Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots

Xueyuan Wang [a,b], Gexiang Zhang [a,1], Ferrante Neri [c,d,1], Tao Jiang [e], Junbo Zhao [a], Marian Gheorghe [f], Florentin Ipate [g], and Raluca Lefticaru [g,1]
[a] *School of Electrical Engineering, Southwest Jiaotong University, Chengdu, P.R. China 610031*
[b] *School of Information Engineering, Southwest University of Science and Technology, MianYang, P.R. China 621010*
[c] *Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LEI 9BH, England, United Kingdom*
[d] *Department of Mathematical Information Technology, University of Jyväskylä, Agora, P. O. Box 35, Jyväskylä 40014*
[e] *Control Engineering College, Chengdu University of Information Technology, Chengdu, P.R. China 610225*
[f] *Faculty of Engineering and Informatics, University of Bradford, Bradford, West Yorkshire, BD7 1DP, UK*
[g] *Faculty of Mathematics and Computer Science, University of Bucharest, Academiei 14, Bucharest, Romania*

Abstract. This paper proposes a novel trajectory tracking control approach for nonholonomic wheeled mobile robots. In this approach, the integration of feed-forward and feedback controls is presented to design the kinematic controller of wheeled mobile robots, where the control law is constructed on the basis of Lyapunov stability theory, for generating the precisely desired velocity as the input of the dynamic model of wheeled mobile robots; a proportional-integral-derivative based membrane controller is introduced to design the dynamic controller of wheeled mobile robots to make the actual velocity follow the desired velocity command. The proposed approach is defined by using an enzymatic numerical membrane system to integrate two proportional-integral-derivative controllers, where neural networks and experts' knowledge are applied to tune parameters. Extensive experiments conducted on the simulated wheeled mobile robots show the effectiveness of this approach.

Keywords. Membrane computing, membrane controller, PID, trajectory tracking, nonholonomic wheeled mobile robot

1. Introduction

Since the pioneering work of Păun in 1998, an increasing attention has been paid to Membrane Computing (MC) in the last years [32,51,50,54,47,53,13]. In recent years, much attention has been paid to MC applications, especially to real-life applications, in parallel with richly theoretical results [50]. The hybrid methods of MC and meta-heuristics have been developed to solve various problems, such as broadcasting problems [54], constrained engineering optimization problems [47]. Membrane systems are also used for some real-life applications. For example, spiking neural P systems (see also [19,17,18]) are used for fault diagnosis [53]; probabilistic membrane systems are applied to model ecological systems [13]. More generally, spiking neural networks are used in an array of applications, including for example image processing [36] and approximation models [39,44].

Another promising real-life application is the use of Numerical P Systems (NPS) and Enzymatic Numerical P Systems (ENPS), which have been proved to be Turing universal [40], to design robots behavior controllers [8,31], see also [41,52,43,26,34]. The frame-

[1]Corresponding Author: Ferrante Neri, Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LEI 9BH, England, United Kingdom; E-mail: fneri@dmu.ac.uk; Gexiang Zhang, School of Electrical Engineering, Southwest Jiaotong University, Chengdu, China; E-mail:zhgxdylan@126.com

works of NPS and ENPS controllers for mobile robots have been developed and showed that membrane controllers have an excellent performance [8,31]. But those controllers only achieve some relatively simple tasks, such as obstacle avoidance, wall following and following a leader, etc. Due to the numerical nature, the parallel distributed structure and computational power of NPS and ENPS, a membrane controller is a good choice for mobile robots.

This paper focuses on the trajectory tracking [46, 5,12] control of Wheeled Mobile Robots (WMR), which is a nonlinear problem with nonholonomic constraints, see also [25,4]. We will design a membrane controller for solving the complex mobile robot trajectory tracking problems (MR2TP). The control of MR2TP consists of two main models: kinematic and dynamic. The main aim of kinematic controller design is to propose some control rules which can make WMR follow reference trajectory quickly, smoothly and accurately. But the design of dynamic model more concern the potential in dealing with large parametric uncertainties and disturbances while WMR running. The main contributions of this paper are summarized as follows:

In this study, a novel trajectory tracking control approach is proposed to enhance the WMR control performance. The novelty of this approach is twofold: both feed-forward and feedback controls are considered together to design the controller for the WMR kinematic model; a Proportional-Integral-Derivative (PID) based Membrane Controller (PIDMC) is presented for the WMR dynamic model. The kinematic controller is designed based on Lyapunov stability theory to produce the precisely desired velocity, which is used as the input of the WMR dynamic model. The dynamic controller uses an enzymatic numerical P system to integrate two PID controllers by considering the idea of neural networks and experts' knowledge. The effectiveness of the introduced control approach is verified by applying simulated WMR.

The rest of this study is organized as follows. Section 2 describes MR2TP. In Section 3, we detail the proposed controller for solving MR2TP. Section 4 presents experimental results. Conclusions are drawn in Section 5.

