Improved Hybrid Trajectory Tracking Algorithm for a 3-link Manipulator Using Artificial Neural Network and Kalman Filter

Dawon Joo and Kiwon Yeom
Department of Human Intelligence and Robot Engineering, Sangmyung University, South Korea
Email: koong1114@gmail.com, pragman@naver.com

***Abstract*—While inverse kinematics is used as a trajectory generator for the path tracking of the end effector of robots, precise control of terminal device is difficult due to accumulated tracking errors. Therefore, artificial neural network or Kalman filter-based inverse kinematics analysis methods have been proposed to minimize the tracking errors of inverse kinematics. However, generating the trajectory of end effectors based on such methods still contain tracking errors, making precise trajectory tracking difficult. To solve this issue, therefore, this study proposes the end effector path control algorithm using artificial neural network and Kalman filter. Furthermore, it demonstrates, through simulation results, that the proposed algorithm can track the trajectory effectively.**

***Index Terms*—inverse kinematics, artificial neural network, Kalman filter, robot manipulator**

I. Introduction

Inverse kinematics is a method to calculate the joint value corresponding to each joint of a robot so that the end effector can be operated in the Cartesian space, and various studies on tracking a robot trajectory through inverse kinematic analysis have been conducted.

To solve the inverse kinematics of robot manipulators, Iliukhin *et al*. (2017) and Bhave *et al*. (2013) used geometric and mathematical methods [1], [2]. Iliukhin *et. al*. (2017) solved the 5-link manipulator through mathematical methods [1], such as the Denavit–Hartenberg convention and the homogeneous transformation matrix and showed trajectory control through the MATLAB simulation. Bhave *et al*. (2013) determined the joint value with the location and direction of the end effector of a robot based on a geometric method in order to control the sliding model of the 3-link manipulator [2].

As stated by Duka (2013), feed-forward neural network was applied to determine the inverse kinematics solution of the 3-link manipulator [3], and a method to generate joint variables for the trajectory tracking of the end effector at a series of target points. In addition, Kumar and Irshad (2012) proposed a method to determine the inverse kinematics solution of 2-link robot using multilayer perceptron (MLP) and backpropagation [4]. However, in these studies, artificial neural network-based joint variables may have many errors when tracking the trajectory, which is problematic, as it leads to different results depending on training data.

On the contrary, other studies proposed methods that use Kalman filter for the trajectory tracking of end effectors of robots [5]-[7]. Song *et al*. (2018) calculated the error between the end effector and the target using image processing, and suggested using the Kalman filter to track the trajectory of robot end effector [5]. Takaba *et al*. (1996) applied the extended Kalman filter (EKF) twice and proposed a method that could track and control the location of the end effector and the joints of 2-link robot [6]. EKF showed the identical values estimated by the robot when the speed of the end effector was slow [4]; however, when its speed was faster, the curvature ratio of the link increased too much to determine a linearized model. Badamchizadeh *et al*. (2010) determined a dynamic model of 5-link robot and proposed EKF and unscented Kalman filter (UKF) to estimate the acceleration and jerk, which is the change of acceleration per unit hour, based on the measured location and speed [7].

This study derived the inverse kinematics joint variables of 3-link robot using artificial neural network, and, by entering the corresponding coordinators of the end effector into the initial vector of Kalman filter, the study proposes a hybrid trajectory tracking algorithm that can improve the trajectory tracking performance of the final 3-link robot end effector. By comparing the trajectory tracking performance between the robot, which is based on the proposed algorithm and the existing robot that used only the artificial neural network or Kalman filter, it verifies the effectiveness of the hybrid trajectory algorithm using artificial neural network and Kalman filter.

After the introduction, Chapter II in this article analyzes the forward kinematics and inverse kinematics of 3-link manipulator geometrically, introduces artificial neural network and Kalman filter, which are the algorithms used for the trajectory tracking of 3-link manipulator, and proposes a hybrid algorithm that combines artificial neural network and Kalman filter.

Manuscript received July 17, 2020; revised January 4, 2021.

