Design of Adaptive Kinematic Controller Using Radial Basis Function Neural Network for Trajectory Tracking Control of Differential-Drive Mobile Robot

Tran Quoc Khai¹ and Young-Jae Ryoo²

¹Department of Electrical Engineering, Mokpo National University, Mokpo, Korea

²Department of Electrical and Control Engineering, Mokpo National University, Mokpo, Korea

ljfis

Abstract

In this study, we propose an adaptive kinematic controller using a radial basis function neural network (RBFNN) to guide a differential-drive mobile robot (DDMR) during trajectory tracking. The kinematic controller is responsible for generating the reference values of the linear and angular velocities delivered to the robot actuators. With the fixed parameters of the kinematic controller, it is difficult for the robots to obtain acceptable performance. By using RBFNN and gradient descent method, the parameters of the kinematic controller can be updated online, thus providing smaller errors and better performance in applications. In this paper, the experimental results are presented to show the effectiveness of the proposed kinematic controller.

Keywords: Differential-drive mobile robot, Radial basis function neural network, Kinematic controller, Gradient descent method, Trajectory tracking

1. Introduction

Recently, there have been many research directions related to autonomous mobile robots [1], including trajectory tracking control [2]. An approach used to design the basic control law in trajectory tracking control is based on kinematic and dynamic features of the robot [3]. Therefore, there are two important controllers used for trajectory tracking: kinematic controller and dynamic controller [4, 5]. The former, as an outer loop controller, is responsible for controlling the trajectory, whereas the latter, as an inner loop controller, is used to control the velocities of the robot.

Using the inverse kinematics of a differential-drive mobile robot (DDMR), the kinematic control law proposed by Martins et al. [6] has been used in many research works [6–11]. However, almost all authors have focused on analyzing the research results related to the dynamic controller only when the parameters of the kinematic controller are constant. With the fixed parameters of the kinematic controller, the DDMR exhibits limited tracking performance, especially when the reference trajectory changes rapidly over time. Therefore, this study is focused on improving the performance of the kinematic controller, and the dynamic controller is designed by conventional PID (proportional-integral-derivative) control law. For that purpose, the new kinematic control method based on radial basis function neural network (RBFNN) is proposed.

Received: Oct. 20, 2019

Revised : Dec. 3, 2019

Accepted: Dec. 4, 2019

Correspondence to: Young-Jae Ryoo
(yjryoo@mkpu.ac.kr)

©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

This method allows estimating the position of the mobile robot via the RBFNN, based on which the parameters of the kinematic controller can be updated online to accommodate the characteristic variation of the process. In this paper, the model of the DDMR is introduced in Section 2. The tracking trajectory controller design is explained in Section 3. The method of designing an adaptive kinematic controller is discussed in Section 4. The experimental testing and discussions are presented in Section 5. Finally, the conclusion and results are presented in Section 6.

2. Differential-Drive Mobile Robot

2.1 Configuration

The DDMR consists of a car body equipped with two castors and a pair of coaxial driving wheels driven by two independent DC motors. By varying the control voltage on each DC motor, the DDMR can move in a straight or curved line or trace different trajectories. Figure 1 depicts the configuration of the DDMR, its parameters, and variables of interest. Here, θ is the robot orientation, φ_R and φ_L are rotating angles of the right and left wheels, respectively, W is the width of the robot body, and $2R$ is the diameter of the driving wheel.

The center-point C , visible on the axis between the driving wheels, is the center of mass of the robot.

The position of the DDMR can be defined in the inertial reference frame (X^I, Y^I) fixed in the plane, and the reference frame (X^R, Y^R) , rotated relative to the inertial reference frame by the orientation angle θ .

Figure 1. Model of differential-drive mobile robot.