2. Mobile Robot Trajectory Tracking Problem

2.1. Problem Statement

Since WMR is a complex system with large delays, highly nonlinear characteristic and nonholonomic constraints, it is very difficult to establish an accurate mathematical model for trajectory tracking control. In this paper, the considered WMR mechanical system shown in Fig.1 consists of two differential driving wheels and a back unpowered universal wheel. The passive wheel does not affect the degree of freedom of the kinematic model, and can work with the nonholonomic constraints as follows:

$$\dot{y} \cdot \cos\theta + \dot{x} \cdot \sin\theta = 0 \tag{1}$$

The posture of WMR in two dimensional coordinates XOY is represented by using the Cartesian coordinate vectors with three degrees of freedom $p = \{x, y, \theta\}^T$; the reference posture of the reference WMR is represented by $p_R = \{x_r, y_r, \theta_r\}^T$, where (x, y) is the centroid of the WMR; the positive direction of θ is anticlockwise, which is used to guide the angle of a robot. The motion posture of WMR is determined by linear velocity v_c and angular velocity ω_c, which can be denoted by vector $V = (v_c, \omega_c)^T$, while the velocity of reference WMR can be denoted by vector $V_R = (v_R, \omega_R)^T$. Note that the two wheels are driven by independent torques from two DC motors, where the radius of two wheels are represented by r, while the tread of WMR is denoted by $2R$. It is assumed that the WMR mass center is located at O_c and mounted with non-deformable wheels. While the kinematic model for WMR can be described by (2) with the nonholomic constraints, where Jacobian matrix J is a transformation matrix.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_c \\ \omega_c \end{bmatrix} = J \cdot V \tag{2}$$

WMR moves from the current posture $p = \{x, y, \theta\}^T$ to the next posture (reference WMR posture) $p_R = \{x_r, y_r, \theta_r\}^T$; the posture error is set to $p_e = \{x_e, y_e, \theta_e\}^T$, where x_e, y_e and θ_e represent the lateral, longitudinal and direction errors in mobile coordinates, respec-

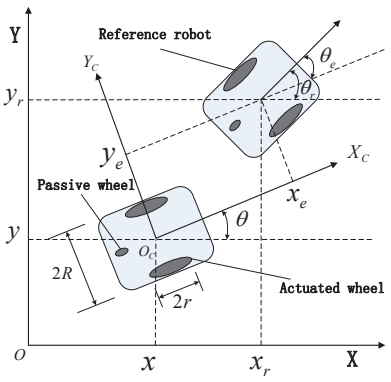


Fig. 1. The mobile robot structure and motion

tively. The posture error transformation between mobile and Cartesian coordinates is shown in (3).

$$\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{3}$$

According to (2) and (3),the differential equations of WMR posture errors can be described as follows:

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} y_e \cdot \omega_c - v_c + v_R \cdot \cos\theta_e \\ -x_e \cdot \omega_c + v_R \cdot \sin\theta_e \\ \omega_R - \omega_c \end{bmatrix} \tag{4}$$

To get $(v_c, \omega_c)^T$, which makes posture error p_e converge to zero as $t \to \infty$, under the effect of $(v_R, \omega_R)^T$, we use the classic speed controller which is based on Lyapunov theory [23]:

$$V_c = \begin{bmatrix} v_c \\ \omega_c \end{bmatrix} = \begin{bmatrix} v_R \cdot \cos\theta_e + k_1 \cdot x_e \\ \omega_R + k_2 \cdot v_R \cdot y_e + k_3 \cdot v_R \cdot \sin\theta_e \end{bmatrix} \tag{5}$$

where k_1,k_2 and k_3 are positive constants and $v_R > 0$. This generic kinematic controller (5) is usually designed by using backstepping method [12,24,11], which will be rewritten by a new kinematic model (6) in our proposed method. The objective of kinematic controller is to generate the input velocities of wheels (v_c, ω_c) for dynamic controller which depicted in dashdot line portion of Fig.2. In fact, the maximal velocity and acceleration constraints of the WMR wheels need to be considered during inner dynamic control. To avoid wheel skid, the accelerations of left and right wheels are set to $a_{\max} = F_{friction}/m_R = \zeta.g$ [27], where $F_{friction}$ is the maximum frictional force which can be provided by the ground, m_R is the mass of the WMR, ζ is the friction coefficient between the wheel and the ground, g is the gravity of earth. In this study, experiments are conducted on Webots [14], the parameter of Coulomb Friction model is set to 1 in Open Dynamics Engine (ODE). So the maximum acceleration is 10 (approximate 9.8). Under this condition, WMR will never slip, and therefore the designed controller can be quite close to the actual working condition.