Chapter III verifies and compares the performance of each algorithm through MATLAB simulation, and Chapter IV concludes the article.

II. Hybrid Trajectory Tracking Algorithm

A. *3-link Manipulator and Geometrical Analysis of Inverse Kinematics*

As shown in Fig. 1, a 3-link manipulator that moves on a x-y plane consists of the base fixed at the origin, three rotating joints, and the three links that connect these joints.

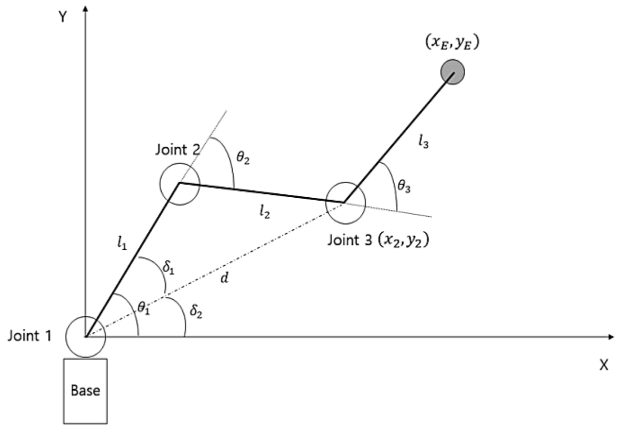


Figure 1. Schematic expression of 3-link planar manipulator

Each joint has limited ranges of motion as set below, and the length of each link is identical.

$$\theta_1 \in [0, \pi]$$

$$\theta_2 \in [-\pi, 0]$$

$$\theta_3 \in [-\frac{\pi}{2}, \frac{\pi}{2}]$$

$$l_1 = l_2 = l_3 = 2\text{m}$$

The kinematic analysis of the manipulator is divided into forward kinematic analysis and inverse kinematic analysis. The forward kinematic analysis determines the location (x_E, y_E) and direction (o_E) of the end effector when the state of the joint $(\theta_1, \theta_2, \theta_3)$ is given, and the forward kinematics equations (1) and (2), and the direction equation (3) of 3-link manipulator can be determined as follows.

$$x_E = l_1 cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad (1)$$

$$y_E = l_1 sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad (2)$$

$$o_E = \theta_1 + \theta_2 + \theta_3 \quad (3)$$

The inverse kinematic analysis determines the state of each joint $(\theta_1, \theta_2, \theta_3)$ when the location (x_E, y_E) and direction (o_E) of the end effector at the standard coordinate space are given, and the inverse kinematics equations (4–10) of 3-link manipulator based on the above region of operation are as follows (see Fig. 2).

$$x_2 = x_E - l_3\cos(o_E) \quad (4)$$

$$y_2 = y_E - l_3\sin(o_E) \quad (5)$$

$$\delta_2 = \text{atan}\left(\frac{y_2}{x_2}\right) \quad (6)$$

$$\theta_2 = -cos^{-1}\left(\frac{x_2^2 + y_2^2 - (l_1^2 + l_2^2)}{2l_1 l_2}\right) \quad (7)$$

$$\delta_1 = atan\left(\frac{l_2\sin(\theta_2)}{l_1 + l_2\cos(\theta_2)}\right) \quad (8)$$

$$\theta_1 = \delta_1 + \delta_2 \quad (9)$$

$$\theta_3 = o_E - (\theta_1 + \theta_2) \quad (10)$$

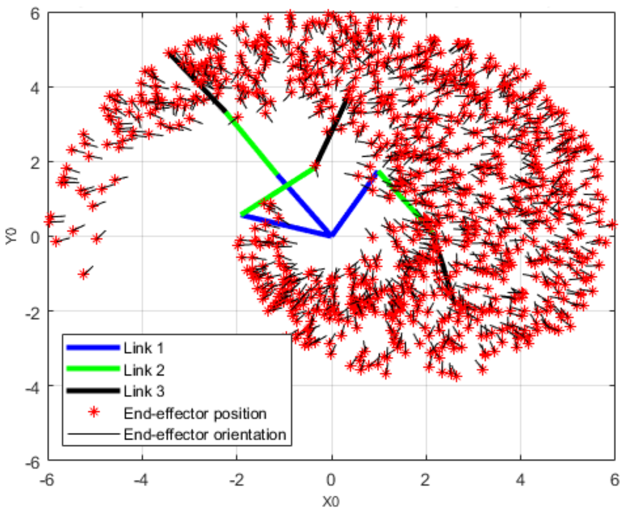


Figure 2. Workspace of 3-link planar manipulator.

