very essential to get an efficient fuzzy control system. In order to alleviate the difficulty of dealing with the IKP of robot manipulators, hybrid approaches that combine neural networks and fuzzy logic are developed (Alavandar et al., 2008; Duka, 2015). The main idea of the developed neuro- fuzzy systems is using the input-output data generated from the forward kinematics to train the neural networks and then using the output of the neural networks to fire the fuzzy rules. Again, the neural network is trained offline which may be not suitable for some real time robotics applications.

In order to overcome the limitations of the approaches mentioned above, a new adaptive fuzzy self-tuning approach is developed in this paper to solve the IKP. The proposed approach works on reducing the number of fuzzy inputs and number of developed fuzzy rules in order to get a simple, very accurate and fast control system which is very suitable for complex robot manipulators' applications. Furthermore, the developed control system adapts online to any changes in the tasks given to the robot or in the environment the robot works in. Compared to other adaptive inverse kinematics algorithms like in (ref), the proposed adaptive fuzzy solution is very simple and robust. The developed solution depends on the heuristic knowledge and thus no need for the complex mathematical computations as in other solutions. In (Elawady et al., 2020) for example, we proposed an adaptive solution for the IKP. However, the proposed solution in our previous work (Elawady et al., 2020) depends on the computation of the mathematical kinematics. Also, the proposed solution like most inverse kinematic solution is based on Lypanov theory which is stable for certain operating conditions. Working towards finding a more simple and robust algorithm than the other proposed approaches, an adaptive fuzzy self-tuning approach is proposed in this paper. Differing from the solution proposed in (Elawady et al., 2020) and other solutions, the developed solution in this paper is stable and robust over a wide range of operating conditions. Furthermore, the proposed adaptive solution is faster, more accurate, and simpler than other approaches as will be illustrated in the next sections.

The rest of this paper is structured as follows. The characteristics of the IKP are introduced in section 2. Sections 3, and 4 present the characteristics of proposed fuzzy self-tuning inverse kinematics approach. Some simulation results are presented in section 5. Conclusions and future work are drawn in section 6.

2. INVERSE KINEMATICS ALGORITHM

The pose of manipulator end-effector can be specified as:

$$x_e = [p_e \quad \varphi_e]^T \tag{1}$$

Where p_e and φ_e represent the end-effector position and orientation respectively. x_e is dependent on joint variables and thus equation (1) can be written as:

$$x_e = k(q) \tag{2}$$

The nonlinear vector function $k(q)$ represents the calculation of the operational space variables given the joint space variables q. By the differentiation of (1), and (2), we get:

$$\dot{x}_e = \left[\dot{p}_e \quad \dot{\varphi}_e\right]^T = J_A(q)\dot{q} \tag{3}$$

Where $\dot{x}_e = \left[\dot{p}_e \;\; \dot{\varphi}_e\right]^T$ represents end-effector velocity and $\dot{q}$ is the vector of joint velocities. $J_A(q)$ is the Jacobian matrix which is given by:

$$J_A(q) = \frac{\partial k(q)}{\partial q} \tag{4}$$

In order to control the robot manipulator, it is necessary to specify the end-effector trajectory in terms of joints' position, velocity and acceleration. Conversely, the robot manipulator is inherently a nonlinear second order system. Therefore, we need to develop an algorithm that allows the inversion of the end-effector trajectory into the equivalent joint positions, velocities and accelerations. The differentiation of equation (3) gives:

$$\ddot{x}_e = J_A(q)\ddot{q} + \dot{J}_A(q,\dot{q})\dot{q} \tag{5}$$

Equation (5) represents the relationship between the accelerations of joint and operational spaces. Under the assumption of a square and nonsingular Jacobian matrix $\mathrm{J_A(q)}$, equation (5) can be rewritten as:

$$\ddot{q} = J_A^{-1}(q)\,(\,\ddot{x}_e - \dot{J}_A(q,\dot{q})\,\dot{q} \tag{6}$$

The velocities and positions can be computed by the integration of equation (6). However, this leads to a drift of the solution (Siciliano et al., 2009). Consequently, it is worth to define the operational space error as follows:

$$e = x_d - x_e \tag{7}$$

Where x_d is the desired end-effector pose.

The time derivative of (7) is:

$$\dot{e} = \dot{x}_d - \dot{x}_e \tag{8}$$

The time derivative of (8) is:

$$\ddot{e} = \ddot{x}_d - \ddot{x}_e \tag{9}$$

This in view of (5),

$$\ddot{e} = \ddot{x}_d - J_A(q)\ddot{q} - \dot{J}_A(q,\dot{q})\dot{q} \tag{10}$$

Equation (10) describes the error evolution over time. To obtain the inverse kinematics model, the joint acceleration vector $\ddot{q}$ is computed in terms of the error e. $\ddot{q}$ can be chosen as:

$$\ddot{q} = J_A^{-1}(q)\,(\ddot{x}_d + K_d\dot{e} + K_p e - \dot{J}_A(q,\dot{q})\,\dot{q}\,) \tag{11}$$

Which leads to the equivalent linear differential equation of the operational space error:

$$\ddot{e} + K_d\dot{e} + K_p e = 0 \tag{12}$$

The robot is asymptotically stable If K_p and K_d matrices are positive definite. The eigenvalues of K_p and K_d will affect the speed of convergence of error along the trajectory. The inverse kinematics model in equation (11) is visually illustrated in Fig. 1.

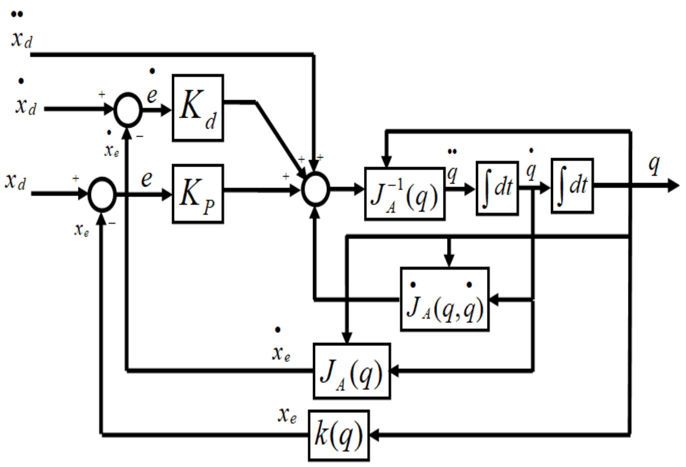


Fig. 1. The inverse kinematics model.