2.2. Related Work

More and more advanced modern control approaches have been proposed and successfully applied to WMR in industrial contexts, see also [9,37,2,3,29,6]. These control approaches can be classified into the following several categories according to different control theories: back-stepping control [16], Sliding-Mode Control (SMC) [11], adaptive control [24], intelligent control such as neural networks (NN) [48,49] and fuzzy control [12,10]. Despite distinct advantages of the control methods listed above, their inherent disadvantages still exist. For example, SMC is quite simple, insensitive to external interference and has fast system response, but the discontinuous items in control law directly result in poor actual control; fuzzy control has strong robustness and fault tolerance so that the impacts on interference and parameter changes can be greatly reduced, so it is suitable for nonlinear, time-varying and pure delay system control. However, there is also difficulty in designing a fuzzy controller for a complex nonlinear system, due to lack of a general guideline. For example, fuzzy rules and membership functions are decided still based on experts' experience.

MR2TP, regarded as one of the most important problems about motion control of WMR, has drawn considerable attention in the last several decades [12, 23,24,11]. Studies on MR2TP can be divided into two types according to mathematical models: kinematic (controlled by velocity) and dynamic (controlled by

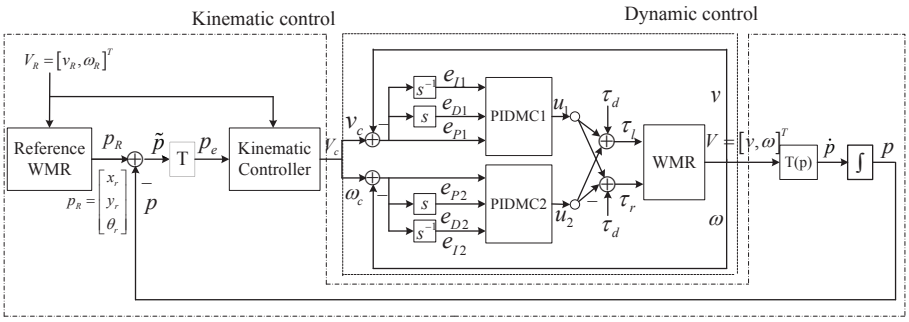


Fig. 2. Kinematic and dynamic controllers for WMR

torque/moment/voltage, etc). Sole dynamic model is only suitable for the relatively special occasions, and consequently there are quite a few studies [48]. Compared with the dynamic model, the kinematic model has more advantages, such as simple structure, easy modeling and analysis, and consistence with the nonholonomic constraints, etc. Thus, many researchers focused on only the kinematic model [23,38,16]. Nevertheless, "excellent velocity tracking" can not be obtained in a real environment, because of the large errors, which are caused by WMR load changes, friction and other instability factors, between the output velocity of kinematic controller and the WMR real velocity.

Unlike one of kinematic and dynamic controllers was emphasized and the other one is simplified in the studies on MR2TP [12,11,10]. In this study, both kinematic and dynamic controllers are thoughtfully considered to obtain the desired velocity generated by kinematic model and to reduce the influence of the external disturbances and system uncertainties of the WMR dynamic model. On one hand, the kinematic model is divided into two parts. SMC and backstepping methods are used to provide more accurate output velocity than classic ones [11,38]. On the other hand, this study also uses NN method to improve fault tolerance and online leaning ability of the nonlinear controller with respect to the dynamic model. Unlike the methods in [48,49] that need to compute all the weights of NN in each cycle, the computation of different branches of NN weights (PI,PD,PID) in this study can be fulfilled inside multiple membranes according to the status of PID experts's knowledge in each cycle. So the computation can be performed in parallel. Thus, the computation is flexible and computing time can be reduced.

3. Design of Kinematic and Dynamic Controllers

This section starts with a brief description of a new approach for kinematic model. Then the detailed description of PIDMC and its implementation for dynamic model are presented. The structure of the proposed kinematic and dynamic controller is depicted in Fig.2.

3.1. Kinematic Controller Design for Wheeled Mobile Robots

The kinematic controller of WMR is enclosed by dash-dot line in Fig.2. A generic control law for the WMR kinematic model is shown in (5), which can be actually divided into two parts: feed-forward part represented as $u_f = (v_f, \omega_f)^T$ and feedback part represented as $u_b = (v_b, \omega_b)^T$. Rewriting (5) as

$$u = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_f + v_b \\ \omega_f + \omega_b \end{bmatrix} \tag{6}$$

and substituting (6) into (4), we can get

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} y_e \cdot (\omega_f + \omega_b) - (v_f + v_b) + v_R \cdot \cos\theta_e \\ -x_e \cdot (\omega_f + \omega_b) + v_R \cdot \sin\theta_e \\ \omega_R - (\omega_f + \omega_b) \end{bmatrix} \tag{7}$$