B. *Inverse Kinematic Analysis of Artificial Neural Network*

First, the joint variables acquired from forward kinematics analysis of 3-link manipulator and the corresponding location and direction of the end effector are used to train artificial neural network. Then, in reverse, when the pose of the end effector is given, the inverse kinematic analysis of artificial neural network is conducted, which is about determining solution of the generalized joint variables from the trained artificial network. The inverse kinematic analysis of artificial neural network can be realized only by the input and output patterns of the system without the mathematical conditions or restrictions of *II.A* and can produce (or derive) appropriate outputs based on the learned values against unlearned input conditions.

To solve an inverse kinematics, the study applies feed-forward neural network (FFNN), as shown in Fig. 3. The structure of FFNN consists of the input layer, the hidden layer, the output layer, and the (calculation) processing flows from the input layer to the output layer—passing through only one hidden layer. While the number of hidden layers is unlimited in theory, it is reported that only one hidden layer is enough for the solution [8], [9]. In this study, the number of hidden layer nodes was set to 110 based on the preliminary test result to determine the suitable number of hidden layer nodes.

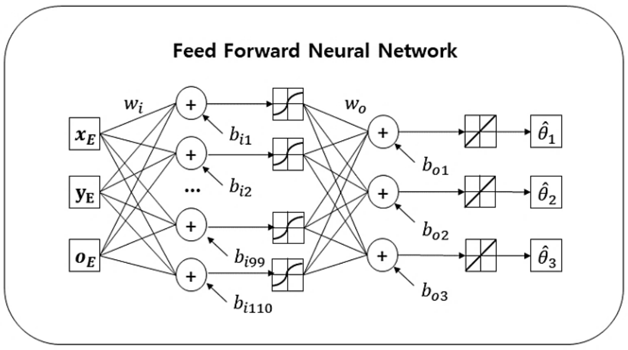


Figure 3. The structure of feed-forward neural network (FFNN).

The input layer has three nodes where the position of end effector $(x_E, y_E, o_E))$ is entered. Then, the input value is multiplied with the weight value (w_i) and the bias (b_i) is added to represent the activation function of hidden layer as equation (11).

$$z = w_i \cdot \begin{pmatrix} x_E \\ y_E \\ o_E \end{pmatrix} + b_i \tag{11}$$

The activation function of the hidden layer is a hyperbolic tangent function as shown in Equation (12) that returns the output between −1 and 1 with the origin as its center, and therefore, compared to a sigmoid function, its convergence speed is faster and the slope loss is smaller [10].

$$\tanh(z) = \frac{e^{2z}-1}{e^{2z}+1} = Val_h \tag{12}$$

The result of the hidden layer (Val_h) is multiplied to the weighting value of the output layer (w_o), and after the bias of the output layer (b_o) is added, the result is entered as the activation function of the output layer, which is a linear transfer function $(y = x)$. The output in the output layer is the joint variables $(\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3)$ or the movement toward the input pose, which consists of three nodes and is calculated as interpreted in equation (13).

$$\begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \hat{\theta}_3 \end{pmatrix} = w_o \cdot Val_h + b_o \tag{13}$$

The shape of such FFNN renews the weighting value and bias through the repetitive process of the LM (Levenberg–Marquardt) backpropagation algorithm and trains the artificial neural network. (Refer to [1] for technical details for the solution of FFNN-based inverse kinematics.)