3. ADAPTIVE FUZZY-SELF TUNING INVERSE KINEMATICS ALGORITHM (AFSTIK)

The Jacobian matrix for the kinematic model of robot manipulator is generally assumed to be known (Cheah et al., 2005; Cheah et al., 2003). Therefore, the stability of the control system cannot be guaranteed for uncertain kinematics from joint space to Cartesian space. Although the kinematic model of the robot manipulator has sometimes sufficient accuracy, the parameters of the kinematic model changes as the robot interacts with its environment. Fuzzy logic is one of the intelligent techniques that implements human's knowledge able to model uncertainty and inaccurate data. Fuzzy logic can be used as a standalone technique or as a methodology to tune the parameters of other controllers. An adaptive self-tuning control structure is developed in this paper to solve the IKP efficiently. The proposed structure consists of two controller levels. The upper level is the fuzzy logic controller used as a supervisory controller to tune and supervise the parameters of the lower level controller which is the conventional PD controller. The goal of the lower level controller is to deliver the control actions to the robot manipulator. The proposed controller can adapt efficiently to any changes in the environment. Furthermore, the developed fuzzy controller structure can compensate for inaccurate kinematic model of the robot manipulator. The block diagram of the developed adaptive fuzzy-self tuning inverse kinematics approach (AFSTIK) is shown in Fig. 2.

The proposed supervisory fuzzy controller uses IF-THEN rules to tune the control gains K_p and K_d according to the values of the error $e(t)$ and derivative of error $\dot{e}(t)$ of the robot manipulator as described in equations (7) & (8). $K_p=diag\{k_{p1}, K_{p2},\ldots\ldots K_{pn}\}$, and $K_d=diag\{k_{d1}, K_{d2},\ldots\ldots K_{dn}\}$ are used as consequents for each rule. The range of the inputs $(e,\dot{e})$ is {-0.1, 0.1} and for the output control gains K_p and K_d is {1, 1000}. The fuzzy membership functions for $(e,\dot{e})$ are chosen to be triangle shape defined as {NL (Negative Large), NM (Negative Medium, NS (Negative Small), Z (Zero), PS (Positive Small), PM (Positive Medium), PL (Positive Large)} as shown in Fig. 3. Similarly, the fuzzy membership functions for (K_p and K_d) are chosen to be triangle shape defined as {VVL (Very Very Low), VL (Very Low), L (Low), M (Medium), H (High), VH (Very High) and VVH (Very Very High).

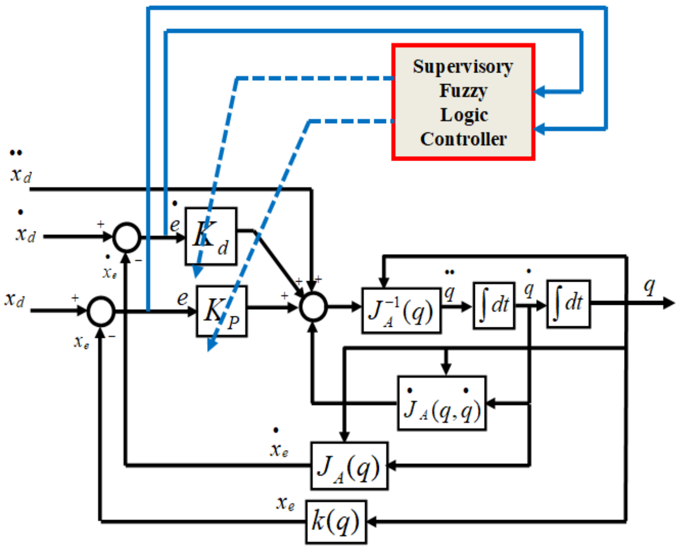


Fig. 2. The adaptive fuzzy inverse kinematics structure.

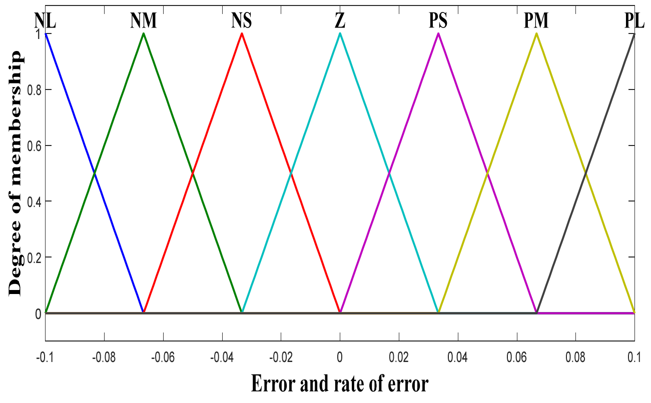


Fig. 3. Membership functions of e and $\dot{e}$.

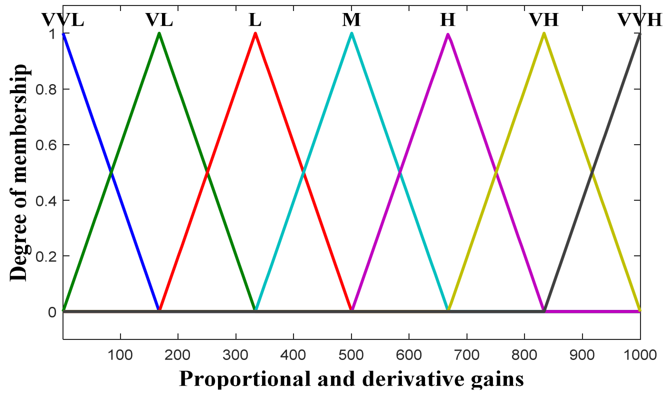


Fig. 4. Membership functions of K_p and K_d.

The proposed adaptive fuzzy control law can be expressed as:

$$R^{(i)} \ : \ IF \ e(t) \ is \ E_1^i \ and \ \dot{e}(t) \ is \ E_2^i$$

$$THEN \ \ K_p \ is \ G^i \ \ and \ \ K_d \ is \ H^i$$

Where E_1^i and E_2^i represent the labels of the input fuzzy sets and G^i and H^i are the labels of the output gains fuzzy sets. $i = 1, 2, \ldots, p$ denotes the rule number. Table 1 shows the rule base of the proposed supervisory fuzzy controller. The crisp values of the outputs are computed using the centre average defuzzification method.