$$
\begin{aligned}
s(p_e) = \begin{bmatrix} \dot{s}_1(p_e) \\ \dot{s}_2(p_e) \end{bmatrix} &= \begin{bmatrix} -\eta_1 \frac{2(1-e^{-s_1(pe)\cdot\mu_1})}{\mu_1(1+e^{-s_1(pe)\cdot\mu_1})} - \kappa_1 s_1(p_e) \\ -\eta_2 \frac{2(1-e^{-s_2(pe)\cdot\mu_2})}{\mu_2(1+e^{-s_2(pe)\cdot\mu_2})} - \kappa_2 s_1(p_e) \end{bmatrix} = \begin{bmatrix} \dot{x}_e \cdot y_e + \dot{y}_e \cdot x_e \\ \dot{\theta}_e + \frac{\partial\phi}{\partial v_R}\dot{v}_R + \frac{\partial\phi}{\partial y_e}\dot{y}_e \end{bmatrix} \\
&= \begin{bmatrix} (y_e \cdot \omega_f - v_f + v_R \cdot \cos\theta_e) \cdot y_e + (-x_e \cdot \omega_f + v_R \cdot \sin\theta_e) \cdot x_e \\ \omega_R - \omega_f + \frac{\partial\phi}{\partial v_R}\dot{v}_R + \frac{\partial\phi}{\partial y_e}(-x_e \cdot \omega_f + v_R \cdot \sin\theta_e) \end{bmatrix}
\end{aligned} \tag{8}
$$

$$
u_f = \begin{bmatrix} v_f \\ \omega_f \end{bmatrix} = \begin{bmatrix} y_e \cdot \omega_f + v_R \cdot \cos\theta_e + \dfrac{x_e \cdot v_R \cdot \sin\theta_e - x_e^2 \cdot \omega_f + \eta_1 \frac{2(1-e^{-s_1(pe)\cdot\mu_1})}{\mu_1(1+e^{-s_1(pe)\cdot\mu_1})} + \kappa_1 s_1(p_e)}{ye} \\ \dfrac{\omega_R + \frac{\partial\phi}{\partial v_R}\dot{v}_R + \frac{\partial\phi}{\partial y_e}(v_R \cdot \sin\theta_e) + \eta_2 \frac{2(1-e^{-s_2(pe)\cdot\mu_2})}{\mu_2(1+e^{-s_2(pe)\cdot\mu_2})} \kappa_2 s_2(p_e)}{1 + \frac{\partial\phi}{\partial y_e} x_e} \end{bmatrix} \tag{9}
$$

$$
u = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_f + v_b \\ \omega_f + \omega_b \end{bmatrix} = \begin{bmatrix} y_e \cdot \omega_f + v_R \cdot \cos\theta_e + \dfrac{x_e \cdot v_R \cdot \sin\theta_e - x_e^2 \cdot \omega_f + \eta_1 \frac{2(1-e^{-s_1(pe)\cdot\mu_1})}{\mu_1(1+e^{-s_1(pe)\cdot\mu_1})} + \kappa_1 s_1(p_e)}{ye} + k_1 \cdot x_e \\ \dfrac{\omega_R + \frac{\partial\phi}{\partial v_R}\dot{v}_R + \frac{\partial\phi}{\partial y_e}(v_R \cdot \sin\theta_e) + \eta_2 \frac{2(1-e^{-s_2(pe)\cdot\mu_2})}{\mu_2(1+e^{-s_2(pe)\cdot\mu_2})} \kappa_2 s_2(p_e)}{1 + \frac{\partial\phi}{\partial y_e} x_e} + k_2 \cdot v_R \cdot y_e \frac{\sin\theta_e}{\theta_e} + k_3 \cdot \theta_e \end{bmatrix} \tag{10}
$$

3.1.1. Design of feed-forward control

In this study, the design of feed-forward part in the kinematic model is mainly based on SMC method. The improved switching function and approaching law are applied to increase the tracking accuracy.

Lemma 1 [42]: For any $x \in R$ and $|x| < \infty$, $\varphi(x) = x \cdot \sin(\arctan x) \geq 0$, the equality occurs only at $x = 0$.

The backstepping method in [23] is used to design the SMC. According to the *Lemma 1*, the second equation of posture error differential equation in (4) and the partial Lyapunov function $V_y = \frac{1}{2}y_e^2$ [42], we can obtain $\theta_e = -\arctan(v_R y_e)$, which can lead to the convergence of y_e as $x_e = 0$. In other words, y_e will converge to zero as $x_e \to 0$ and $\theta_e \to -\arctan(v_R y_e)$. In this study, a switching function is proposed to make sure $x_e = 0$ and $\theta_e = -\arctan(v_R y_e)$ when the system enters the slide mode. The switching function can be represented as follows:

$$
s(p_e) = \begin{bmatrix} s_1(p_e) \\ s_2(p_e) \end{bmatrix} = \begin{bmatrix} x_e \cdot y_e \\ \theta_e + arctg(v_R \cdot y_e) \end{bmatrix} \tag{11}
$$