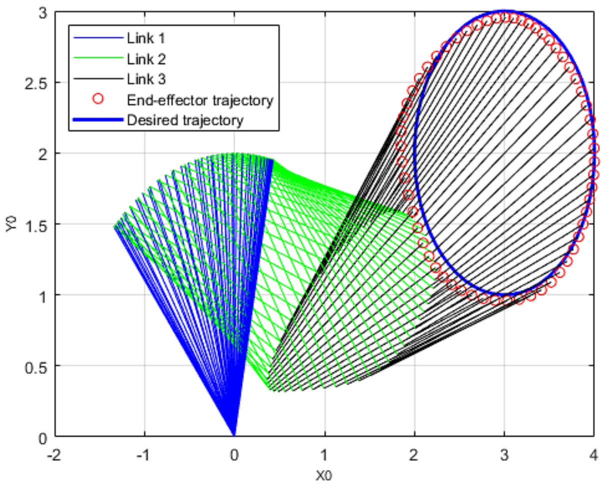


Figure 4. The solution of inverse kinematics for the trajectory tracking of 3-link robot using artificial neural network.

C. Manipulator Trajectory Tracking Based on Kalman Filter

The study uses linear Kalman filter for tracking of the trajectory of 3-link manipulator, and the state equation (14) and measurement equation (15) of Kalman filter can be simply shown as follows.

$$x_k = Ax_{k-1} + Bu_{k-1} + W_{k-1} \tag{14}$$

$$z_k = Hx_k + V_k \tag{15}$$

Here, x is the state vector of the manipulator, z_k is the measurement vector, A is the state matrix, B is the input matrix, u_{k-1} is the input vector, H is the measurement derivative matrix, W_{k-1} is the system model noise, and V_k is the measured noise.

To track the trajectory of 3-link manipulator based on Kalman filter, the state vector of the system was defined by the location and speed of the end effector being estimated, as shown in equation (16), and based on this, state equation (17) and measurement equation (1) are presented below, similar to equation (14) and (15) state equation. T indicates sampling time.

$$x_k = [p_x, p_y, v_x, v_y] \tag{16}$$

$$x_k = \begin{bmatrix} p_k \\ v_k \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ v_k \\ v_{y_k} \end{bmatrix}$$

$$= \begin{bmatrix} x_{k-1} + v_{x_{k-1}} \cdot T + \frac{1}{2} u_{k-1} \cdot T^2 \\ y_{k-1} + v_{y_{k-1}} \cdot T + \frac{1}{2} u_{k-1} \cdot T^2 \\ v_{x_{k-1}} + u_{x_{k-1}} \cdot T \\ v_{y_{k-1}} + u_{y_{k-1}} \cdot T \end{bmatrix} + W_{k-1} \tag{17}$$

$$= \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ v_{x_{k-1}} \\ v_{y_{k-1}} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ \frac{T^2}{2} \\ T \\ T \end{bmatrix} [u_{k-1}] + W_{k-1}$$

$$[z_k] = \begin{bmatrix} x_k \\ y_k \end{bmatrix} = = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ v_{x_k} \\ v_{y_k} \end{bmatrix} + V_k \qquad (18)$$

Kalman filter consists of the estimation and update stages, and its algorithm is as below (equations (19) – (23))

(1) Estimation stage

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \qquad (19)$$

$$P_k^- = AP_{k-1}A^T + Q \qquad (20)$$

(2) Update stage

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \qquad (21)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \qquad (22)$$

$$P_k = (I - K_k H)P_k^- \qquad (23)$$

Here, Q is the covariance matrix of the system model noise (W_{k-1})), R is the covariance matrix of the measured noise (V_k), and P is the error covariance matrix, where Q and R can be determined as follows.

$$Q = \begin{bmatrix} \frac{T^4}{4} & 0 & \frac{T^3}{2} & 0 \\ 0 & \frac{T^4}{4} & 0 & \frac{T^3}{2} \\ \frac{T^3}{2} & 0 & T^2 & 0 \\ 0 & \frac{T^3}{2} & 0 & T^2 \end{bmatrix}, \ R = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}. \qquad (24)$$

In the estimation stage, the state vector ($\hat{x}_{k-1}$) of the manipulator in the previous stage is used to estimate the state vector ($\hat{x}_k^-$) of the manipulator. In the update stage, the state vector ($\hat{x}_k^-$) of the current manipulator estimated and the measured location vector (z_k) in the estimation stage are used to update the state vector of the current manipulator.