Table 1. ΔK_p & ΔK_d supervisory fuzzy rule table.

$\dot{e}(t)$ \ $e(t)$	NL	NM	NS	Z	PS	PM	PL
NL	M	L	VL	VVL	VL	L	M
NM	H	M	L	VL	L	M	H
NS	VH	H	M	L	M	H	VH
Z	VVH	VH	H	M	H	VH	VVH
PS	VH	H	M	L	M	H	VH
PM	H	M	L	VL	L	M	H
PL	M	L	VL	VVL	VL	L	M

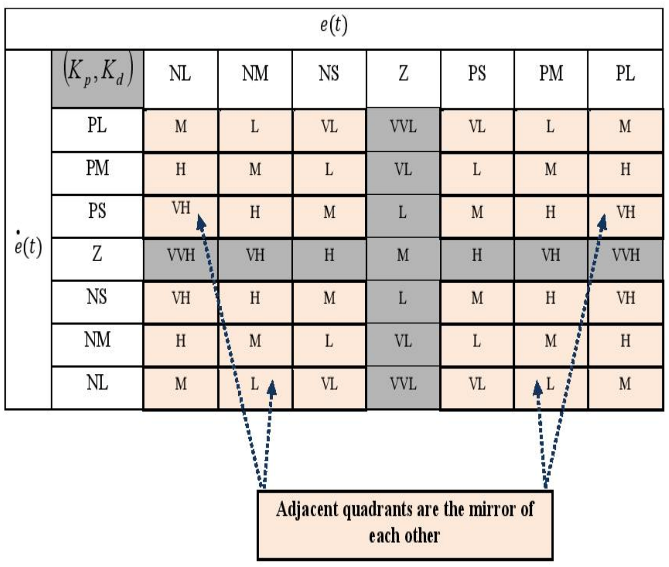
4. REDUCED RULE BASE SINGLE INPUT ADAPTIVE FUZZY-SELF TUNING INVERSE KINEMATICS ALGORITHM (RRBSI-AFSTIK)

In the AFSTIK approach proposed in the previous section, 49 fuzzy rules are used to tune the output gains K_p and K_d of the lower level controller. This high number of rules increases the complexity and computational time of the proposed control system which restricts using the proposed system to simple robot manipulators. In order to extend the proposed control system to be used with complex manipulators and real time applications, a rule base reduction method is proposed in this section. Furthermore, the number of inputs to the proposed adaptive fuzzy self-tuning inverse kinematics system is reduced to only one input.

Reconsidering the rules shown in Table 1, it can be inferred through vigilant observation that the adjacent quadrants are the mirrors of each other as illustrated in Figure 5. This leads to the reduction of the rule base of the proposed fuzzy controller. Two methods can be used to reduce the previous rule base. The first method works by considering the absolute value of the error $e(t)$ and the derivative of error $\dot{e}(t)$ as the inputs of the fuzzy controller. The rules of one quadrant are used to represent all cases represented in all quadrants. This will reduce the number of rules to only 16 rules as shown in Table 2.

The second method is motivated by the approach proposed in (Jae, 2001) for reducing the number of fuzzy controller inputs. The fuzzy inputs $e(t)$ and $\dot{e}(t)$ can be replaced by the magnitude difference d given as:

$$d(t) = |e(t)| - |\dot{e}(t)| \tag{13}$$



	$e(t)$						
$\dot{e}(t)$ / (K_p, K_d)	NL	NM	NS	Z	PS	PM	PL
PL	M	L	VL	VVL	VL	L	M
PM	H	M	L	VL	L	M	H
PS	VH	H	M	L	M	H	VH
Z	VVH	VH	H	M	H	VH	VVH
NS	VH	H	M	L	M	H	VH
NM	H	M	L	VL	L	M	H
NL	M	L	VL	VVL	VL	L	M

Adjacent quadrants are the mirror of each other

Fig. 5. The mirror image similarity of adjacent quadrants.

Table 2. Reduced rule for ΔK_p & ΔK_d of supervisory fuzzy logic controller.

$\lvert\dot{e}(t)\rvert$ \ $\lvert e(t)\rvert$	Z	PS	PM	PL
PL	VVL	VL	L	M
PM	VL	L	M	H
PS	L	M	H	VH
Z	M	H	VH	VVH

The corresponding block diagram of reduced rule base single input adaptive fuzzy self-tuning inverse kinematics algorithm (RRBSI-AFSTIK) is shown in Fig. 6. By taking the variable d to be the input of the new fuzzy system, the developed fuzzy system is now a single-input system and the rule base can be reduced to be as shown in Table 3 using the following linguistic fuzzy form:

$$R^{(p)} \ :: \ IF \ d \ is \ D^p \quad THEN \ \ K_p \ is \ G^p \ and \quad K_d \ is \ H^p$$

Where D^p represents the label of the input fuzzy sets. G^p and H^p are the output gains' labels. $p = 1, 2, 3,4,5,6,7$ denotes the number of the fuzzy rules. Using a single-input to the supervisory fuzzy controller leads to a great reduction in the number of rules as shown in table 3 which in turn reduce the computational complexity of the control algorithm. The centre average defuzzification method is used to compute the crisp values of the outputs.

Table 3. Reduced single input supervisory fuzzy logic controller rules.