According to the theory of SMC, the system will enter the surface of slide mode $s = 0$ in a limited time under varying structure control, as $x_e = 0$ and $\theta_e = -\arctan(v_R y_e)$. Since the asymptotic stability of this sliding mode, the posture error will converge to zero $p_e = \{x_e, y_e, \theta_e\}^T \to 0$ as $t \to 0$. If the system states tend to stay in surface of sliding mode, it must meet the condition $\dot{s}^T(p_e) \cdot s(p_e) < 0$, where $s(p_e)$ is expressed as

$$
\dot{s}_i(p_e) = -\eta_i \frac{2(1 - e^{-s_i(pe)\cdot\mu_i})}{\mu_i(1 + e^{-s_i(pe)\cdot\mu_i})} - \kappa_i s_i(p_e), i = 1, 2 \tag{12}
$$

where η_i and κ_i are positive constants; μ_i is the adjustable parameter, which affects the shape of the sigmoid function and the boundary layer around the surface of switching mode. In order to decrease the jitter effect of SMC, we use sigmoid function $\frac{2(1-e^{-s\cdot\mu})}{\mu(1+e^{-s\cdot\mu})}$ to replace the common sign function $sgn(s)$ in (12).

As $\phi = \arctan(v_R \cdot y_e) \geq 0$, $v_b = 0$ and $\omega = 0$, substituting (12) and (7) into (11) implies formula (8). The

feed-forward control law (9) can be derived from (8), where, $\partial\phi/\partial y_e = v_R \Big/ 1+(v_R \cdot y_e)^2$ and $\partial\phi/\partial v_R = y_e \Big/ 1+(v_R \cdot y_e)^2$.

3.1.2. Design of feedback control

The design of feedback control is mainly based on Lyapunov function [2,3], which is suitable for both of the reference path and real posture of WRM under known conditions. By analyzing (5), (13) can be treated as the feedback part of this trajectory tracking controller.

$$u_b = \begin{bmatrix} v_b \\ \omega_b \end{bmatrix} = \begin{bmatrix} k_1 \cdot x_e \\ v_R(k_2 \cdot y_e + k_3 \cdot \sin\theta_e) \end{bmatrix} \tag{13}$$

In [7], the author proposed a new simple control law based on (5) as the feedback part of the kinematic controller.

$$u_b = \begin{bmatrix} v_b \\ \omega_b \end{bmatrix} = \begin{bmatrix} k_1 \cdot x_e \\ k_2 \cdot v_R \cdot y_e \frac{\sin\theta_e}{\theta_e} + k_3 \cdot \theta_e \end{bmatrix} \tag{14}$$

Combining (9) and (14) together, the new kinematic controller, which contains feed-forward and feedback, can be represented as formula (10).

3.2. Dynamic Controller Design for Wheeled Mobile Robots

The traditional robot motion control usually uses the PID controller [48,35]. In fact, the WMR movement is a nonlinear system with large external disturbances and system uncertainties, so it is extremely difficult to properly tune the parameters of the PID controller. To solve this problem, PIDMC are presented in this section to control the WMR dynamic model.

3.2.1. Dynamic model of wheeled mobile robots

The WMR considered has an n-dimensional configuration space $p \in R^{n\times 1}$ and is subject to m constraints, which can be expressed by using the classical well known Euler Lagrange dynamic equation [24].

$$\begin{aligned} &M(p)\cdot\ddot{p} + V_m(p,\dot{p})\cdot\dot{p} + F_G(p) + G(p) + \tau_d \\ &= B(p)\cdot\tau - J^T(p)\cdot\lambda \end{aligned} \tag{15}$$

$$M(p) = \begin{bmatrix} m_R & 0 & 0 \\ 0 & m_R & 0 \\ 0 & 0 & I \end{bmatrix}, B(p) = \frac{1}{r}\begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ R & -R \end{bmatrix} \tag{16}$$

where $M(p)$ is a positive definite and symmetric inertia matrix and $M(p) \in R^{n\times n}$, I is the moment of inertia of the WMR about the mass center; $\ddot{p}$ and $\dot{p}$ represent acceleration and velocity vectors, respectively; $V_m(p,\dot{p}) \in R^{n\times n}$ is a Coriolis and centripetal matrix; $G(p) \in R^{n\times 1}$ is the gravitational vector; $F_G(\dot{p}) \in R^{n\times 1}$ represents the surface friction; $\tau_d \in R^{n\times 1}$ represents bounded unknown disturbances including unstructured un-modeled dynamics and $\tau = [\tau_r, \tau_l]^T$, where τ_r and τ_l are right and left torque inputs generated by DC motors of the two wheels, respectively; $B(p) \in R^{n\times(n-m)}$ is the input transformation matrix.