D. *Robot Trajectory Tracking Algorithm Based on Artificial Neural Network-Kalman Filter*

To acquire precise trajectory tracking results for 3-link manipulator, this study proposes a hybrid algorithm that combines artificial neural network and Kalman filter (Fig. 5).

In this study, FFNN is learned by the joint variables (($\theta_1, \theta_2, \theta_3$)) randomly determined within the scope of the operation of 3-link manipulator and the pose of the end effector (x_E, y_E, o_E)) corresponding to each state variable. The measurement vector of Kalman filter (z_k) is the target location (x_d, y_d) of the end effector that the manipulator needs to reach, and direction (o_d) is required for the input to the artificial neural network. The direction (o_d) corresponding to the target location can be derived from equation (25).

$$o_d = tan^{-1}(\frac{y_d}{x_d}) \qquad (25)$$

Since artificial neural network analyses and processes complex nonlinear relations in parallel, it has excellent generation capacity, but as it verifies the state vector through the forward kinematics of the joint variables ($\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3$) which are the result from artificial neural network alone, considerable errors are found in the manipulator tracking performance, which makes precise tracking difficult. To address this problem, the study uses Kalman filter, a recursive filter that estimates the state of a dynamic linear system based on state vectors even if they contain errors. Since this article aims to precisely track the trajectory, it assumes uniform motion, and accordingly, sets the input vector of Kalman filter, which signifies acceleration, to 0.

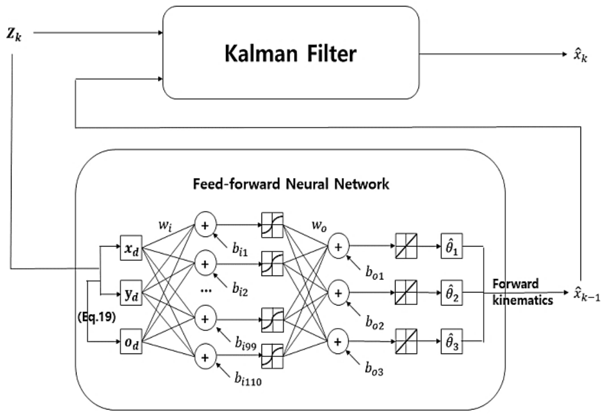


Figure 5. The structure of the hybrid manipulator control algorithm using FFNN and Kalman filter algorithm.

III. TEST RESULT

To verify the performance of the proposed artificial neural network-Kalman filter based trajectory tracking algorithm for 3-link manipulator, the study conducted a test in the following environment.

A. *Test Environment*

The hardware environment for the performance evaluation is as follows:

Hardware: Intel CORE i5 (2.64GHz), 8GB RAM

Software: MATLAB® /Simulink® 2019b

Tracking Trajectory: the trajectory of a circle with radius at 1 (Ground truth–equation (26))

$$(x-3)^2 + (y-2)^2 = 1^2 \qquad (26)$$

B. *Result of Artificial Neural Network-based Trajectory Tracking*

Shown in Fig. 6 is the result of the inverse kinematics based on artificial neural network of *II.A.* for 3-link manipulator trajectory tracking. Trajectory tracking starts from the point at (4, 2) where the red square is located and moves along the given counter-clockwise trajectory.