$d(t)$	NL	NM	NS	Z	PS	PM	PL
(K_p, K_d)	VVL	VL	L	M	H	VH	VVH

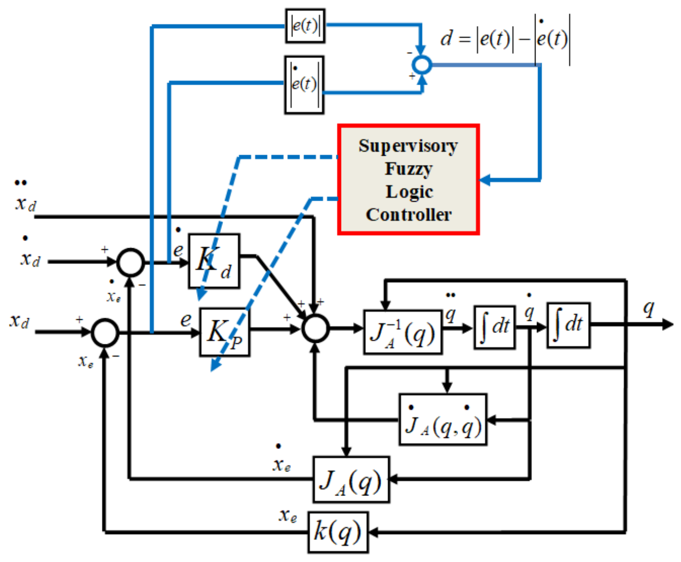


Fig. 6. Block diagram of the of the proposed RRBSI-AFSTIK approach.

5. SIMULATIONS AND RESULTS

In this section some simulations and results are presented to validate and assess the performance of the proposed RRBSI-AFSTIK approach, two different robot configurations are used; two-link robotic arm and three-link robotic arm.

5.1 Two-link robot arm

The direct kinematics of the 2 DOF robotic arm shown in Fig. 7 can be chosen as follows (Siciliano et al., 2009):

$$x_e = p_e = k(q) = \begin{bmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \end{bmatrix} \quad (14)$$

The Jacobian of this 2 DOF manipulator is deduced as:

$$J_A(q) = \frac{\partial k(q)}{\partial q} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \end{bmatrix} \quad (15)$$

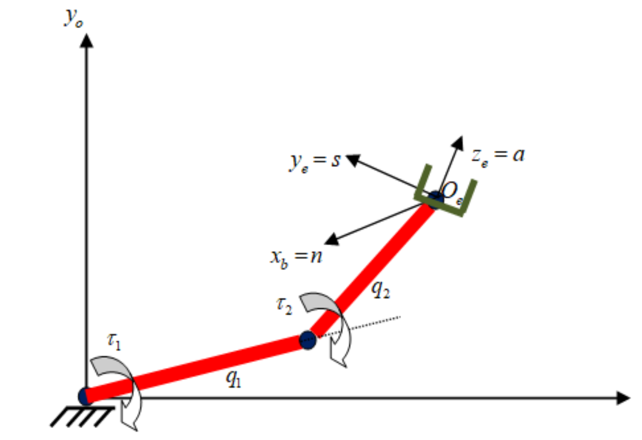


Fig. 7. Two-link robot manipulator.

By the differentiation of 15, $\dot{J}_A(q)$ is deduced as:

$$\dot{J}_A(q,\dot{q}) = \frac{d}{dt} J_A(q) = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}$$

$$\begin{aligned} J_{11} &= -a_1 c_1 \dot{q}_1 - a_2 c_{12}(\dot{q}_1 + \dot{q}_2) \\ J_{12} &= -a_2 c_{12}(\dot{q}_1 + \dot{q}_2) \\ J_{21} &= -a_1 s_1 \dot{q}_1 - a_2 s_{12}(\dot{q}_1 + \dot{q}_2) \\ J_{22} &= -a_2 s_{12}(\dot{q}_1 + \dot{q}_2) \end{aligned} \quad (16)$$

Where the link lengths are $a_1 = a_2 = 1$ m and:

$$\begin{aligned} c_1 &= \cos q_1 \qquad c_{12} = \cos(q_1 + q_2) \\ s_1 &= \sin q_1 \qquad s_{12} = \sin(q_1 + q_2) \end{aligned} \quad (17)$$

An eight trajectory shown in Fig. 8 is chosen as the desired end effector trajectory. This trajectory is chosen to show the effectiveness of the proposed approach in achieving complex tasks. The desired coordinates of the chosen trajectory are given as follows (Siciliano et al., 2009):

$$p_d(t) = \begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} 0.4 \sin(0.05\,\pi\, t) \\ 1.5 \sin(0.025\,\pi\, t) \end{bmatrix} \quad (18)$$

The initial posture of the robot end-effector is assumed to be at $q(0) = [0 \quad 0.9\,\pi]^T$ rad. Figure 9 and Fig. 10 show the X and Y coordinates position tracking errors. As seen, the proposed RRBSI-AFSTIK approach outperforms the conventional inverse kinematics approach (CIK), and the continuous second order sliding mode inverse kinematics algorithm (CSOSM-AIK) proposed in (Elawady et al., 2020). Furthermore, the proposed approach rise time is further reduced compared with the other approaches which makes our approach is best fit in real time applications. The actual trajectories of the robot end-effector for the proposed approach, the CIK and CSOSM-AIK approaches are shown in Fig. 11. As seen in Fig. 11, the developed RRBSI-AFSTIK approach outperforms the other approaches.

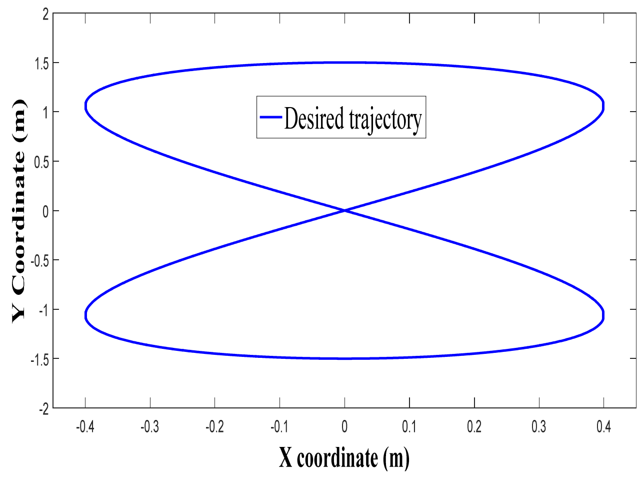


Fig. 8. Desired trajectory in task space.

In order to more assess the tracking performance of the proposed RRBSI-AFSTIK approach, several performance metrics are considered. These metrics includes the integral absolute error (*IAE*), integral of squared error (*ISE*), and integral time multiplied absolute error (*ITAE*) (Zhu et al., 2009) as given in equation (19). Table 4 shows a comparison between the proposed RRBSI-AFSTIK approach and the CIK approach in terms of these metrics.

