Neural network trajectory tracking of two-wheeled mobile robot

2015SC004 Madoka ASAI
Supervisor : Gan CHEN

1 Introduction

A mobile robot is more and more widely used in industry, especially unmanned ground vehicles play role in many fields because of their advantages being able to use it for many applications where it may be inconvenient, or impossible to carry a human operator. The basic task in controlling these kind of mobile robots is trajectory tracking. What makes the tracking control problem a difficult one is its nonlinear characteristic. In the area of trajectory tracking study, the PID control algorithm has been used in order to control a mobile robot. However, because of the nonholonomic structure of a mobile robot and the complicated change of lane curvature, the PID gains cannot be tuned easily.

In this paper, the two-wheeled mobile robot tracks a given trajectory using a neural network controller itself without using the PID controller. The mobile robot has only three binary sensor to get its position. We use the structure of the novel PID-like neural network controller (PIDNNC) [1] to design the neural network controller. In the output layer of the neural network controller, we propose applying sigmoid function as an activation function, which is not usually used in the output layer but in hidden layer, in order to consider input saturation of motors. We also propose the neural network controller with using two kinds of derivatives, and the cost function in consideration of suppressing oscillating motion of the the robot.

2 Modeling

In this research, a two-wheeled mobile robot is used as a plant. The schematic diagram of the two-wheeled mobile robot is shown in Figure 1. For this model, we have the following dynamical model described by [2]. The variables and physical parameters are shown in Table 1. The equation of the mobile robot is given as follows;

Table 1 Physical parameters

mass of the body	m	[kg]
input voltage to each motor	V_r, V_l	[V]
moment of inertia of each wheel	I_w	[kgm^2]
moment of inertia of the body	I_c	[kgm^2]
resistance of motor	R_m	[Ω]
motor torque	τ_r, τ_l	[Nm/A]
Counter electromotive force of motor	K_b	[Vs/rad]
motor torque constant	K_t	[Nm/A]
wheel radius	r	[m]
angle of each wheels	ϕ_r, ϕ_l	[rad]
heading angle of the body	θ	[rad]
center position of the body	(x_m, y_m)	[m]
distance between wheel center and center of the body	d	[m]
gear ratio from motor to wheel	n	[−]

$$\begin{bmatrix} \frac{mr^2}{4}+\frac{I_c r^2}{4d^2}+I_w & \frac{mr^2}{4}-\frac{I_c r^2}{4d^2} \\ \frac{mr^2}{4}-\frac{I_c r^2}{4d^2} & \frac{mr^2}{4}+\frac{I_c r^2}{4d^2}+I_w \end{bmatrix}\begin{bmatrix} \ddot{\phi}_r \\ \ddot{\phi}_l \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{1}$$

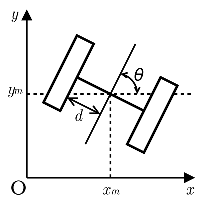

Figure 1 Two-wheeled mobile robot

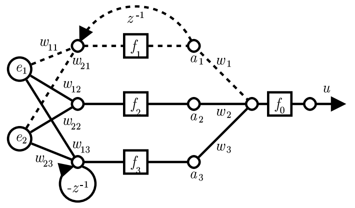

Figure 2 Structure of neural network

$$\begin{bmatrix} \dot{x}_m \\ \dot{y}_m \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos\theta & \frac{r}{2}\cos\theta \\ \frac{r}{2}\sin\theta & \frac{r}{2}\sin\theta \\ \frac{r}{2d} & -\frac{r}{2d} \end{bmatrix}\begin{bmatrix} \dot{\phi}_r \\ \dot{\phi}_l \end{bmatrix} \tag{2}$$

The relation between torque of the motor/wheel and input to the motor/wheel can be written as;

$$\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} = \begin{bmatrix} -\frac{n^2K_tK_b}{R_m} & 0 \\ 0 & -\frac{n^2K_tK_b}{R_m} \end{bmatrix}\begin{bmatrix} \dot{\phi}_r \\ \dot{\phi}_l \end{bmatrix} + \begin{bmatrix} -\frac{nK_t}{R_m} & 0 \\ 0 & -\frac{nK_t}{R_m} \end{bmatrix}\begin{bmatrix} V_r \\ V_l \end{bmatrix} \tag{3}$$

The two-wheeled mobile robot has only three binary sensors which return "0" when the trajectory is detected or "1" when the background is detected. We introduce the error $e_1(k)$ shown in Table 2 as the distance between the position of the robot and the trajectory.