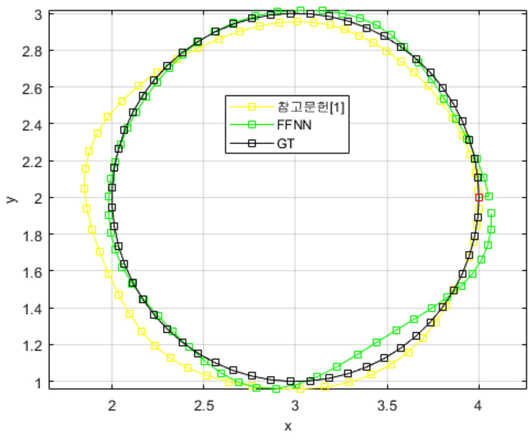


Figure 6. Comparison of FFNN and reference [1] of trajectory tracking using artificial neural network.

In this study, the trajectory tracking performance of artificial neural network-based robot was evaluated with the mean square error (MSE). MSE is defined, shown in equation (12), as the mean average of all values from the square of the difference between the estimated value ($\hat{Y}_i$) of the model and the measured value (Y_i), and the smaller the resulting value, the smaller is the error of the model.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{21}$$

To use the artificial neural network whose performance is more improved than that of the artificial neural network used by Duka (2014) [3], the study set a different number of hidden layer nodes against the identical 1000 input data, as shown in Table I, and examined the number of the hidden layer nodes to the optimal MSE.

When the number of input data is 1000, the MSE of No. 1 based on the method by reference [1] was 0.01660433, and the average MSE of No. 2 was 0.01491391 which was an approximately 10.182% increase in performance.

TABLE I. MSE Based on the Number of Hidden Layer Nodes of Artificial Neural Network

No.	Number of input data	Number of hidden layer nodes	Average MSE	Average time (sec)	Remarks
1	1000	100	0.01660433	0.6	Reference [1]
2	1000	110	0.01491371	1.9	FFNN
3	1000	120	0.01736290	1.5	
4	1000	130	0.01818620	2.3	

C. Result of Kalman Filter-based Trajectory Tracking

Fig. 7 shows the result of the Kalman filter-based 3-link manipulator trajectory tracking, where the trajectory tracking starts from the point at (4, 2), where the red square is located to the counter-clockwise trajectory. To virtually realize the instability of the actual manipulator, a normal distribution random error with the standard deviation by about 10 °is added to the joint variable in the proposed Kalman filter-based trajectory tracking, and the state vector from the forward kinematics is entered to Kalman filter. The system noise of Kalman filter is set to 1, and the measurement noise to 0.5.

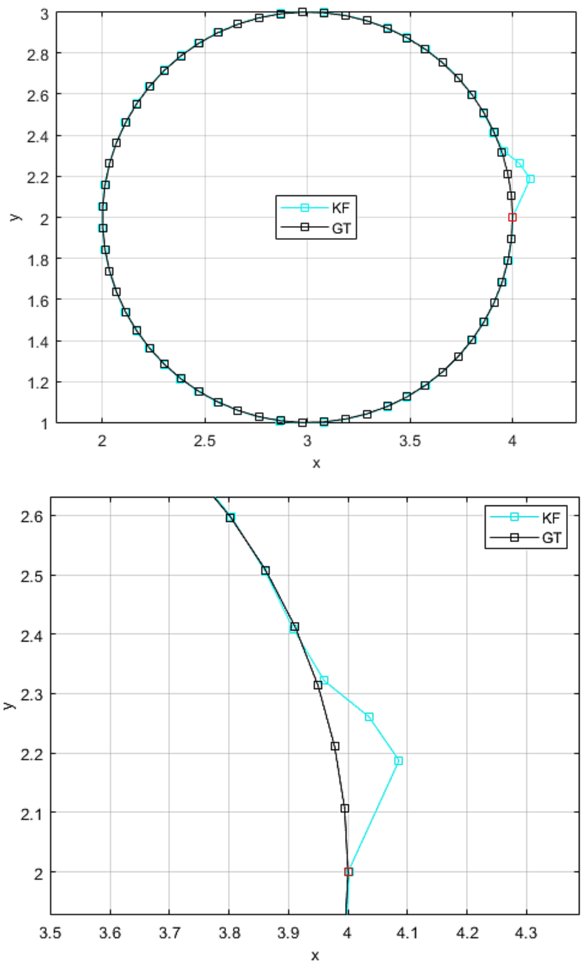


Figure 7. Kalman filter-based trajectory tracking. At the beginning part, the Kalman filter based trajectory control was distorted. However, it showed the algorithm can compensate the error quickly.

