Trajectory tracking intelligent controller for differential wheel mobile robot

Vo Thu Ha*, Nguyen Thi Thanh, Than Thi Thuong, Bui Huy Hai

Faculty of Electrical Engineering, University of Economics - Technology for Industries, Vietnam.
*Corresponding author: vtha@uneti.edu.vn
Received 18 Jun. 2023; Revised 06 Aug. 2023; Accepted 10 Oct. 2023; Published 25 Oct. 2023.
DOI: https://doi.org/10.54939/1859-1043.j.mst.90.2023.11-21

ABSTRACT

This paper presents the design and development of a Neural Netwwork-Adaptive backstepping (NN-Adaptive backstepping) controller to track the trajectory of a differentially controlled mobile robot. The controller is designed based on the robot dynamics equation and on the basis of the Backstepping controller. The effectiveness of the proposed controller is verified by simulation on different trajectories and compared with the backstepping controller. Model results The simulation shows that the proposed controller achieves quality better than the Backstepping controller in different trajectories. The tracking performance of the NN-Adaptive backstepping controller compared to the Backstepping controller is clearly improved (performance improved by over 52%), which means that the proposed controller trajectory error output will decrease by this percentage.

Keywords: Mobile robot; Neural Netwwork-Adaptive backstepping controller.

1. INTRODUCTION

The control issue of differential-type autonomous mobile robotic systems has received a lot of attention lately. This is mostly because these systems are being used more frequently in both industrial and service sectors. Automated warehouse order delivery robots, office building delivery robots, and deep exploration robots are a few common uses for such systems [1, 2], etc.

The most straightforward and well-liked wheel mobile robot is the differential wheel mobile robot (DWMR). It is a platform featuring two rear coaxial driving wheels in addition to two front wheels. The DWMR represents a nonholonomic system in general [3]. Based on the assumption that the DWMR does not slip while in motion, which is mathematically equivalent to a set of first-order differential constraints that cannot be integrated, the nonholonomic constraints can be visualized directly important in a situation where the DWMR cannot be shifted horizontally, and the velocity in that direction cannot be integrated [4]. According to the Brockett theorem [5], nonholonomic systems cannot be stabilized solely through time-invariant feedback control laws [6]. Therefore, it is challenging to develop an appropriate controller to achieve stability and trajectorial tracing of nonholonomic DWMR [7].

Several controllers have been proposed for nonholonomic DWRB, of which the two main approaches to DWRB control are stabilizing the robot's posture and tracking the robot's motion trajectory. The purpose of postural stabilization is to stabilize the robot about a certain reference point, while the purpose of trajectory tracking is to have the robot follow a specified reference trajectory [8]. Many theoretical and practical contributions have increased interest in mobile robot control monitoring. There are many tracking control methods for mobile robots that have been proposed such as Sliding mode control [9], Backstepping control [10], Adaptive control [11], and Intelligent control, etc. In these methods, the robot can follow the trajectory well with the ability to change the many different types of DWMR payloads, the backstepping control has become one of the most popular and effective methods [12]. However, the disadvantage of this method is that it needs an accurate model and it cannot deal with uncertainties in the model and in the environment (friction, uncertainty noise).

Recently, the study of Neural Networks (NN) based on tracking algorithms for DWMR with system parameter uncertainty has attracted the attention of many researchers. NN can estimate nonlinear functions with arbitrary precision under certain domains [13], and has been widely used to solve control problems of uncertain nonlinear systems [14]. Many attempts have been made to solve the trajectory tracking problem of DWMR with unknown system parameters. In [15], a NN method for the cling problem of DWMR was proposed. The nonlinear approximation capabilities of NNs have been used to improve the control quality of classical dynamic feedback control schemes. In [16], an adaptive NN based on a tracking control algorithm was proposed for the DWMR system with all-state constraints. If the system parameters are selected correctly, the proposed scheme can ensure that the final limit is consistent with all signals in the DWMR system, and that the tracking error converges to a tightly bounded set of 0.

Therefore, in this paper, the author proposes to design a Neural Network-Adaptive backstepping (NN-Adaptive backstepping) controller to track the trajectory for DWMR. The Backstepping controller will make the system follow the trajectory well when changing with different types of loads, and the NN will adjust the parameters for the Backstepping controller in the environment with model error and uncertainty noise.

This paper is presented in five main sections. Sections 1 and 2 introduce target research and kinetic and dynamic models. Section 3 presents a new proposal for this article. That is the design of the neural network controller - adaptive backstepping controller controls for robots. Part 4 is the simulation and experimental results of the controller. The last part is the conclusion.

2. KINEMATICS AND DYNAMICS FOR DIFFERENTIAL WHEEL MOBILE ROBOTS

Consider an autonomous robot of the differential type moving in any predetermined trajectory, assuming the wheel rolls without slip and no side slip, the passive wheel has a negligible impact on the dynamics. In the plane of the moving medium attach a fixed frame of reference $[O, \mathrm{x}, \mathrm{y}, \mathrm{z}]$ as depicted in figure 1. Where: A is the midpoint of two active wheels, C is the coordinates of the center of gravity of the robot, a is the distance between the coordinates of the center of gravity to the wheel axle, Ra is the radius of the active wheel, 2L is the distance between the two wheels, m is the mass of the robot, m_ω is the mass of the wheel and engine, mc is the mass of chassis, I moment of inertia, υ, ω is velocity and angular velocity, q is fund the robot's set direction, $\dot{\varphi}_r, \dot{\varphi}_l$ are angular velocity, θ is the orientation angle.