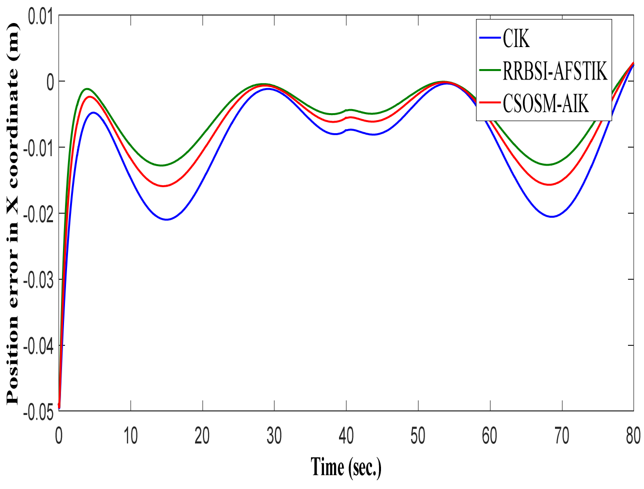


Fig. 9. The X-coordinate position error.

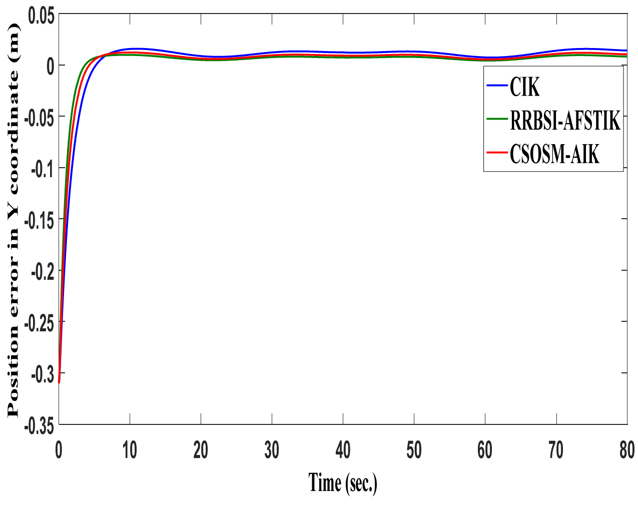


Fig. 10. The Y-coordinate position error.

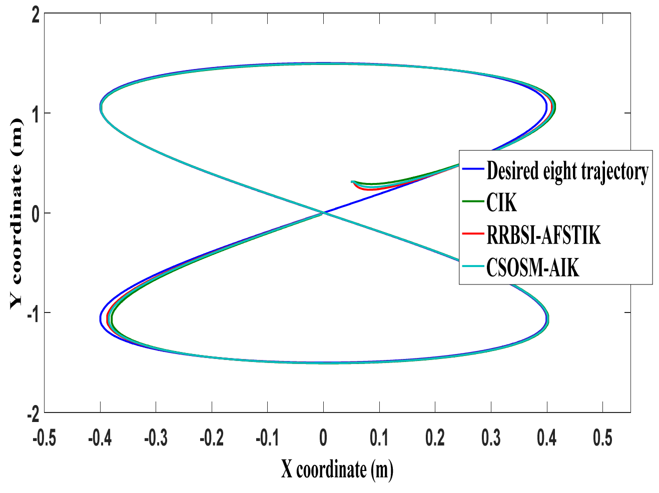


Fig. 11. The end effector trajectories of 2 DOF robot.

Figs. 12, and 13 show how the adaptive gains K_p, and K_d are changing during tracking the desired manipulator trajectory. These adaptive gains are self-tuned using the proposed fuzzy logic rules in table 3. Thus, the adaptive gains are updated using a reduced single input supervisory fuzzy logic controller rules.

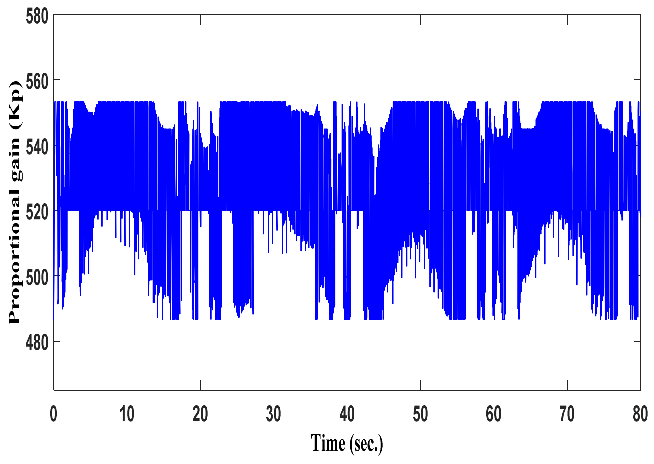


Fig. 12. Adaptive K_p for the 2 DOF robot.

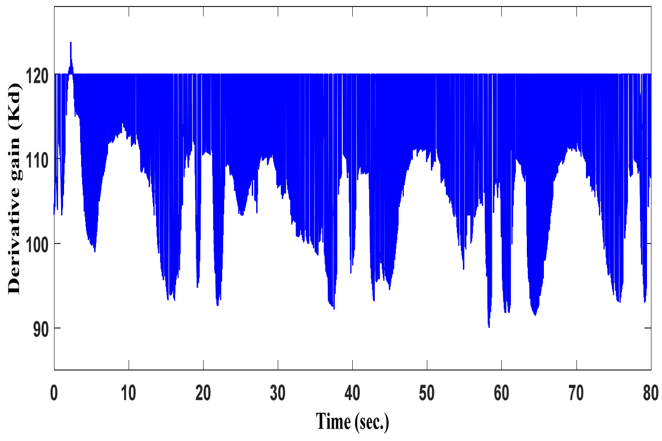


Fig. 13. Adaptive K_d for the 2 DOF robot.

The changes in the adaptive gains K_p, and K_d shown in Figs. 12 & 13 show that the proposed adaptive controller is covering a wide range of operating conditions and the gains K_p, and K_d are changing at every point of the search space to best track the desired complex eight trajectory.

The indices *IAE*, *ISE* and *ITAE* are computed as follows:

$$IAE = \int |e(t)|\, dt$$

$$ISE = \int e(t)^2\, dt \tag{19}$$

$$ITAE = \int t|e(t)|\, dt$$

Table 4. Performance comparison for two link robot.