As shown in Fig. 7, the three initial stages of the trajectory tracking are deviated from the x-axis trajectory by ~0.1 m, but in the other areas, the control of the manipulator which is done by the excellent performance of Kalman filter can be identified.

D. Trajectory Tracking Results Based on the Hybrid Algorithm and Performance Comparison

Fig. 8 shows the result of the proposed hybrid algorithm (KFNN-KF).

Fig. 9 shows the performance comparison of the tracked trajectory by each algorithm where the number of the tracked locations that did not reach within the threshold location after the threshold location is determined (Left in Fig. 9). For the threshold location, the study set the difference in absolute value between the target location (x_d, y_d) and the tracked location to be 0.0005 m (0.5 mm), and if the tracked location of the end effector is below this value, it is deemed to have successfully passed the target location. Out of the total 60 location points (or locations), the number of location detection error using artificial network and Kalman filter was 59 and 52, respectively, with rates of 98% and 97% each, whereas that of the algorithm proposed in this study was 4 with an error rate of 4% which was a drastic decrease from the previous two.

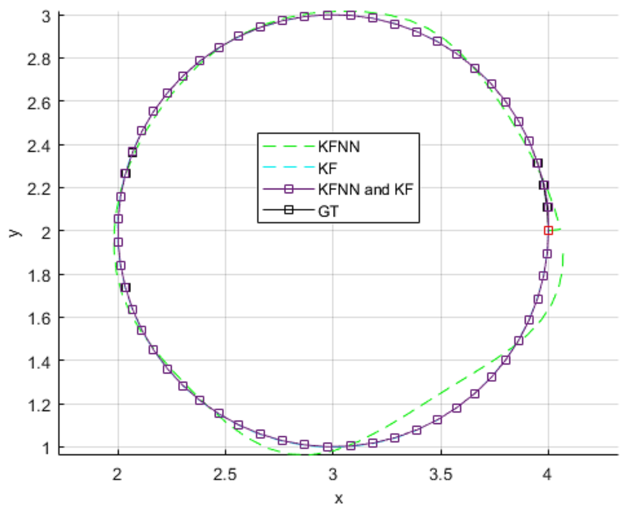
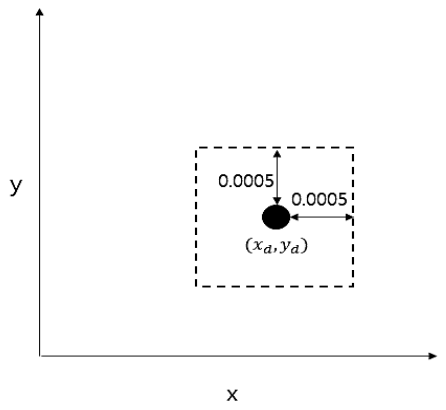


Figure 8. Comparison of the trajectory tracking for the hybrid algorithm, Kalman filter, and FFNN with respect to the ground truth.



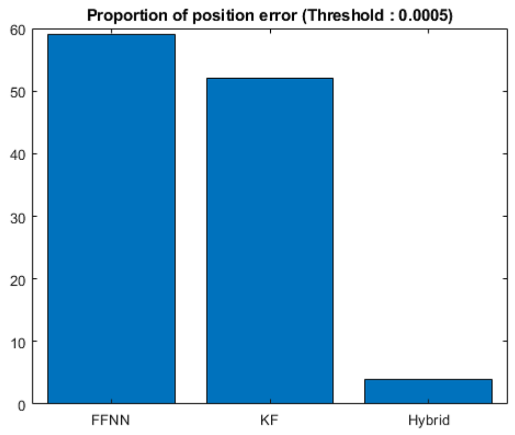


Figure 9. Threshold location against the target location, and the corresponding performance comparison among the algorithms.