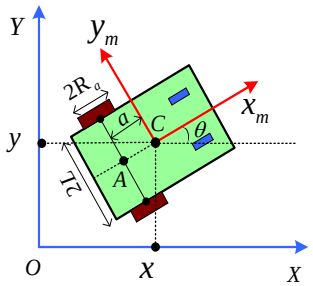

Figure 1. *Kinetic relationship of differential autonomous robot.*

2.1. Forward kinematics model

According to [17], the kinematics model of DWMR is built as follows

The non-holonomic constraint of DWMR is:

$$\dot{x}\cos(\theta) + \dot{y}\sin(\theta) = 0. \tag{1}$$

The non-holonomic constraint can be rewritten to $A(q)\dot{q} = 0$. The Matrix $H(q)$ formed by a set of linear and smooth vector files spanning the empty space of its mean $H^T(q)A(q) = 0$. From there, the kinematics model of the DWMR is:

$$\dot{q} = H(q)\upsilon \tag{2}$$

Therefore: $H(q)=\begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}$ and $\upsilon=\dot{q}=\left[\dot{x},\dot{y},\dot{\theta}\right]^T$ (3)

2.2. Dynamics model

The dynamics model equation of DWMR according to [17] has the following form:

$$M(q)\ddot{q}+C(q,\dot{q})+F(q,\dot{q})+g(q)+\tau_d=B(q)\tau-A^T(q)\lambda \quad (4)$$

Where: $M(q)$ is positive inertia matrix; $V(q,\dot{q})$ is centripetal Matrix; $F(\dot{q})$ is surface friction; $G(q)$ is gravity acceleration matrix; τ_d is noise component; $B(q)$ is input matrix; $A^T(q)$ is binding matrix; λ is Lagrange multiplier vector.

According to [17], with the method of calculating Lagrange dynamics, we have:

$$\begin{bmatrix} m & 0 & ma\sin\theta \\ 0 & m & -ma\cos\theta \\ ma\sin\theta & -ma\cos\theta & I_C+2ma^2 \end{bmatrix}\ddot{q}+\begin{bmatrix} ma\dot{\theta}^2\cos\theta \\ ma\dot{\theta}^2\sin(\theta) \\ 0 \end{bmatrix}=\begin{bmatrix} F_x \\ F_y \\ \tau \end{bmatrix}+\begin{bmatrix} C_x \\ C_y \\ C_\theta \end{bmatrix} \quad (5)$$

Where: F_x is the actuator force in the x-direction, F_y is the actuator force in the y-direction, τ is the ac I tutor rotational torque on the robot, C_x, C_y, C_θ are the constraint forces in the directions.

3. DESIGNING NN - ADAPTIVE BACKSTEPPING CONTROLLER FOR DWMR WITH DIFFERENTIAL TYPE

3.1. The backstepping controller design

From the forward kinematics equation (2), the dynamic equation (4) will be written:

$$M(q)\dot{\upsilon}(t)+C(q,\dot{q})\upsilon(t)+F(\dot{q})+G(q)+\tau_d=B(q)\tau \quad (6)$$

The choice control law equation is:

$$\tau_{dk}=\bar{B}^{-1}(q)[\bar{M}(q)u+\bar{C}(q,\dot{q})\upsilon(t)+\bar{F}(\dot{q})] \quad (7)$$

Where $\bar{M}(q),\bar{C}(q,\dot{q}),\bar{F}(\dot{q})$ are the estimated model parameters of $M(q),C(q,\dot{q})$ va $F(\dot{q})$.

From equations (6) and (7), we have the closed dynamics equation:

$$M(q)\dot{\upsilon}(t)+C(q,\dot{q})\upsilon(t)+F(\dot{q})+G(q)+\bar{\tau}_d=B(q)\left[\bar{B}^{-1}(q)[\bar{M}(q)u+\bar{C}(q,\dot{q})\upsilon(t)+\bar{F}(\dot{q})]\right] \quad (8)$$

Assuming that the model parameters we estimate are correct $\bar{M}(q)=M(q),\bar{C}(q,\dot{q})=C(q,\dot{q}),$ $\bar{F}(\dot{q})=F(\dot{q})$), equation (8) will then be written equivalent to equation (9).

$$(8)\Leftrightarrow\dot{\upsilon}(t)=u \quad (9)$$

Therefore, the control system : $\begin{cases}\dot{q}=S(q)\upsilon(t)\\ \dot{\upsilon}(t)=u\end{cases}$; with system: $\begin{cases}\dot{x}=f(x)+g(x)\xi\\ \dot{\xi}=u\end{cases}$ (10)

The Backstepping controller is designed using a stable feedback control for the system (10). The recommended control inputs to stabilize the system (10) are as follows:

$$u=\dot{\upsilon}_c+K_c(\upsilon_c-\upsilon) \text{ (with } K_c \text{ being a positive definite matrix)} \quad (11)$$

With $\upsilon_c = \begin{bmatrix} \upsilon_r \cos e_\theta + K_x e_x \\ \omega_r + K_y \upsilon_r e_y + K_\theta \upsilon_r \sin e_\theta \end{bmatrix}$ and $e_p = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} = \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}$ (12)

where K_x, K_y, K_θ is a positive definite constant.