2.2 Kinematic Model

A mobile robot is a typical nonlinear control system with non-holonomic constraints. Before we proceed with modeling and performing relevant analyses, the following assumptions are presented:

- A robot can only move forward and backward, not side-ward. In the rotating reference frame, this condition means that the velocity of the center-point C is zero along the lateral axis:

$$\dot{y}_c^R = 0. \quad (1)$$

In the inertial reference frame, Eq. (1) can be written as

$$-\dot{x}_c \sin \theta + \dot{y}_c \cos \theta = 0. \quad (2)$$

- The DDMR moves on horizontal surface with pure rolling constraint. With this condition, the velocities of the right and left wheels can be written as follows:

$$\begin{cases} v_R = \dot{x}_L + \dot{y}_L = R\dot{\varphi}_R, \\ v_L = \dot{x}_R + \dot{y}_R = R\dot{\varphi}_L. \end{cases} \quad (3)$$

Eq. (3) can be written in the terms of the velocity of center-point C in the inertial reference frame:

$$\begin{cases} v_R = \dot{x}_c \cos \theta + \dot{y}_c \sin \theta + \frac{W}{2}\dot{\theta} = R\dot{\varphi}_R, \\ v_L = \dot{x}_c \cos \theta + \dot{y}_c \sin \theta - \frac{W}{2}\dot{\theta} = R\dot{\varphi}_L. \end{cases} \quad (4)$$

Therefore, the linear and angular velocities of the DDMR are calculated as follows:

$$\begin{cases} v = \frac{v_R + v_L}{2} = \frac{R}{2}(\dot{\varphi}_R + \dot{\varphi}_L), \\ \omega = \frac{v_R - v_L}{W} = \frac{R}{W}(\dot{\varphi}_R - \dot{\varphi}_L). \end{cases} \quad (5)$$

From Eqs. (3) and (4), the forward kinematic model of the DDMR can be represented in two matrix forms in terms of point C velocities in the inertial reference frame as follows:

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{R}{2} \cos \theta & \frac{R}{2} \cos \theta \\ \frac{R}{2} \sin \theta & \frac{R}{2} \sin \theta \\ \frac{R}{W} & -\frac{R}{W} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix}, \quad (6)$$

Figure 2. Conventional trajectory tracking control scheme.

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (7)$$

3. Design of Trajectory Tracking Controller

The control scheme for trajectory tracking of the DDMR is based on the dynamic and kinematic of the DDMR, as shown in Figure 2. The reference trajectory generated by the block trajectory generator is sent to the kinematic controller. Here, the latter compares it to the robot's posture and generates the reference linear and angular velocities for the dynamic controller based on the kinematic control algorithm.

For the purpose of designing the kinematic controller, a point of interest (point *A*) is selected on the symmetry axis of the robot body displaced for distance *a* from center-point *C*, as shown in Figure 1. The kinematic equations of DDMR in terms of velocities at point *A* in the inertial reference frame from the Eq. (7) can be obtained.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & -a \sin \theta \\ \sin \theta & a \cos \theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (8)$$

The output of the kinematic model is the coordinates of point *A*. Hence,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \cos \theta & -a \sin \theta \\ \sin \theta & a \cos \theta \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \lambda \begin{bmatrix} v \\ \omega \end{bmatrix} \quad (9)$$

with

$$\lambda = \begin{bmatrix} \cos \theta & -a \sin \theta \\ \sin \theta & a \cos \theta \end{bmatrix}. \quad (10)$$

Therefore, the inverse kinematics is presented as follows:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\frac{1}{a} \sin \theta & \frac{1}{a} \cos \theta \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \lambda^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}. \quad (11)$$

The conventional kinematic controller proposed in the study [6] is given by

$$\begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} = \lambda^{-1} \begin{bmatrix} \dot{x}_d + l_x \tanh \left(\frac{k_x}{l_x} e_x \right) \\ \dot{y}_d + l_y \tanh \left(\frac{k_y}{l_y} e_y \right) \end{bmatrix}. \quad (12)$$

Here, $k_x > 0$ and $k_y > 0$ are the gains of the controllers, $l_x > 0$ and $l_y > 0$ are the saturation constants, $e_x = x_d - x$ and $e_y = y_d - y$ are the current position errors in the axes X^I and Y^I , respectively, and (x, y) and (x_d, y_d) are the current and the reference coordinates of the point of interest, respectively.

In this paper, the design of the dynamic controller will be presented in Section 5.

4. Adaptive Kinematic Controller with RBFNN

In this section, the method of designing an adaptive kinematic controller using an RBFNN is presented. As mentioned above, the adaptive kinematic controller can adjust its parameters automatically and online during the operation of the DDMR. This adjustment is based on the nonlinear identification of the RBFNN and the adaptation of the kinematic controller parameters.