Table 2 Definition of the error $e_1(k)$

$e_1(k)$	$y(k)$
+3 (positive large)	$[1,1,1]$ & $e_1(k-1)>0$
+2 (positive middle)	$[0,1,1]$
+1 (positive small)	$[0,0,1]$
0 (on the trajectory)	$[1,0,1]$, $[0,0,0]$
−1 (negative small)	$[1,0,0]$
−2 (negative middle)	$[1,1,0]$
−3 (positive large)	$[1,1,1]$ & $e_1(k-1)<0$

Even though the distance/error $e_1(k)$ is not accurate, we consider the following direction $e_2(k)$ as a sort of backward derivative.

$$e_2(k) = \begin{bmatrix} \frac{25}{12} & -\frac{48}{12} & \frac{36}{12} & -\frac{16}{12} & \frac{3}{12} \end{bmatrix}\begin{bmatrix} e_1(k) \\ e_1(k-1) \\ e_1(k-2) \\ e_1(k-3) \\ e_1(k-4) \end{bmatrix} \tag{4}$$

3 Structure of neural network controller

We consider the following PD-like neural network controller f_{PDNN} for the trajectory tracking problem;

$$u(k) = f_{PDNN}(e_1(k), e_2(k)) \tag{5}$$

where, $u(k)$ is steering input for the mobile robot. We use the structure of PIDNNC [1] for the PD-like neural network controller. The structure of the PD-like neural network controller is shown in Figure 2. The functions $f_1(\cdot)$, $f_2(\cdot)$, and $f_3(\cdot)$ are activation functions in the hidden layer, which are represented by $f_1(x) = f_2(x) = f_3(x) = x$ in this research. The weights of output layer are $w_j = 1$ $(j = 1, 2, 3)$. The function $f_o(\cdot)$ is the activation function in the output layer given by

$$f_o(x) = \frac{K_{pwm}}{1 + exp(-\frac{4}{K_{pwm}}x)} - \frac{1}{2}K_{pwm} \tag{6}$$

where K_{pwm} is coefficient which determines upper input limit $u_{max}$and lower input limit u_{min} to motors, i. e. $K_{pwm} = u_{max} - u_{min}$.

4 Updating rules for weights

We consider the following cost function $J(k)$ to update weights of the neural network;

$$J(k) = J_1(k) + J_2(k) + \gamma_1 J_3(k) \tag{7}$$

$$J_1(k) = \frac{1}{2}e_1(k)^2, \ J_2(k) = \frac{1}{2}e_2(k)^2 \tag{8}$$

$$J_3(k) = \frac{1}{2}\frac{u(k-1)^2}{\gamma_2(e_{1Max}^2 - e_1(k)^2) + 1}, \ e_{1Max} = 3 \tag{9}$$

where γ_1 and $\gamma_2 \gg 1$ are positive constants to determine the effect of $J_3(k)$ on $J(k)$. The cost function $J_3(k)$ is introduced to avoid oscillation.

We use the gradient descent and back propagation to derive the rule of updating weights. Let $f_s(\cdot)$ be as follows;

$$f_s(x) = \frac{1}{1 + exp(-\frac{4}{K_{pwm}}x)} \tag{10}$$

The rule of updating weights w_{ij}, $(i = 1, 2)$, $(j = 1, 2, 3)$ in hidden layer is as follows;

$$w_{ij}(k) = w_{ij}(k-1) + \Delta w_{ij}(k-1) \tag{11}$$

$$\Delta w_{ij}(k-1) = -\eta_{ij}\frac{\partial J(k)}{\partial w_{ij}(k-1)} \tag{12}$$