Algorithm	***IAE***		***ISE***		***ITAE***	
	X	**Y**	**X**	**Y**	**X**	**Y**
CIK	80.9	100.3	60.6	20.6	140.2	138.2
CSOSM-AIK	70.3	70.8	5.6	5.4	86.2	85.8
RRBSI-AFSTIK	65.5	60.6	2.9	1.9	82.5	80.2

As seen in Table 4, the proposed RRBSI-AFSTIK approach provides a superior tracking performance compared to the CIK and CSOSM-AIK approaches.

5.2 Three-link robot arm

The developed RRBSI-AFSTIK approach is secondly examined on the 3 DOF robot arm shown in Fig. 14 (Siciliano et al., 2009).

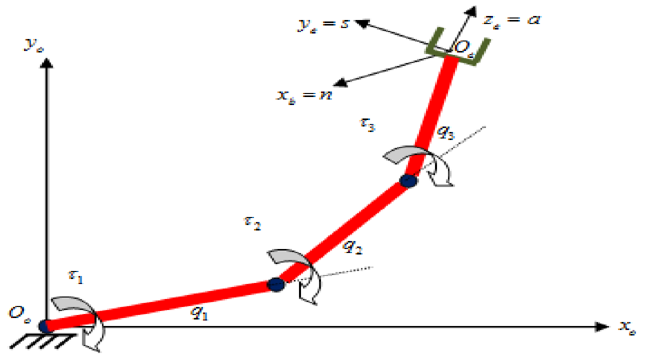


Fig. 14. Three-link robot manipulator.

The kinematics for this robot is chosen as follows (Siciliano, 2009):

$$x_e = k(q) = \begin{bmatrix} a_1c_1 + a_2c_{12} + a_3c_{123} \\ a_1s_1 + a_2s_{12} + a_3s_{123} \\ q_1 + q_2 + q_3 \end{bmatrix} \tag{20}$$

The Jacobian of this 3 DOF manipulator is deduced as:

$$J_A(q) = \frac{\partial k(q)}{\partial q}$$
$$= \begin{bmatrix} -a_1s_1 - a_2s_{12} - a_3s_{123} & -a_2s_{12} - a_3s_{123} & -a_3s_{123} \\ a_1c_1 + a_2c_{12} + a_3c_{123} & a_2c_{12} + a_3c_{123} & a_3c_{123} \\ 1 & 1 & 1 \end{bmatrix} \tag{21}$$

By the differentiation of 21, $\dot{J}_A(q)$ is deduced as:

$$\dot{J}_A(q,\dot{q}) = \frac{d}{dt}J_A(q) = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ 0 & 0 & 0 \end{bmatrix}$$
$$J_{11} = -a_1c_1\dot{q}_1 - a_2c_{12}(\dot{q}_1 + \dot{q}_2) - a_3c_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)$$
$$J_{12} = -a_2c_{12}(\dot{q}_1 + \dot{q}_2) - a_3c_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3),$$
$$J_{13} = -a_3c_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)$$
$$J_{21} = -a_1s_1\dot{q}_1 - a_2s_{12}(\dot{q}_1 + \dot{q}_2) - a_3s_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)$$
$$J_{22} = -a_2s_{12}(\dot{q}_1 + \dot{q}_2) - a_3s_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)$$
$$J_{23} = -a_3s_{123}(\dot{q}_1 + \dot{q}_2 + \dot{q}_3) \tag{22}$$

Where the link lengths are $a_1 = a_2 = a_3 = 1$ m. A butterfly path shown in Fig. 15 is chosen as another complex trajectory. The X and Y coordinates of this desired end-effector path are described by:

$$p_d(t) = \begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} 0.1\cos(t)\,e^{\cos(t)} - 2\cos(4t) - \sin^5(t/12) + 1 \\ 0.2\sin(t)\,e^{\cos(t)} - 2\cos(4t) - \sin^5(t/12) + 1 \end{bmatrix} \tag{23}$$

The initial posture of the robot end-effector is assumed to be at $q(0) = \begin{bmatrix} \pi & \frac{-\pi}{2} & \frac{-\pi}{2} \end{bmatrix}^T$ rad.

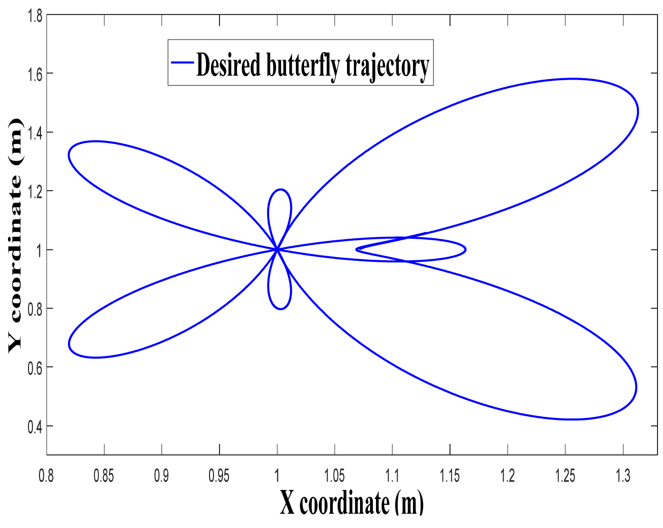


Fig. 15. Desired trajectory in task space.

Fig. 16 and Fig. 17 depict the X and Y coordinates position tracking errors. As seen, the proposed RRBSI-AFSTIK approach outperforms the CIK and CSOSM-AIK approaches in terms of error and speed even if the trajectory is very complex as chosen. The trajectory of the end-effector is shown in Fig. 18.

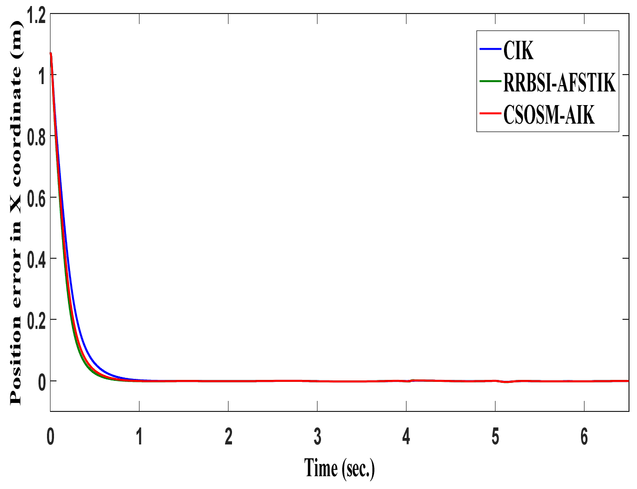


Fig. 16. The X-coordinate position error.