4.1 Radial Basis Function Neural Network

An RBFNN is a three-layered feed-forward neural network using radial basis functions as activation functions [12, 13]. Its advantage is that it can identify the nonlinear system models,

Figure 3. The proposed control scheme of trajectory tracking using RBFNN.

such as mobile robots, with high accuracy [14]. Thus, we can use an RBFNN to estimate the position of the DDMR in the trajectory control algorithm. Figure 3 depicts the control scheme of trajectory tracking of the DDMR using an RBFNN. As shown in Figure 3, two RBFNNs are supposed. The first, denoted as RBFNN_x, is used to estimate the x coordinate, while the second, denoted as RBFNN_y, is used to estimate the y coordinate of the DDMR in the inertial reference frame. The structure of each RBFNN proposed in Figure 4 has three, five, and one neurons (nodes) in the input, hidden, and output layers, respectively.

The performance of the RBFNN is related to the number of neurons in the hidden layer. When the number of the neurons of RBFNN increases, the accuracy of position identification improves. However, the running time also increases, affecting the control quality of the entire system [15]. In this study, the selected number of neurons in the hidden layer of each RBFNN is 10.

The configuration and training algorithm of RBFNN_y are similar to that of RBFNN_x.

Configuration of RBFNN_x:

In the first layer of RBFNN_x, the input vector is

$$x = [x_1 \ x_2 \ x_3]^T, \quad (13)$$

where $x_1 = Q_x(k)$; $x_2 = Q_y(k)$ and $x_3 = x(k-1)$ are the x coordinates of the DDMR in inertial reference frame at $k-1$

Figure 4. Radial basis function neural networks: (a) RBFNN_x and (b) RBFNN_y.

step.

As shown in Figure 4, the signals Q_x and Q_y sent to the input layer of each RBFNN are generated based on the proposed kinematic functions:

$$Q_x = v_{ref} \cos \theta - \omega_{ref} a \sin \theta = \dot{x}_d + l_x \tanh\left(\frac{k_x}{l_x} e_x\right), \quad (14)$$

$$Q_y = v_{ref} \sin \theta + \omega_{ref} a \cos \theta = \dot{y}_d + l_y \tanh\left(\frac{k_y}{l_y} e_y\right). \quad (15)$$

The above two functions were proposed for the following main reasons: first, they simplify the calculation of the derivatives mentioned in Eq. (28) because they are independent of the angle θ and unrelated in terms of x and y coordinates; second, they also act as control signals from the kinematic controller. Therefore, along with the feedback position signals from the DDMR, they will enable the RBFNNs to identify the position of the DDMR more accurately.

In the hidden layer, the activated function is implemented by a Gaussian function:

$$\phi_j = \exp \left[-\frac{\|x - c_j\|^2}{2\sigma_j^2} \right], \quad j = 1, 2, \dots, 10, \quad (16)$$

where $\sigma_j = [\sigma_{j1} \ \sigma_{j2} \ \sigma_{j3}]^T$ and $c_j = [c_{j1} \ c_{j2} \ c_{j3}]^T$ are the width and center vectors of the network at node j .

Finally, the output of the RBFNN is defined by

$$x_m(k) = \sum_{j=1}^m \phi_j w_j, \quad (17)$$

where $m = 10$ is number of neurons, $x_m(k)$ is the estimated x coordinate of the DDMR in the inertial reference frame at k step; w_j is the connective weight between the node j of the hidden and output layers.

Hence, the cost function of the RBFNNx can be calculated as

$$E_{RBF}(k) = \frac{1}{2} (x(k) - x_m(k))^2. \quad (18)$$

Online Neural Network Training:

To build the neural network model plant, we employed the back-propagation algorithm [13] to train the weights. Therefore, the adjustments of the neural model used to minimize the cost function E_{RBF} are defined as follows:

$$\begin{cases} \Delta w_j(k) = \eta [x(k) - x_m(k)] \phi_j, \\ w_j(k) = w_j(k-1) + \Delta w_j(k), \end{cases} \quad (19)$$

$$\begin{cases} \Delta \sigma_j(k) = \eta [x(k) - x_m(k)] w_j \phi_j \frac{\|x - c_j\|^2}{\sigma_j^3}, \\ \sigma_j(k) = \sigma_j(k-1) + \Delta \sigma_j(k), \end{cases} \quad (20)$$

$$\begin{cases} \Delta c_{ji}(k) = \eta [x(k) - x_m(k)] w_j \frac{x_j - c_{ji}}{\sigma_j^2}, \\ c_j(k) = c_j(k-1) + \Delta c_j(k), \end{cases} \quad (21)$$

where η is the learning rate.