As considering the WMR motion only on the horizontal plane, the effect of gravity will be $G(p) = 0$. By regarding the surface friction F_G and disturbance torque τ_d as the disturbance and un-modeled uncertainties items (such as free wheel), and then the more simple and appropriate dynamic model is given as follows [12,11]:

$$\bar{M}(p)\cdot\dot{v} + \bar{\tau}_d = \bar{B}(p)\cdot\tau \tag{17}$$

where, $\bar{M}(p) = J^T M(p) J = \begin{bmatrix} m_R & 0 \\ 0 & I \end{bmatrix} \in R^{2\times 2}$, $\bar{B} = J^T B = \frac{1}{r}\begin{bmatrix} 1 & 1 \\ R & -R \end{bmatrix}$ and $\bar{\tau}_d = J^T \tau_d$.

If the disturbance torque τ_d is regarded as external additional items similar to that in Fig.2, the relationship between the WMR velocity and the torque of DC motor can be derived from (17) as

$$\dot{v}(t) = E\cdot\tau(t) \tag{18}$$

where the transformation matrix E is

$$\begin{aligned} E &= \bar{M}^{-1}(p)\cdot\bar{B}(p) = \begin{bmatrix} m_R & 0 \\ 0 & I \end{bmatrix}^{-1} \cdot \frac{1}{r}\begin{bmatrix} 1 & 1 \\ R & -R \end{bmatrix} \\ &= \frac{1}{m_R \cdot r \cdot I}\begin{bmatrix} I & I \\ Rm_R & -Rm_R \end{bmatrix} \end{aligned} \tag{19}$$

3.2.2. Enzymatic numerical P systems model

NPS has a tree-like membrane structure, in which numerical variables store information, a set of rules responsible for evolving by parallel computing and transmitting information between the nodes (membranes). ENPS is extended from NPS by adding more rules inside a single membrane, which are controlled by a special enzyme-like variable. A standard ENPS is defined as follows [40]:

$$\begin{aligned}\Pi =&(m,H,\mu,(Var_1,E_1,Pr_1,Var_1(0)),\\ &...,(Var_m,E_m,Pr_m,Var_m(0)))\end{aligned} \tag{20}$$

where:

1. m is the number of membranes, $m \geq 1$;
2. H is an alphabet that contains m symbols;
3. μ is a membrane structure;
4. Var_i is the set of variables from membrane i, and $Var_i(0)$ is the initial values for these variables;
5. E_i is a set of enzyme variables from membrane i, $E_i \subset Var_i$;
6. Pr_i is the set of programs (rules) in membrane i, composed of a production function and a repartition protocol;

ENPS represents a distributed and parallel computing model with flexible process control: enzyme variables can be used for conditional transmembrane transport and control the program flow; active rules are performed in parallel inside their membranes and unnecessary rules are not executed; the calculation results are distributed in globally uniform way; thus the computing power of the ENPS is optimized [40], and the membrane structure representation is very efficient for designing robotic behaviors. Both of Java simulator Simp [30] and SNUPS [1] can be used to execute ENPS models.

3.2.3. Proportional-integral-derivative based membrane controller design

PID algorithm has occupied almost 95% of industry control applications due to its simple model and strong robustness [33]. However, the general PID with fixed parameters cannot address the more and more complex nonlinearity control engineering problems, which inspires the research of various kinds of modified PID approaches combined with advanced or intelligent methods in recent years [33,22,20,15]. In this section, we propose the adaptive controller PIDMC with a powerful capability of continuously learning, which benefits from the enzymatic membrane systems property that the enzymes can make it possible to execute more rules inside a single membrane while keeping to find the most suitable PID parameters.

The structure of PIDMC is shown in Fig.2, where the motion of WMR is controlled by linear velocity v_c and angular velocity ω_c. The use of the double PIDMC is to force the linear velocity error $e_1 = v_c - v$ and the angular velocity error $e_2 = \omega_c - \omega$ to converge to zero. Thus, the WMR output from PIDMC is

$$\begin{bmatrix}\tau_l\\ \tau_r\end{bmatrix} = \begin{bmatrix}u_1+u_2\\ u_1-u_2\end{bmatrix} \tag{21}$$

where u_1 and u_2 are the outputs of PIDMC1 and PIDMC2, respectively.

In order to obtain the adaptive PID parameters using a auto-tuning strategy based on the deterministic ENPS, the proposed approach consists of three general steps: (1) we transform the experts' empirical knowledge into a series of rules and allocate them to the corresponding membranes; (2) some rules inspired from the idea of neural networks are endowed with continuous learning capacity to enhance the adaptivity of the PID controller; (3) the rules are executed according to the action of enzymes. Unlike many impractical WMR engineering applications with large time consumption causing bad real-time performance, the proposed approach can overcome the drawback since all the executed rules are performed in a naturally parallel way.