From equations (2) and (4) combined with equations (11) and (12), we have the backstepping control structure diagram as follows:

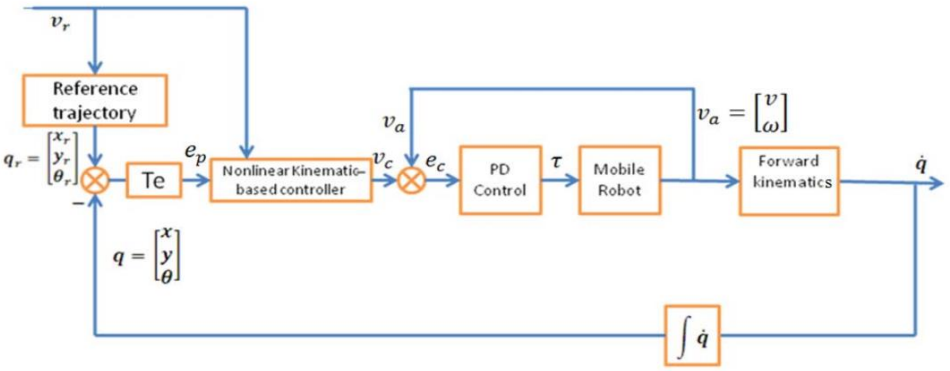

Figure 2. *Structure diagram of the backstepping control system.*

Choose the Lyapunov function: $V = \frac{1}{2}(e_x^2 + e_y^2) + \frac{(1-\cos e_y)}{K_y}$ (13)

$$\Rightarrow \dot{V} = \dot{e}_x e_x + \dot{e}_y e_y + \frac{\dot{e}_\theta \sin e_\theta}{K_y} \quad (14)$$

With: $e_c = \upsilon - \upsilon_c = \begin{bmatrix} e_4 \\ e_5 \end{bmatrix} = \begin{bmatrix} \upsilon \\ \omega \end{bmatrix} - \begin{bmatrix} \upsilon_r \cos e_\theta + K_x e_x \\ \omega_r + K_y \upsilon_r e_y + K_\theta \upsilon_r \sin e_\theta \end{bmatrix} = \begin{bmatrix} \upsilon - \upsilon_r \cos e_\theta - K_x e_x \\ \omega - \omega_r - K_y \upsilon_r e_y - K_\theta \upsilon_r \sin e_\theta \end{bmatrix}$ (15)

From (14) and (15) we have: $\dot{V} = -K_x e_x^2 - \frac{K_\theta \upsilon_r \sin^2 e_\theta}{K_y} \le 0$ (16)

$\Rightarrow$ Stable system.

3.2. Building a NN – Adaptive backstepping controller

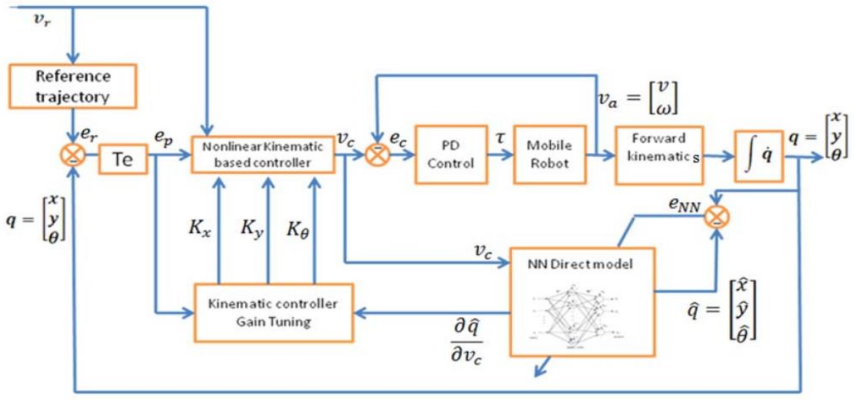

Figure 3. *Motion control structure diagram for a mobile robot with NN-Adaptive backstepping controller.*

As can be seen in equations (11) and (12) above, the controller has three parameters mentioned as positive constant values (K_x, K_y, K_θ). The disadvantage of the Backstepping controller above is that we determine the control law based on precisely determining the parameters of the model, and determining the constants so that the system error approaches K_x, K_y, K_θ zero. The proposed neural network algorithm provides parameters for the backstepping controller that can be flexibly changed according to the robot's motion trajectory. Neural network- Adaptive backstepping controller, is shown in Fig. 3.

The mathematical model of a neuron is depicted in Fig. 4:

The input weights w_j, the firing threshold b (also called the bias), the summation of the weighted inputs, and the nonlinear activation function are shown in the above figure. If the cell inputs are n signals at the time instant k, $x_1(k), x_2(k), x_3(k), ...x_n(k)$ and the output is the scalar $y(k)$ the mathematical equation of the neuron can be written as follows:

$$y(k) = f(\sum_{j=1}^{n} w_j x_j(k) + b) \quad (17)$$

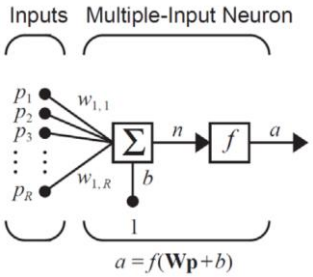

Figure 4. *Mathematical model of a neuron.*

We can write the output vector: $y(t) = \begin{bmatrix} y_0 & y_1 & \cdots & y_n \end{bmatrix} as: y(t) = f(W^T x + b_W)$ (18)