$$\begin{aligned}&= -\eta_{ij}\{\sum_{i=1}^{2}\left(e_i(k)sgn\left[\frac{e_i(k) - e_i(k-1)}{u(k-1) - u(k-2)}\right]\right)\\ &+ \gamma_1(\frac{\gamma_2 e_1(k)u(k-1)^2}{\{\gamma_2(9 - e_1(k)^2) + 1\}^2}\\ &\cdot sgn\left[\frac{e_1(k) - e_1(k-1)}{u(k-1) - u(k-2)}\right] + \frac{u(k-1)}{\gamma_2(9 - e_1(k)^2) + 1})\}\\ &\cdot 4 \cdot f_s(a_o(k-1))\{1 - f_s(a_o(k-1))\}e_i(k-1)\end{aligned} \tag{13}$$

where, constants η_{ij}'s are the learning coefficients.

5 Simulation

We present the effectiveness of two different controllers in simulations to analyze contribution of the controllers; PDNN(e_1, e_2), PDNN(e_1), which are PD-like neural network controller using e_1 and e_2, and PD-like neural network controller using only e_1, respectively. We also present the effectiveness of $J_3(k)$ which is the term of the cost function $J(k)$.

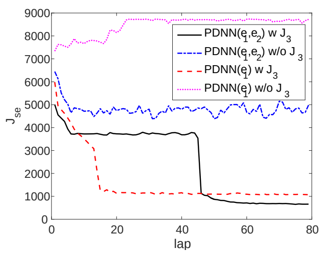

Figure 3 Sum of squared error for each lap

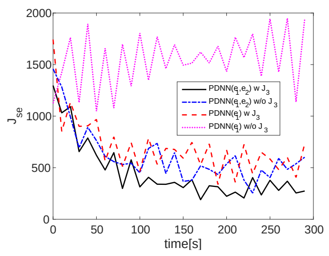

Figure 4 Sum of squared error for every 10[s]

To consider the control performance, we use the evaluation function which calculate the sum of squared error defined as $J_{se} = \sum e_1(k)^2$. Figure 3 shows the control performance evaluated by J_{se} for each lap. Black solid line, blue dash-and-dotted line, red dashed line, and magenta dotted line indicate J_{se} for each lap via PDNN(e_1, e_2) with or without $J_3(k)$ and PDNN(e_1) with or without $J_3(k)$ respectively. PDNN(e_1, e_2) shows better performance than the others that uses only one derivative, and PDNN with $J_3(k)$ shows better performance than the others without $J_3(k)$. It seems that two redundant derivatives and $J_3(k)$ improve the control performance.

6 Experiment

Figure 4 shows the control performance evaluated by J_{se} for every 10[s]. Black solid line, blue dash-and-dotted line, red dashed line, and magenta dotted line indicate J_{se} for each 10[s] via PDNN(e_1, e_2) with or without $J_3(k)$ and PDNN(e_1) with or without $J_3(k)$ respectively. PDNN(e_1, e_2) shows better performance than PDNN(e_1), and the controller with $J_3(k)$ shows better performance than the controller without $J_3(k)$. Experimental results also indicate that two redundant derivatives and $J_3(k)$ improve the control performance.

7 Conclusion

In this paper, the effectiveness of the neural network controllers for a two-wheeled mobile robot is presented. We propose 1. applying sigmoid function in output layer, 2. using the redundant information e_2 as the input for the controller, and 3. deriving the learning algorithm from the cost function including $J_3(k)$. The updating rules for weights in hidden layer are given by using the back propagation algorithm applied *sign* function, and weights can be updated on-line. Simulation and experimental results show that the neural network controller which uses the error $e_2(k)$ and the cost function $J_3(k)$ contributes considerably to get better performance.

References

[1] S. Cong, G. Li, and B. Ji, "A NOVEL PID-LIKE NEURAL NETWORK CONTROLLER," Proc. IFAC world congress, Vol. 38, pp. 121-126, 2005.

[2] Y. Yamamoto, and X. Yun, "Coordinating Locomotion and Manipulation of a Mobile Manipulator," University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-92-18, pp.1-13, 1992.