IV. Conclusion

This study proposed Kalman filter and artificial neural network-based control algorithms for tracking the trajectory of a 3-link manipulator. Furthermore, it compared the proposed algorithm with the algorithm based on either Kalman filter or artificial neural network using MATLAB simulation. The inverse kinematics analysis based on artificial neural network is conducted via training data to make optimal decision or find estimates, and therefore, accurate tracking of the trajectory in an environment other than training data was difficult. Kalman filter allows for trajectory tracking by considering only the previous state value and the current state value, but it produced considerably different results depending on noise.

However, the Kalman filter- and artificial neural network-based control algorithm proposed in this study produced estimates that were closer to the measured values by comparing the estimates resulting from the trained artificial neural network to the previous state value of Kalman filter and the measured values, and thus, it showed the best performance among the three algorithms.

Further research is required to directly track the pose of the nonlinear end effector (x_E, y_E, O_E) by replacing Kalman filter, which was used in this study for the trajectory tracking toward the measurement vector, with extended Kalman filter (EKF), which is used for a nonlinear state estimation.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

All authors had conducted the research and had approved the final version.

Acknowledgment

This research is supported in part by 2020 research grant from Sangmyung University.

References

[1] V. N. Iliukhin, K. B. Mitkovskii, D. A. Bizyanova, A. A. Akopyan, “The modeling of inverse kinematics for 5 DOF manipulator,” *Procedia Eng.*, vol. 176, pp. 498-505, Jan. 2017.
[2] M. Bhave, L. Dewan, S. Janardhanan, “A novel third order sliding mode controller for the orientation and position of planar three link rigid robotic manipulator,” 2013.
[3] A. V. Duka, “Neural network based inverse kinematics solution for trajectory tracking of a robotic arm,” *Procedia Technol.*, vol. 12, no. 1, pp. 20-27, Jan. 2014.
[4] S. Kumar and K. Irshad, “Implementation of artificial neural network applied for the solution of inverse kinematics of 2-link serial chain manipulator,” *Int. J. Eng. Sci. Technol.*, vol. 4, no. 9, pp. 4012-4024, 2012.
[5] L. Song, Z. Duan, B. He, and Z. Li, “Application of federal Kalman filter with neural networks in the velocity and attitude matching of transfer alignment,” *Complexity*, 2018.
[6] K. Takaba, Y, Iiguni, and H. Tokumaru, “An improved tracking Kalman filter using a multilayered neural network,” *Math. Comput. Modell.*, vol. 23, no. 1-2, pp. 119-128, Jan. 1996.

[7] M. A. Badamchizadeh, I. Hassanzadeh, and M. A. Fallah, “Extended and unscented Kalman filtering applied to a flexible-joint robot with jerk estimation,” *Discrete Dyn. Nat. Soc.*, 2010.
[8] G. H. Ryu, J. D. Jung, “Inverse kinematics analysis of a binary robot manipulator using neural network,” *J. Korean Soc. Precis. Eng.*, vol. 16, no. 1, pp. 211-218. 1999.
[9] S. Y. Park, C. Y. Lee, “A Study on the forecasting of container volume using neural network,” *J. Navig. Port Res.*, vol. 26, no. 2, pp. 183-188 2002.
[10] B. Karlik, A. V. Olgac, “Performance analysis of various activation functions in generalized mlp architectures of neural networks, *Int. J. Artif. Intell. Expert Syst.*, vol. 1, no. 4, pp. 111-122. 2011.

Dawon Joo is currently studying in the department of human intelligence and robot engineering, Sangmyung University, South Korea. She pursues intelligent control algorithm for manipulators and is interested in HCI and HRI for robot control.

Kiwon Yeom is a professor in the department of human intelligence and robot engineering, Sangmyung University, South Korea**.** He is interested in intelligent robot control and swarm robot control. He is now pursing the intelligent control algorithm for the identical and multiple robots to be used in the disaster situation.