Because neural networks are intricate nonlinear distributed systems, they have many different uses. The neural networks' categorization (for pattern recognition) and function approximation are two of their most crucial characteristics. The universal function approximation property of NN comprising at least two layers is a vital component in NN closed-loop control applications. One layer NNs lack the capacity to approximate any function at all (Fig. 5).

$$f(x) = W^T \sigma(V^T x) + \zeta \quad (19)$$

$\|\zeta\| = \zeta_N$ for all in the compact set *S*, for some sufficiently large value L of hidden layer neurons. The value ϵ is called the NN function approximation error and it decreases as the number of hidden layer neurons L increase. On the other hand, on the compact set S, as S becomes larger, the required L generally increases correspondingly. The neural network acting as a function approximator is shown in Fig. 6:

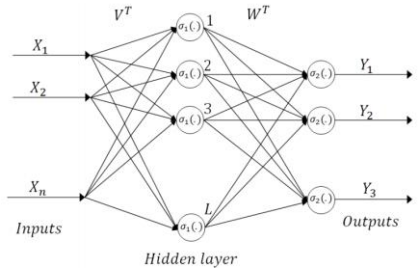

Figure 5. *Two-layer neural network.*

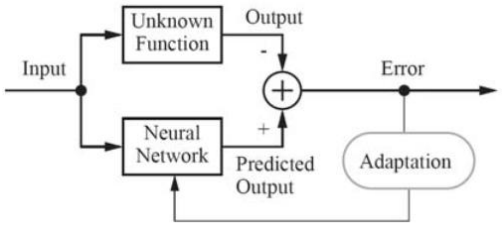

Figure 6. *The function approximation structure for neural networks.*

Today, these problems have, for the most part, been solved and there are very good algorithms for NN weight selection and tuning. The backpropagation training algorithm is one of the famous algorithms because of its simplicity and power. The backpropagation algorithm for a two-layer neural network shown in Fig. 5 using the sigmoid activation function is explained as follows:

$$y_i = \sigma\left(\sum_{l=1}^{L} W_{il}\sigma\left(\sum_{j=1}^{n} V_{lj}x_j + \upsilon_{l0}\right) + w_{i0}\right) \qquad \begin{matrix} i = 1.2...., m \\ l = 1, 2, ..., L \end{matrix} \tag{20}$$

To build NN approximating multi-input and multi-output as for DWMR control, we need a neural network initialization algorithm, then use a back-propagation algorithm to train the network and use the error of NN approximation to show the performance as shown in Fig. 7.

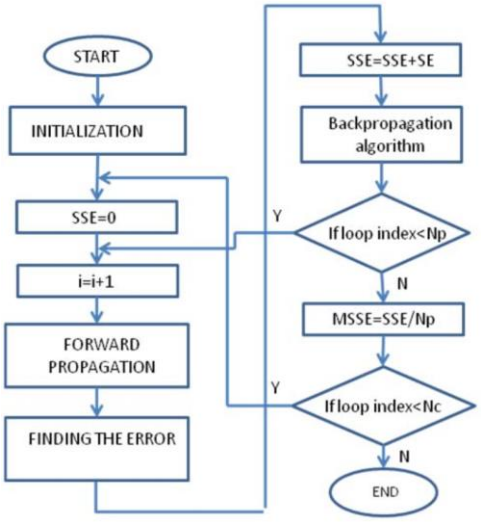

Figure 7. *The general algorithm for implementation of neural network training using backpropagation.*

According to the diagram of the control system structure using NN-Adaptive backstepping controller (Fig. 3). The NN inputs are ($\upsilon = \left[\upsilon_c, \omega_c\right]^T$) υ and ω, the NN outputs are ($\hat{q} = \left[\hat{x}, \hat{y}, \hat{\theta}\right]^T$) $\hat{x}, \hat{y},$ and $\hat{\theta}$. Therefore, we have the NN network structure diagram in Fig. 8:

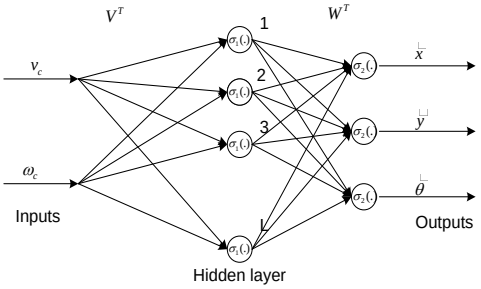

Figure 8. *The NN for DWMR.*

Based on the control structure diagram Fig. 3 and the model deviation equation (7) and equation (12) we have the equation for the PD controller

$$\tau_{dk} = K_p e_c \tag{21}$$

This control structure in Fig. 3 adjusts the parameters K_x, K_y, K_θ of the backstepping controller based on the robot dynamics to ensure the following cost function is minimized:

$$J = \frac{1}{2}\sum \gamma_x {e_x}^2 + \gamma_y {e_y}^2 + \gamma_\theta {e_\theta}^2 \tag{22}$$

With: $\alpha = [K_x, K_y, K_\theta]; \gamma = \begin{bmatrix} \gamma_x & 0 & 0 \\ 0 & \gamma_y & 0 \\ 0 & 0 & \gamma_\theta \end{bmatrix}$ and $e_p = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} = T_e e_r$

$$\Rightarrow \frac{\partial J}{\partial \alpha} = \gamma_x e_x \frac{\partial e_x}{\partial \alpha} + \gamma_y e_y \frac{\partial e_y}{\partial \alpha} + \gamma_\theta e_\theta \frac{\partial e_\theta}{\partial \alpha} = \gamma {e_p}^T \frac{\partial e_p}{\partial \alpha} = -\gamma {e_p}^T T_e \frac{\partial q}{\partial \alpha} \tag{23}$$