4.2 Adaptation of Kinematic Controller Gains

The proposed control algorithm in Figure 3 adapts the kinematic based controller gains to minimize the following cost function:

$$E = \frac{1}{2} (e_x^2 + e_y^2) = \frac{1}{2} ((x_d - x)^2 + (y_d - y)^2). \quad (22)$$

The kinematic controller gains are represented by the set $\rho = [k_x \ k_y]$. The partial derivative of the cost function with respect to ρ is

$$\frac{\partial E}{\partial \rho} = \left[\frac{\partial E}{\partial k_x} \ \frac{\partial E}{\partial k_y} \right] = e_x \frac{\partial e_x}{\partial \rho} + e_y \frac{\partial e_y}{\partial \rho} = -e^T \frac{\partial q}{\partial \rho}, \quad (23)$$

where $q = \begin{bmatrix} x \\ y \end{bmatrix}$ and $e = \begin{bmatrix} e_x \\ e_y \end{bmatrix}$.

Using the chain rule, the derivative $(\partial q / \partial \rho)$ can be obtained

$$\frac{\partial q}{\partial \rho} = \frac{\partial q}{\partial Q} \frac{\partial Q}{\partial \rho}, \quad (24)$$

where $Q = [Q_x \ Q_y]^T$.

The first derivative in Eq. (24) is defined as

$$\frac{\partial q}{\partial Q} = \begin{bmatrix} \frac{\partial x}{\partial Q_x} & \frac{\partial x}{\partial Q_y} \\ \frac{\partial y}{\partial Q_x} & \frac{\partial y}{\partial Q_y} \end{bmatrix} = \begin{bmatrix} J_x & J_{xy} \\ J_{yx} & J_y \end{bmatrix}, \quad (25)$$

where $J_x = \partial x / \partial Q_x$, $J_{xy} = \partial x / \partial Q_y$, $J_y = \partial y / \partial Q_y$ and $J_{yx} = \partial y / \partial Q_x$ are the Jacobian values that can be defined by RBF neural networks as

$$J_{x(y)} = \frac{\partial x(y)}{\partial Q_{x(y)}} \cong \frac{\partial x_m(y_m)}{\partial Q_{x(y)}} = \sum_{j=1}^m w_j \phi_j \frac{c_{j1} - x_1}{\sigma_j^2}, \quad (26)$$

$$J_{xy(yx)} = \frac{\partial x(y)}{\partial Q_{y(x)}} \cong \frac{\partial x_m(y_m)}{\partial Q_{y(x)}} = \sum_{j=1}^m w_j \phi_j \frac{c_{j2} - x_2}{\sigma_j^2}. \quad (27)$$

The second derivative in Eq. (24) is defined as

$$\begin{aligned} \frac{\partial Q}{\partial \rho} &= \begin{bmatrix} \frac{\partial Q_x}{\partial k_x} & \frac{\partial Q_x}{\partial k_y} \\ \frac{\partial Q_y}{\partial k_x} & \frac{\partial Q_y}{\partial k_y} \end{bmatrix} \\ &= \begin{bmatrix} \left[1 - \tanh^2 \left(\frac{k_x}{l_x} e_x \right) \right] e_x & 0 \\ 0 & \left[1 - \tanh^2 \left(\frac{k_y}{l_y} e_y \right) \right] e_y \end{bmatrix}. \end{aligned} \quad (28)$$