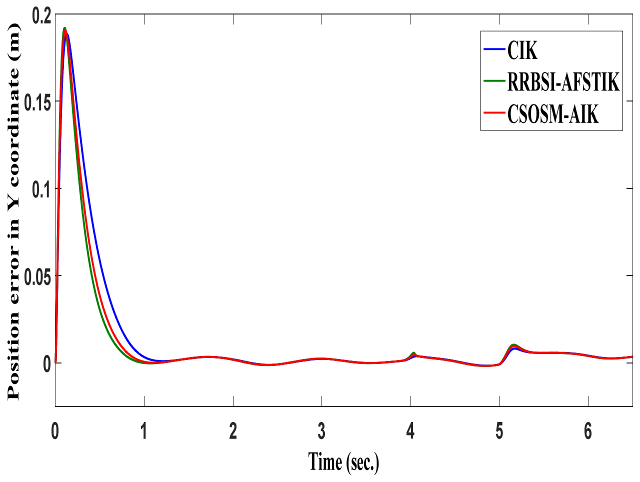


Fig. 17. The Y-coordinate position error.

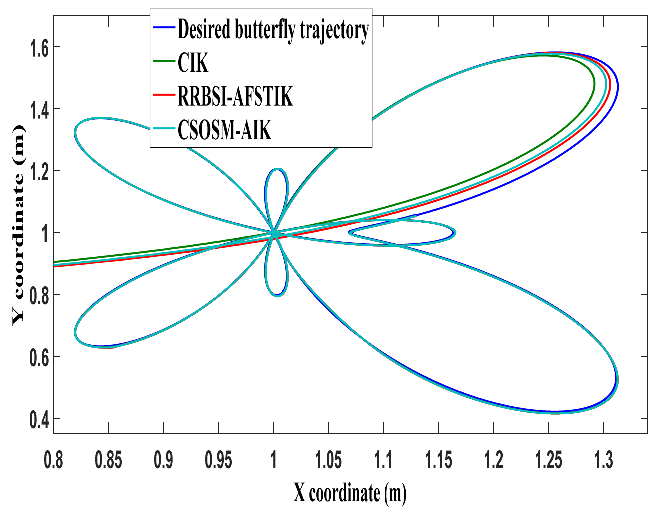


Fig. 18. The end effector trajectories of 3 DOF robot.

Figs. 19, and 20 show how the adaptive gains K_p, and K_d are changing during tracking the desired manipulator trajectory. These adaptive gains are self-tuned using the proposed fuzzy logic rules in table 3. Thus, the adaptive gains are updated using a reduced single input supervisory fuzzy logic controller rules.

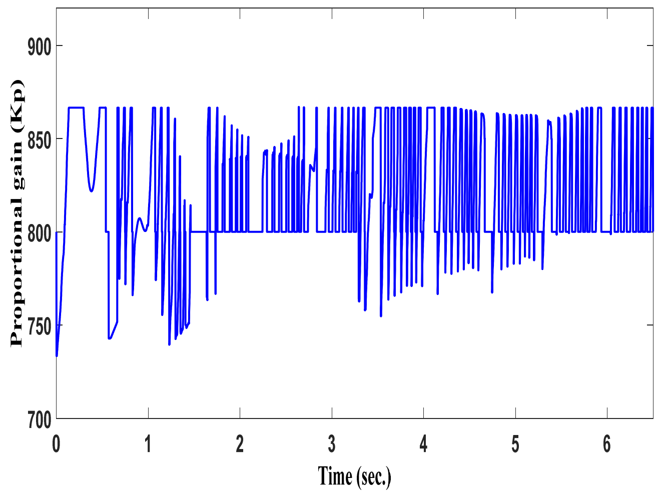


Fig. 19. Adaptive K_p for the 3 DOF robot.

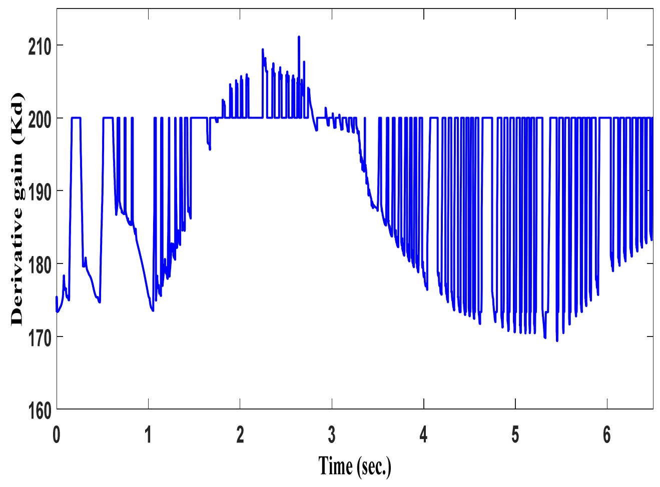


Fig. 20. Adaptive K_d for the 3 DOF robot.

The changes in the adaptive gains K_p, and K_d shown in Figs. 19 & 20 show that the proposed adaptive controller is covering a wide range of operating conditions and the gains K_p, and K_d are changing at every point of the search space to best track the desired complex butterfly trajectory.

Table 5 shows a comparison of tracking response between the proposed RRBSI-AFSTIK approach, the CIK, and CSOSM-AIK approaches in terms of *ISE*, *IAE* and *ITAE* described by equation (19). The data of Table 5 illustrates that the proposed approach provides a superior performance compared to the other approaches.

Table 5. Performance comparison for three link robot.