From equations (12), (22), and (23) we have:

$$\frac{\partial \upsilon_c}{\partial \alpha} = \begin{bmatrix} \frac{\partial \upsilon_c}{\partial K_x} & \frac{\partial \upsilon_c}{\partial K_y} & \frac{\partial \upsilon_c}{\partial K_\theta} \\ \frac{\partial \omega_c}{\partial K_x} & \frac{\partial \omega_c}{\partial K_y} & \frac{\partial \omega_c}{\partial K_\theta} \end{bmatrix} = \begin{bmatrix} e_x & 0 & 0 \\ 0 & \upsilon_r e_x & \upsilon_r \sin e_\theta \end{bmatrix} \tag{24}$$

From equations (23) and (24) we have:

$$\frac{\partial q}{\partial \alpha} = Jac_v \begin{bmatrix} e_x & 0 & 0 \\ 0 & \upsilon_r e_x & \upsilon_r \sin e_\theta \end{bmatrix} \quad \text{(where } Jac_v \text{ is the Jacobian matrix)} \tag{25}$$

Backstepping controller parameters are updated according to:

$$K_x = K_x + \Delta K_x; K_y = K_y + \Delta K_y; K_\theta = K_\theta + \Delta K_\theta \tag{26}$$

Which: $\Delta K_x = -\eta_{K_x} \frac{\partial J}{\partial K_x}; \Delta K_y = -\eta_{K_y} \frac{\partial J}{\partial K_y}; \Delta K_\theta = -\eta_{K_\theta} \frac{\partial J}{\partial K_\theta}$ ($\eta_{K_x}, \eta_{K_y}, \eta_{K_\theta}$ learning speed).

The Jacobian matrix: $$Jac_\upsilon = \frac{\partial q}{\partial \upsilon_c} = \frac{\partial q}{\partial \upsilon_a} . \frac{\partial \upsilon_a}{\partial \tau} . \frac{\partial \tau}{\partial e_c} . \frac{\partial e_c}{\partial \upsilon_a} \tag{27}$$

According to formula (2), we have: $q = \int \dot{q} dt = \int H \upsilon_a dt$ (28)

Which: $\frac{\partial \upsilon_a}{\partial \tau} = Jac_\tau = \begin{bmatrix} \frac{1}{m} t & 0 \\ 0 & \frac{1}{I_c + ma^2} t \end{bmatrix}; \frac{\partial \tau}{\partial e_c} = \begin{bmatrix} K_{p_R} & 0 \\ 0 & K_{p_L} \end{bmatrix}; \frac{\partial e_c}{\partial \upsilon_a} = \frac{\partial (\upsilon_c - \upsilon_a)}{\partial \upsilon_c} = 1$ (29)

According to (21) $\tau = K_p e_c$ and $K_p = \begin{bmatrix} K_{P_R} & 0 \\ 0 & K_{P_L} \end{bmatrix}$ is a positive symmetric matrix. (30)

From equations (27), (28), (29) and (30) we have: $Jac_v = \int H . Jac_\tau . \begin{bmatrix} K_{P_R} & 0 \\ 0 & K_{P_L} \end{bmatrix} dt$ (31)

The output of the Neural network will be calculated (Use the 2-layer NN like Fig. 8):

$$y_i = \sigma \left(\sum_{l=1}^{L} W_{il} \sigma \left(\sum_{j=1}^{n} V_{lj} x_j + \upsilon_{l0} \right) + w_{i0} \right) \tag{32}$$

$$\Leftrightarrow y_i = \sigma \left(\sum_{l=1}^{L} W_{il} Z_l \right) \text{(i=1,2,3…m; l=1,2,3,…L)(47), and } Z_l = \sigma \left(\sum_{j=1}^{n} W_{lj} Z_j \right) \tag{33}$$

Using the Jacobian matrix as:

$$\frac{\partial y_i}{\partial x_i} = \frac{\partial y_i}{\partial Z_l} . \frac{\partial Z_l}{\partial x_i} = W_{il} . \dot{\sigma} \left(\sum_{l=1}^{L} W_{il} Z_l \right) . V_{lj} . \dot{\sigma} \left(\sum_{j=1}^{n} V_{lj} x_i \right) \tag{34}$$

$\Rightarrow$ Equation (34) is the equation used to calculate the Jacobian matrix-like equation (31). The neural network in this method performs all the exact calculations that approximate the controller's parameters.

4. SIMULATION RESULTS USING THE NN - ADAPTIVE BACKSTEPPING CONTROLLER FOR DWMR

The parameters for a neural network of the system:

$N_h = 20$: Number of neurons in the neural network; $\eta_x = \eta_y = \eta_\theta = 0.2$: Speed coefficient.

$\beta_x = \beta_y = \beta_\theta = 0.3$: The coefficient used for the back-propagation algorithm; $N_c = 100$: Repeat cycle; $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$: Activation function for the hidden layer of the training network; $f_1(x) = x$: Activation function for the output layer; Kinetic model parameters: m = 10 kg; J = 0.56 kgm^2.