In Fig.2, the input values of the PIDMC controller are the reference velocity $V_c(k)$ and the WMR real velocity $V(k)$, which are preprocessed as the state variables $x_1(k)$, $x_2(k)$, $x_3(k)$ in controller online learning, where $x_1(k)$, $x_2(k)$ and $x_3(k)$ represent error, error change rate and change rate of error change rate, respectively, and are as follows:

$$\begin{aligned}x_1(k) &= V_c(k+1) - V(k+1) = e(k)\\ x_2(k) &= \Delta e(k) = e(k) - e(k-1)\\ x_3(k) &= \Delta e(k) - \Delta e(k-1)\end{aligned} \tag{22}$$

In order to balance various types of errors at each time sample k, the parameters K_I, K_P, K_D is considered as time-varying weighting factors $w_1(k)$, $w_2(k)$, $w_3(k)$, namely

$$u(k) = u(k-1) + K\sum_{i=1}^{3} w_i(k)x_i(k) \tag{23}$$

where K is the scale factor. Unlike the consideration of the quadratic error as the performance index function in robot applications in [5,48], this proposed method introduces the quadratic of the control incremental value $\Delta u(k)$ into performance index function, which can avoid the overshoot phenomenon that $\Delta u(k)$ often faces. Performance index function I is defined as

$$I = \tfrac{1}{2}P \cdot E^2(k) + \tfrac{1}{2}Q \cdot \Delta u^2(k) \tag{24}$$

where the output error $E(k) = V_c(k) - V(k)$, P and Q are the proportional coefficients for output error $E(k)$ and control increment $\Delta u(k)$ in (24), respectively. So, the correction of weighting factors $w_i(k)$ ought to the decreasing direction of performance index I. In other words, it ought to seek along the negative gradient direction of $E(k)$ for $w_i(k)$. By using the partial differential operator on $w_i(k)$, we can get

$$\begin{aligned}\frac{\partial I}{\partial w_i(k)} &= -P \cdot E(k) \cdot \frac{\partial V(k)}{\partial u(k)} \cdot \frac{\partial u(k)}{\partial w_i(k)} \\ &\quad + Q \cdot K \cdot \left[\sum_{i=1}^{3} w_i(k)x_i(k)\right] \cdot x_i(k)\end{aligned} \tag{25}$$

Thus, the adjustment $w_i(k)$ is

$$\Delta w_i(k) = w_i(k+1) - w_i(k) = -\zeta_i \cdot \frac{\partial I}{\partial w_i(k)} \tag{26}$$

where ζ_i is the convergence rate. $\Delta w_i(k)$ can be expanded as the following formula according to (22),(23)and (25):

$$\begin{aligned}\Delta w_i(k) &= \zeta_i \cdot P \cdot K \cdot E(k) \cdot x_i(k) \cdot \frac{\partial V(k)}{\partial u(k)} \\ &\quad - \zeta_i \cdot Q \cdot K \cdot \left[\sum_{j=1}^{3} w_j(k)x_j(k)\right] \cdot x_i(k)\end{aligned} \tag{27}$$

where $i = 1, \ldots, 3$.

In PID control algorithm, as usual $\frac{\partial V(k)}{\partial u(k)}$ can not be directly obtained, instead, it is approximately represented by the sigmoid function $S(x)$. For convenience, we use sign function $sgn(u(k))$ as the approximate part to design PIDMC controller in this study. The inaccurate calculation results can be compensated by the convergence rate ζ_i. Thus, by combining (26) with (27), we can obtain the gain tuning formula as follows

$$\begin{aligned}w_i(k+1) &= w_i(k) + \zeta_{pki} \cdot x_i(k) \cdot x_i(k) \cdot S_{gn}(k) \\ &\quad - \zeta_{qki} \cdot x_i(k) S_{um}(k)\end{aligned} \tag{28}$$

where $\zeta_{pki} = \zeta_i \cdot P \cdot K$, $\zeta_{qki} = \zeta_i \cdot Q \cdot K$, $i = 1, \ldots, 3$, $S_{gn}(k) = \mathrm{sgn}(u(k))$, $S_{um}(k) = \sum_{j=1}^{3} w_j(k)x_j(k)$.

Substituting (28) into (23), we get the output value of PID controller

$$\begin{aligned}\bar{w}_i(k) &= \frac{w_i(k)}{\sum_{i=1}^{3} w_i(k)} \\ u(k) &= u(k-1) + K\sum_{i=1}^{3} \bar{w}_i(k)x_i(k)\end{aligned} \tag{29}$$

In order to further improve the convergence rate of I in(24), the PID controller is decomposed into several sub- controllers, Proportional (P), Proportional-Derivative (PD), and PI in the corresponding zones. The weighting factors $w_1(k)$, $w_2(k)$, $w_3(k)$ are assigned and sub-periods redefined as a combination of the experts' knowledge, which can be generally divided into several cases.

First, define the normalized velocity as

$$N(v) = \frac{v}{v_{\max}} = \begin{cases} 1 & |v| > v_{\max} \\ v/v_{\max} & |v| < v_{\max} \end{cases} \tag{30}$$

where $v_{\max}$ is the maximal velocity of the WMR permitted. In this study, the range of the normalized velocity error is between 0 and 1; the functional zones (error signal of step response) is divided into three parts according to two set-points M_1, M_2, where $0 < M_2 < M_1 < 1$.