Algorithm	***IAE***		***ISE***		***ITAE***	
	X	**Y**	**X**	**Y**	**X**	**Y**
CIK	95.2	79.9	15.2	60.5	139.5	142.4
CSOSM-AIK	83.004	67.35	5.34	6.87	82.221	85.325
RRBSI-AFSTIK	58.1	62.9	2.6	3.9	78.5	80.9

6. CONCLUSIONS

As the inverse kinematic problem is very essential to consider for any robot manipulator control system, many techniques have been developed trying to find good solutions for this problem. However, this track is still challenging and needs more efforts especially when working with complex tasks/robots. This paper presents a new adaptive self-tuning fuzzy solution for inverse kinematics for industrial robot manipulators. The proposed solution is distinguished from the other existing approaches by its simplicity, fastness, and robustness. These features make the developed fuzzy approach a good one to consider in many robotic applications. The conducted simulations and results showed the effectiveness and adaptation of the proposed approach compared to other approaches in the literature.

Funding: This research was funded by the Deanship of Scientific Research at Prince Sattam Bin Abdulaziz University, KSA under the research project # 2019/01/11117.

REFERENCES

Adelhedi, F., Jribi, A., Bouteraa, Y. and Derbel, N. (2015). Adaptive sliding mode control design of a SCARA robot manipulator system under parametric variations. *Journal of Engineering Science and Technology Review*, Volume 5, No. 15, 17-123.

Alavandar S., and Nigm M. (2008). Neuro-fuzzy based approach for inverse kinematics solution of Industrial Manipulators, *International Journal of Computers Communications & Control*, Volume 3, 224-234.

Batista J., Souza D. ,dos Reis L., Barbosa A. and Araújo R (2020). Dynamic model and inverse kinematic identification of a 3-DOF manipulator using RLSPSO, *Sensors Journal*, 20, 416.

Cheah, C., Hirano, M., Kawamura, S., Arimoto, S. (2003). Approximated Jacobian control for robots with uncertain kinematics and dynamics. *IEEE Trans. Robot. Autom.,* 19, 692–702.

Cheah, C., and Liaw H. (2005). Inverse Jacobian regulator with gravity compensation: stability and experiment, *IEEE Trans. Robot*., 741–747.

Ching C., Chien-Chun W., Yi Tun W., and Po Tung W. (2017). Fuzzy logic controller design for intelligent robots, *Mathematical Problems in Engineering*, Volume 2017.

Dinh H. (2009). Approximation of the inverse kinematics of a robotic manipulator using a neural network, *Computer Science*.

Duka A. (2015). ANFIS based solution to the inverse kinematics of a 3DOF planar manipulator, *Procedia Technology*, 19, 526-533.

Duka A. (2014). Neural network based inverse kinematics solution for trajectory tracking of a robotic arm, *Procedia Technology*, Volume 12, 12-20.

Elawady W., Bouteraa Y., and Elmogy A. (2020). An adaptive second order sliding mode inverse kinematics approach for serial kinematic chain robot manipulators. *Robotics journal*, 9(1), 4.

El-Sherbiny A, Elhosseini MA, and Haikal AY. (2017). A comparative study of soft computing methods to solve inverse kinematics problem, *Ain Shams Engineering journal.* 9:2535–2548.

Hasan A., Hayder M., Ahmad A. (2011). Neural networks' based inverse kinematics solution for serial robot manipulators passing through singularities, *InTech artificial neural networks-industrial and control engineering applications*, 459–78.

Ignacy D., and Michal O. (2013). A Comparison Of Jacobian–based methods of inverse kinematics for serial robot manipulators, *International Journal of Applied Mathematics in Computer science*, 23(2), 373-382.

Jae C. (2001). Design of single-input direct adaptive fuzzy logic controller based on stable error dynamics, *International Journal of Fuzzy Logic and Intelligent Systems*,

Koker R. (2013). Genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, *Inf. Sci.*, 528–543.

Kutuk, M., Taylan,M., and Canan, L. (2017). Forward and inverse kinematics analysis of Denso robot. *In Proceedings of the International Symposium of Mechanism and Machine Science*, Baku, Azerbaijan.

Mahmoodabadi M., and A. Ziaei (2019) .Inverse dynamics based optimal fuzzy controller for a robot manipulator via particle swarm optimization, *Journal of Robotics*, Volume 2019.

Manan K., Vinay P., and Ashish T. (2018). Comparative Study of iterative inverse kinematics methods for serial manipulators, *International Journal of Engineering Research & Technology (IJERT)*, 7(5).

Mary, T. Kara and A. H. Miry (2016). Inverse kinematics solution for robotic manipulators based on fuzzy logic and PD control, *Al-Sadeq International Conference on Multidisciplinary in IT and Communication Science and Applications (AIC-MITCSA)*, Baghdad, Iraq.

Olsen A., and Petersen H. (2011). Inverse kinematics by numerical and analytical cyclic coordinate descent. *Robotica*, 29(4),619–626.

Perez A., McCarthy J. (2005). Sizing a serial chain to fit a task trajectory using Clifford algebra exponentials, *IEEE international conference on robotics and automation. ICRA*.

Raheem F.A., Kareem A.R., and Humaidi, A.J (2016). Inverse kinematics solution of robot manipulator end-effector position using multi-neural networks. *Eng. Technol. journal.*, 34, 1360–1369.

Selig J. (2013). *Geometrical methods in robotics*. Springer Science & Business Media.

Siciliano, L. Sciavicco, L. Villani and G. Oriolo (2009). *Robotics, Modeling, Planning and Control*, Springer.

Tarokh M. and Kim M. (2007). Inverse kinematics of 7-DOF robots and limbs by decomposition and approximation, *IEEE Transactions on Robotics*, 23(3).

Xanthidis M., Kyriakopoulos K.J., Rekleitis, I. (2018). Dynamically efficient kinematics for hyper-redundant manipulators, *In Proceedings of the 24th Mediterranean Conference on Control and Automation*, Athens, Greece, 207–213.

Xu J., Wang W., and Sun Y. (2010). Two optimization algorithms for solving robotics inverse kinematics with redundancy. *J Control Theory Appl.*, 2010.

Zhou H., Chen R., Zhou S. and Liu Z. (2019). Design and analysis of a drive system for a series manipulator based on orthogonal-fuzzy PID control. *Electronics Journal. 8*(9), 1051.

Zhu H. , Li L. , Zhao Y. , Guo Y. ,Yang Y. (2009). CAS algorithm-based optimum design of PID controller in AVR system, *Chaos, Solitons & Fractals*. 42. 792-800.