Based on the controller structure diagram in Fig. 3, we have a system control simulation diagram on Matlab/Simulink.

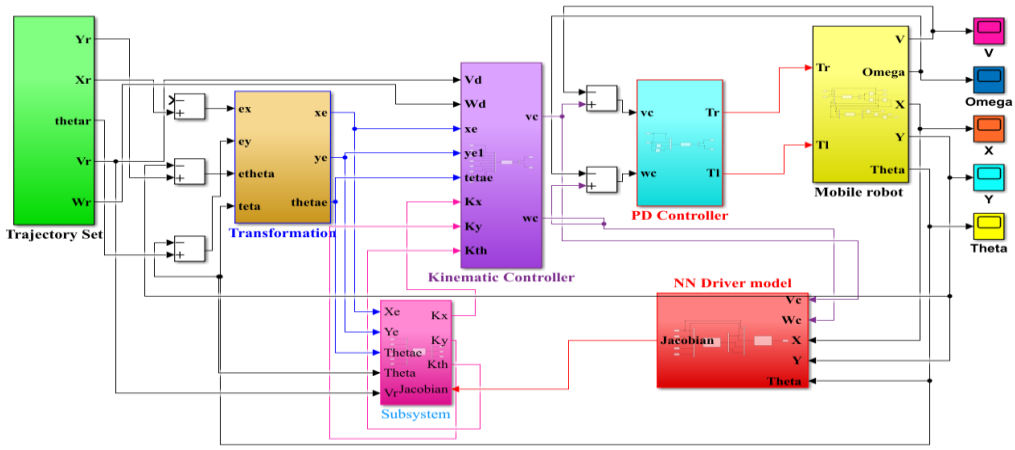

Figure 9. *Block diagram of control system simulation on Matlab.*

Case 1: Simulation for a mobile robot moving in a linear trajectory ($y=x$) from the starting point with coordinates (0,10,0) with parameters of the controllers as follows:

$K_x = 1, K_y = 55, K_\theta = 15$ for Backstepping controller. The parameters for the NN-Adaptive backstepping controller: $J = \frac{1}{2}\sum \gamma_x e_x^{\ 2} + \gamma_y e_y^{\ 2} + \gamma_\theta e_\theta^{\ 2}, \gamma_x = 1, \gamma_y = 50, \gamma_\theta = 1$

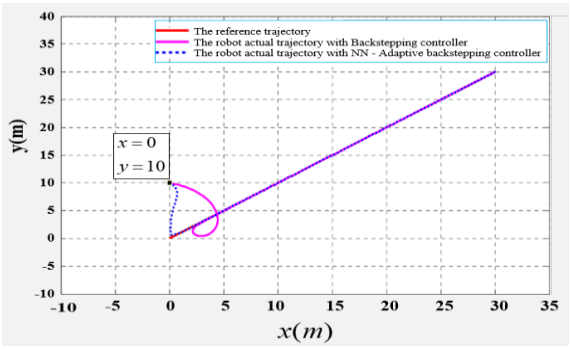

Figure 10. *Motion in a linear linear trajectory.*

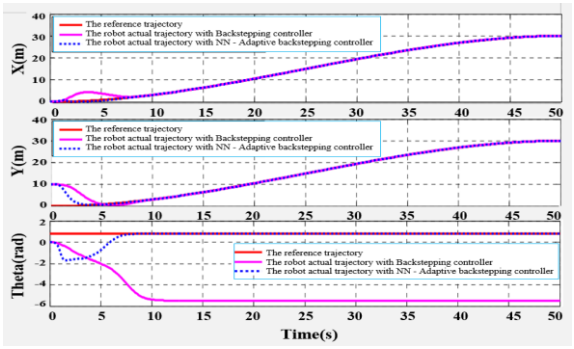

Figure 11. *Motion in x, y, and θ (linear trajectory).*

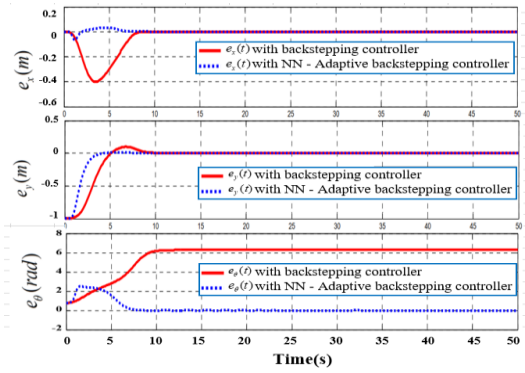

Figure 12. *Deviations in x, y and when moving along a straight-line trajectory.*

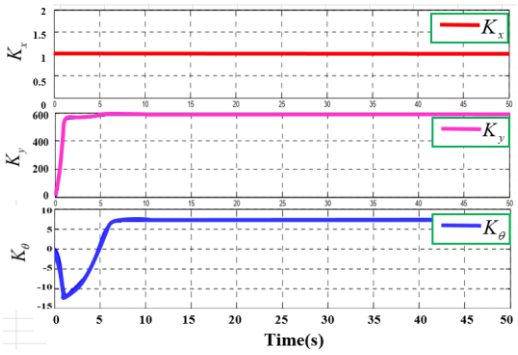

Figure 13. *NN-Adaptive backstepping controller parameters.*

In case 1, the trajectory is a diagonal line, the system error to zero in the transient time for the Backstepping controller is 10 s, and for NN-Adaptive backstepping controller only 1.8 s. The set of parameters that NN calculates at a fixed value when the system is stable is $K_X = 1; K_y = 600; K_\theta = 7$. The special thing in this case is that the Backstepping controller does not bring the theta bias to zero, while the NN-Adaptive backstepping controller completely responds to zero within 6 s.