Substituting Eqs. (25) and (28) in (24), and then substituting

the derived equation in Eq. (23), we have

$$\frac{\partial E}{\partial \rho} = - \begin{bmatrix} (e_x J_x + e_y J_{yx}) \left(1 - \tanh^2 \left(\frac{k_x}{l_x} e_x \right) \right) e_x \\ (e_x J_{xy} + e_y J_y) \left(1 - \tanh^2 \left(\frac{k_y}{l_y} e_y \right) \right) e_y \end{bmatrix}^T. \quad (29)$$

Therefore, the kinematic controller gains will change and adapt to render the cost function zero according to the gradient descent algorithm

$$\begin{aligned} k_x(k) &= k_x(k-1) - \alpha_x \frac{\partial E}{\partial k_x} \\ &= k_x(k-1) \\ &\quad + \alpha_x (e_x J_x + e_y J_{yx}) \left(1 - \tanh^2 \left(\frac{k_x}{l_x} e_x \right) \right) e_x, \end{aligned} \quad (30)$$

and

$$\begin{aligned} k_y(k) &= k_y(k-1) - \alpha_y \frac{\partial E}{\partial k_y} \\ &= k_y(k-1) \\ &\quad + \alpha_y (e_x J_{xy} + e_y J_y) \left(1 - \tanh^2 \left(\frac{k_y}{l_y} e_y \right) \right) e_y, \end{aligned} \quad (31)$$

where α_x and α_y are the learning rates.

4.3 Stability Condition

From Eqs. (11) and (12), the closed-loop equation in terms of the velocity errors is

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \end{bmatrix} + \begin{bmatrix} l_x \tanh \left(\frac{k_x}{l_x} e_x \right) \\ l_y \tanh \left(\frac{k_y}{l_y} e_y \right) \end{bmatrix} = \lambda \Delta, \quad (32)$$

where $\dot{x}_e = \dot{x}_d - \dot{x}$, $\dot{y}_e = \dot{y}_d - \dot{y}$, and $\Delta = \begin{bmatrix} v_{ref} - v \\ \omega_{ref} - \omega \end{bmatrix}$.

Now, defining the output error vector $e = [e_x \ e_y]^T$. Eq. (32) can be written as

$$\dot{e} + f(e) = \lambda \Delta \quad (33)$$

with

$$f(e) = \begin{bmatrix} l_x \tanh \left(\frac{k_x}{l_x} e_x \right) \\ l_y \tanh \left(\frac{k_y}{l_y} e_y \right) \end{bmatrix}.$$

In this tracking control algorithm, $\Delta(t)$ is assumed to move

quickly to zero, whereas the tracking control error vector $e(t)$ remains different from zero. Hence, Eq. (33) can be written as

$$\dot{e} = -f(e), \quad (34)$$

which has a unique equilibrium point at the origin.

We choose the Lyapunov candidate function as

$$V = \frac{1}{2} e^T e > 0. \quad (35)$$

Taking the derivative of V , we obtain

$$\dot{V} = e^T \dot{e} = -e_x l_x \tanh \left(\frac{k_x}{l_x} e_x \right) - e_y l_y \tanh \left(\frac{k_y}{l_y} e_y \right). \quad (36)$$

We also have: $k_x > 0$, $k_y > 0$, $l_x > 0$ and $l_y > 0$, hence $\dot{V} < 0$. Thus, it can be concluded that the tracking system has asymptotically stable equilibrium at the origin, which means the tracking control error vector $e(t) \rightarrow 0$ as $t \rightarrow 0$.

During tracking operation, all the parameters of the kinematic controller are always updated. For these parameters to be positive, the following rule for updating is supposed:

$$\begin{cases} \text{if } (k_x(k-1) + \Delta k_x(k)) < 0 \text{ then } \Delta k_x(k) = 0, \\ \text{if } (k_y(k-1) + \Delta k_y(k)) < 0 \text{ then } \Delta k_y(k) = 0. \end{cases} \quad (37)$$

5. Experimental and Discussion

Figure 5 depicts the experimental setup for the trajectory tracking of the DDMR. The reference trajectory is generated by a personal computer and then uploaded to the embedded control board of STM32F4DISCOVERY [16]. The communication between the main and embedded control boards is supported by the Wi-Fi protocol. The control signals, v_{ref} and ω_{ref} , generated by the embedded board will be sent to the main control board, Arduino Mega2560 [17], on the DDMR.