Case 2: Simulation for a mobile robot moving in a square reference trajectory from the starting point with coordinates (10,0,0) with the parameters of the controllers as follows:

$K_x = 1, K_y = 65, K_\theta = 15$ for Backstepping controller. The parameters for the NN-Adaptive backstepping controller: $J = \frac{1}{2}\sum \gamma_x e_x^{\ 2} + \gamma_y e_y^{\ 2} + \gamma_\theta e_\theta^{\ 2}, \gamma_x = 1, \gamma_y = 50, \gamma_\theta = 1$.

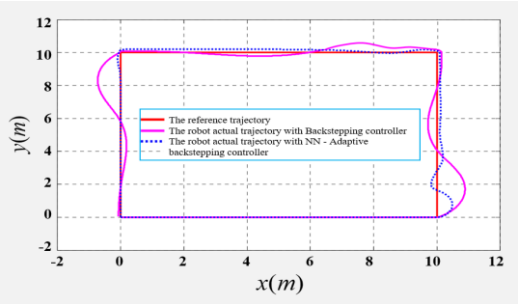

Figure 14. *Response to the motion simulation in a rectangular trajectory.*

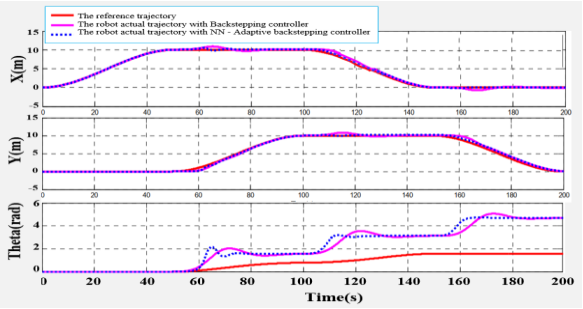

Figure 15. *Motion in x, y, and θ.*

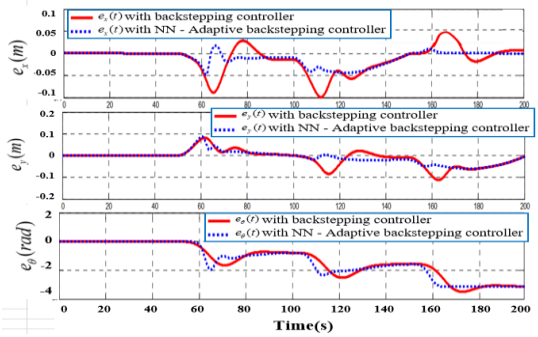

Figure 16. *Response to simulated deviation when moving along a rectangular trajectory.*

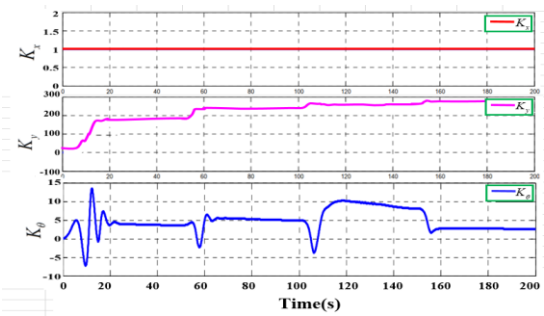

Figure 17. *NN-Adaptive backstepping controller parameter response.*

In case 2 of a rectangular trajectory, the systematic error (Fig.16) will vary with the trajectory and fluctuate between 0.0216 m and 0.0976 m, but for the NN-Adaptive backstepping controller the deviation is significantly smaller (like table 1 below). The set of parameters that NN calculates will change according to the deviation as shown in Fig.17 above.

The deviation of the robot's motion trajectory is calculated by the formula:

$$e_{tr} = \sqrt{e_x^{\ 2} + e_y^{\ 2}} = \sqrt{(x - x_r)^2 + (y - y_r)^2} \quad (35)$$

From the formula for calculating trajectorial deviation (35) as well as from the above simulation results, we have a summary table of deviation parameters and the performance of the proposed controller as follows (table 1):

Table 1. *Trajectory error and tracking response performance for 2 controllers.*

Motion trajectory type	Error backstepping controller (m)	Error NN-Adaptive backstepping controller (m)	The percentage improves the error between the two controllers
Linear line	0.0547	0.0216	60.51%
Square line	0.0976	0.0468	52.04%

5. CONCLUSIONS

In order to overcome the need to accurately determine the model's parameters, the robot's uncertainty in motion control for differential autonomous robots as well as the nonlinearity of the control system, we propose to control the system by Backstepping combined with adjusting controller parameters by an artificial neural network. The feasibility of the controller is suggested through simulation results on Matlab. The performance of the proposed controller has been verified through the Lyapunov theory. Furthermore, tracking simulations of 2 different types of trajectories were performed to verify the performance of the proposed algorithm. Based on the analysis and comparison of data obtained through simulation results on 2 different types of trajectories (linear line trajectories, rectangular trajectories). The proposed NN-Adaptive backstepping controller inherits the advantages of the Backstepping controller, controls the nonlinear system stably, and improves the disadvantages of uncertainty about model parameters as well as external uncertainty noise. Furthermore, the trajectorial tracking test results also confirm its trajectorial tracking performance and durability. In addition, the noise test results confirmed that the proposed controller leads to more stable performance than the Backstepping controller by over 52% for rectangular trajectories and more than 60% for straight-line trajectories. It is expected to apply and develop the controller in practice to improve the cost and performance mode in the construction of service robots in restaurants, robots in warehouses, production workshops, etc.

REFERENCES

[1]. J. Brozak, *"Using Underwater Robotics for Autonomous Deep-Sea Exploration"*, Mobility engineering Produced by SAE Media Group, (2022).

[2]. H. Yang, X. Fan, Y. Xia, and C. Hua, *"Robust tracking control for wheeled mobile robot based on extended state observer"*, Advanced Robotics, vol. 30, no. 1, p. 1–11, (2015).

[3]. R. Jagielski, *"Autonomous mobile robots in intralogistics. Trends and possible applications"*, AMR Technology industry 4.0 warehouse automation, (2020).

[4]. W. Sun et al., *"Two time-scale tracking control of nonholonomic wheeled mobile robots"*, IEEE Transactions on Control Systems, vol. Vol. 24, no. 6, p. 2059–2069, (2016).

[5]. R. W. Brocket, *"Asymptotic stability and feedback stabilization"*, Differential Geometric Control Theory, vol. 27, pp. 181-191, (1983).

[6]. A. M. Bloch et al., *"Control and stabilization of nonholonomic dynamic systems"*, IEEE Transactions on Automatic Control, vol. 37, no. 11, pp. 1746-1757, (1992).

[7]. Dongliang Wang et al., *"A Robust Model Predictive Control Strategy for Trajectory Tracking of Omni-directional Mobile Robots"*, Journal of Intelligent & Robotic Systems, vol. 98, pp. 439-453, (2020).

[8]. Erkan Kayacan et al., *"Robust Trajectory Tracking Error Model-Based Predictive Control for Unmanned Ground Vehicles"*, preprint version: IEEE/ASME transactions on mechatronics, vol. 21, no. 2, (2021).

[9]. Tri Duc Tran et al., *"Parameter-Adaptive Event-Triggered Sliding Mode Control for a Mobile Robot"*, Robotics, vol. 78, no. 11, (2022).

[10]. Seyed Mohammad Ahmadi et al., *"A state augmented adaptive backstepping control of wheeled*

mobile robots", Sage Journals Home, vol. 43, no. 2, (2020).

[11]. *Mingyue Cui et al., "Adaptive Control for Simultaneous Tracking and Stabilization of Wheeled Mobile Robot with Uncertainties",* Journal of Intelligent and Robotic Systems, vol. 108, no. 3, (2023).

[12]. En Lu et al., *"Adaptive Backstepping Control for a Unicycle-Type Mobile Robot,"* International Journal of Agricultural and Biological Engineering, vol. 13, no. 4, pp. 178-187, (2020).

[13]. Chen et al., *"Observer-based adaptive backstepping consensus tracking control for high-order nonlinear semistrict-feedback multiagent systems",* IEEE Transactions on Cybernetics, vol. 46, no. 7, pp. 1591-1601, (2017).

[14]. J. Chen, H. Qiao, *"Muscle-synergies-based neuromuscular control for motion learning and generalization of a musculoskeletal system,"* IEEE Transactions on Systems, Man, and Cybernetics, vol. 51, no. 6, pp. 3993 - 4006, (2021).

[15]. J. Enrique Sierra-García et al., *"Development and Experimental Validation of Control Algorithm for Person-Following Autonomous Robots",* Electronics, vol. 19, no. 2, (2023).

[16]. Changshun Wang et al., *"Neural Network Based Adaptive Dynamic Surface Control for Omnidirectional Mobile Robots Tracking Control with Full-state Constraints and Input Saturation",* International Journal of Control, Automation and Systems, vol. 19, no. 2, pp. 4067-4077, (2021).

[17]. J. Kacprzyk, *"Wheeled mobile robot control",* Springer Nature Switzerland AG, vol. 380, pp. 2198-4182, (2022).

TÓM TẮT

Bộ điều khiển thông minh bám quỹ đạo chính xác robot di động

Bài báo này trình bày việc thiết kế và phát triển bộ điều khiển bước lùi thích ứng mạng nơ-ron (NN-Adaptive backstepping) để theo dõi quỹ đạo của một robot di động được điều khiển khác biệt. Bộ điều khiển được thiết kế dựa trên phương trình động lực học robot và trên cơ sở bộ điều khiển Backstepping. Hiệu quả của bộ điều khiển đề xuất được xác minh bằng mô phỏng trên các quỹ đạo khác nhau và được so sánh với bộ điều khiển bước lùi. Kết quả mô hình Mô phỏng cho thấy bộ điều khiển đề xuất đạt chất lượng tốt hơn so với bộ điều khiển Backstepping ở các quỹ đạo khác nhau. Hiệu suất theo dõi của bộ điều khiển bước lùi thích ứng NN so với bộ điều khiển bước lùi được cải thiện rõ ràng (hiệu suất được cải thiện hơn 52%), điều đó có nghĩa là đầu ra lỗi quỹ đạo của bộ điều khiển được đề xuất sẽ giảm theo tỷ lệ phần trăm này.

Từ khoá: Robot di động; Bộ điều khiển thích nghi Backstepping kết hợp với mạng nơron.