The DDMR with the dynamic controller will implement the trajectory tracking and send kinematic feedback signals including x , y coordinates and orientation angle θ of the DDMR to the embedded control board at every sample time interval of 0.01 seconds.

Thus, to verify the control law proposed in Section 4, we designed the adaptive kinematic controller using MATLAB & Simulink, as shown in Figures 6 and 7, and embedded it afterwards onto the control board STM32F4DISCOVERY.

In this experimental design, the distance, a , is set to 0.2 m.

Figure 5. Block diagram of control system of DDMR.

Figure 6. Simulink block of adaptive kinematic controller.

The parameters of the conventional kinematic controller are set with $k_y = 0.4$, $l_y = 1$, $k_x = 0.4$, and $l_x = 1$, which are also the initial values of the parameters of the adaptive kinematic controller. The constraint max of (k_x, k_y) defined

by experimental testing is 0.8. Finally, 0.002 is the value of the learning rates α_x and α_y for updating the gains of the kinematic controller.

In the experimental scenario, the desired (reference) trajec-

Figure 7. Simulink block of kinematic controller.

Figure 8. Trajectory tracking of DDMR.

Figure 9. Time response of robot's x position.

tory was the eight shape. The comparison of the robot's location on the x-y plane between the conventional and adaptive kinematic controllers is shown in Figure 8. Figures 9 and 10 show the time response of the robot's location on the x and y axes, respectively.

Figures 11 and 12 depict the time response of robot velocities

between the conventional and adaptive kinematic controller.

The response time of the tracking errors between the conventional and adaptive kinematic controllers is shown in Figure 13. It is clear that the adaptive kinematic controller of the DDMR exhibits better performance in trajectory tracking.

During the entire tracking process, the gains k_x and k_y were

Figure 10. Time response of robot's y position

Figure 11. Time response of linear velocity.

Figure 12. Time response of angular velocity.

constantly changing to accommodate the characteristic variation

Figure 13. Tracking errors.

Figure 14. Time response of gains k_x and k_y of the adaptive kinematic controller.

of the reference trajectory, as shown in Figure 14.

6. Conclusion

In this study, an adaptive kinematic control algorithm based on RBFNN has been described and developed successfully. Compared to the conventional kinematic controller, the adaptive neural network kinematic controller has a shorter response time and better performance in terms of trajectory tracking. Moreover, this paper also presents the method to design kinematic controllers using the MATLAB & Simulink with an embedded control board. Similarly, our work also describes a system more suitable for fast testing, turn-up, and troubleshooting.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

References

- [1] N. V. Vui and Y. T. Kim, "An advanced motion planning algorithm for the multi transportation mobile robot," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 19, no. 2, pp. 67-77, 2019. <http://doi.org/10.5391/IJFIS.2019.19.2.67>
- [2] K. T. Do, K. Lee, D. Y. Im, B. Kwak, and Y. J. Ryoo, "Design of fuzzy-PID controller for path tracking of mobile robot with differential drive," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 18, no. 3, pp. 220-228, 2018. <https://doi.org/10.5391/IJFIS.2018.18.3.220>
- [3] D. DeVon and T. Bretl, "Kinematic and dynamic control of a wheeled mobile robot," in *Proceedings of 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, 2007, pp. 4065-4070. <https://doi.org/10.1109/IROS.2007.4399599>
- [4] O. Mohareri, R. Dhaouadi, and A. B. Rad, "Indirect adaptive tracking control of a nonholonomic mobile robot via neural networks," *Neurocomputing*, vol. 88, pp. 54-66, 2012. <https://doi.org/10.1016/j.neucom.2011.06.035>
- [5] F. A. Salem, "Kinematics and dynamic models and control for differential drive mobile robots," *International Journal of Current Engineering and Technology*, vol. 3, no. 2, pp. 253-263, 2013.
- [6] F. N. Martins, W. C. Celeste, R. Carelli, M. Sarcinelli-Filho, and T. F. Bastos-Filho, "An adaptive dynamic controller for autonomous mobile robot trajectory tracking," *Control Engineering Practice*, vol. 16, no. 11, pp. 1354-1363, 2008. <https://doi.org/10.1016/j.conengprac.2008.03.004>
- [7] N. Leena and K. K. Saju, "Modelling and trajectory tracking of wheeled mobile robots," *Procedia Technology*, vol. 24, pp. 538-545, 2016. <https://doi.org/10.1016/j.protcy.2016.05.094>
- [8] N. A. Majid, Z. Mohamed, and M. A. M. Basri, "Velocity control of a unicycle type of mobile robot using optimal PID controller," *Jurnal Teknologi*, vol. 78, no. 7-4, pp. 7-14, 2016.
- [9] F. N. Martins and G. M. Almeida, "Tuning a velocity-based dynamic controller for unicycle mobile robots with genetic algorithm," in *Proceedings of the 7th Argentinian Robotics Conference (VII JAR)*, Olavarria, Argentina, 2012, pp. 262-269.
- [10] F. G. Rossomando, C. Soria, P. Patino, and R. Carelli, "Model reference adaptive control for mobile robots in trajectory tracking using radial basis function neural networks," *Latin American Applied Research*, vol. 41, no. 2, pp. 177-182, 2011.
- [11] E. Kayacan and M. A. Khanesar, *Fuzzy Neural Networks for Real Time Control Applications: Concepts, Modeling and Algorithms for Fast Learning*. Amsterdam: Butterworth-Heinemann, 2015.
- [12] S. Elanayar and Y. C. Shin, "Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 594-603, 1994. <https://doi.org/10.1109/72.298229>
- [13] S. B. Park and S. K. Oh, "Design of compensator-based radial basis function neural network classifier for error compensation," *Journal of Korean Institute of Intelligent Systems*, vol. 29, no. 3, pp. 163-169, 2019. <https://doi.org/10.5391/JKIIIS.2019.29.3.163>
- [14] C. Pislaru and A. Shebani, "Identification of nonlinear systems using radial basis function neural network," *International Journal of Computer, Information, Systems and Control Engineering*, vol. 8, no. 9, pp. 1528-1533, 2014.
- [15] X. Kong, X. Chen, and J. Guan, "PID controller design based on radial basis function neural networks for the steam generator level control," *Cybernetics and Information Technologies*, vol. 16, no. 5, pp. 15-26, 2016. <https://doi.org/10.1515/cait-2016-0048>
- [16] STM32F4DISCOVERY kit, Available <https://www.st.com/en/evaluation-tools/stm32f4discovery.html>
- [17] ARDUINO MEGA 2560 REV3, Available <https://store.arduino.cc/usa/mega-2560-r3>

Tran Quoc Khai received his M.S., and B.S. in Department of Electrical Drives and Machines, Kazan State Power Engineering University, Russian Federation. From 2014 to 2019, he was a lecture at Ho Chi Minh City University of Transport. Since 2019, he has been studying a doctor courses in the Department of Electrical Engineering, Mokpo National University, Korea.

E-mail: khai_dv@hcmutrans.edu.vn

Young-Jae Ryoo received his Ph.D, M.S., and B.S. degrees in the Department of Electrical Engineering, Chonnam National University, South Korea in 1998, 1993, and 1991, respectively. He was a visiting researcher in North Carolina A&T State University, USA in 1999. He was a visiting professor in the Department of Mechanical Engineering, Virginia Tech, US from 2010 to 2012. He is currently a professor in the Department of

Electrical and Control Engineering, Mokpo National University, South Korea from 2000. He also served as a director with the intelligent space laboratory in Mokpo National University, where he is responsible for the research projects in the area of intelligence, robotics and vehicles. He is currently a board member of Korean Institute of Intelligent Systems from 2002, an editor for the *Journal of Korean Institute of Electrical Engineering* from 2010, an editor for the *Journal of Fuzzy Logic and Intelligent Systems* from 2009, and a committee member of the International Symposium on Advanced Intelligent Systems from 2005. He served as a general chair of International Symposium on Advanced Intelligent System in 2014 and 2015. He won the outstanding paper awards, the best presentation awards, and the recognition awards in International Symposiums on Advanced Intelligent Systems. He is the author of over 200 technical publications. His research interests include intelligent space, humanoid robotics, legged robotics, autonomous vehicles, unmanned vehicles, wheeled robotics and biomimetic robotics.

E-mail: yjryoo@mokpo.ac.kr