(1) If $|N(e(k))| > M_1$, it means that the absolute value of velocity error is too big, so the PID model should be switched into P or PD model;

(2) If $M_1 > |N(e(k))| > M_2$, it means that the error is relatively big, PID model should be applied;

(3) If $M_2 > |N(e(k))|$, it means that the error is small, PI model should be used to eliminate or minimize the static error.

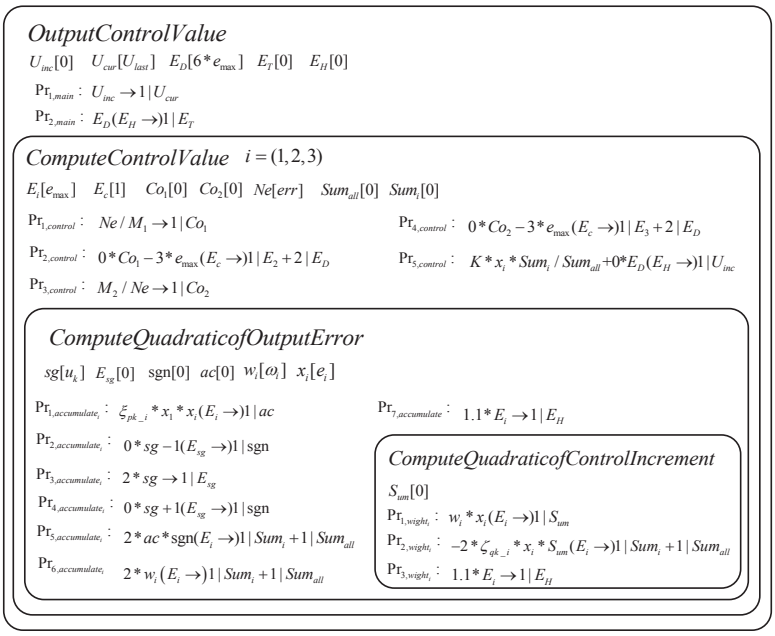


Fig. 3. Enzymatic numerical membrane system for PID controller

Note that the objective of the division of the three functional zones is to improve the controller performance, which includes settling time, rising time and static error, maintained by an appropriate PID controller in each zone.

As shown in Fig.3, the PIDMC controller is designed by using a membrane system with a nested membrane structure containing four membranes. The skin membrane *OutputControlValue* has five variables, where $U_{inc}[0]$ is the control increment with the initial value 0; $U_{cur}[U_{last}]$ is the current position of the controller (the initial value is the control value of the controller at the previous time unit); $E_D[6*e_{\max}]$, $E_T[0]$ and $E_H[0]$ are enzymes, which control the algorithm flow, i.e., E_D has the initial value $6*e_{\max}$, $e_{\max}$ represents a very big value, E_H and E_T have the initial value 0, when E_T does not equal zero, the controller algorithm ends. At the same time, the control value of the controller needs to be updated and the error information are read again followed by the reaction of the controller.

The inner membrane *ComputeControlValue* has seven variables, where enzyme $E_i[e_{\max}]$ is used to judge whether the current weights of parameters K_P, K_I and K_D should be computed or not. Enzyme $E_c[1]$ is used to decide whether the rule $\mathrm{Pr}_{2,control}$ or $\mathrm{Pr}_{4,control}$ should be executed or not according to the variable value of Co_1 or Co_2. $Ne[err]$ is the normalized velocity error (the initial value is the input normalized error of the PID controller at the current time) and is divided by M_1 (rule $\mathrm{Pr}_{1,control}$) or M_2 (rule $\mathrm{Pr}_{3,control}$). Thus, the results are assigned to Co_1 or Co_2. $Sum_i[0]$ and $Sum_{all}[0]$ represent the current weight $w_i(k)$ and the sum of weights $\bar{w}_i(k)$ in (28) and (29), respectively. Rule $\mathrm{Pr}_{5,control}$ should be executed after the two summation results (Sum_i and Sum_{all}) have been obtained, which are decided by the relationship between the sizes of E_D and E_H.

The inner membrane *ComputeQuadraticofOutputError* has seven variables, where $sg[u_k]$ is the input variables of a sign function (the control value u_k at the previous time unit is the initial value). Enzyme $E_{sg}[0]$ is combined with sg in rules ($\mathrm{Pr}_{2,accumulate_i}$ - $\mathrm{Pr}_{4,accumulate_i}$) to compute sign function. $E_{i1}[e_{\max}]$ is an enzyme, which has the initial value $e_{\max}$, $sgn[0]$ is a sign variable. $ac[0]$ is the quadratic error in (24). The result of $ac*sgn$ is assigned to Sum_i and Sum_{all} simultaneously in rule $\mathrm{Pr}_{5,accumulate_i}$. $w_i[\omega_i]$ is a time-

varying weighting variable and its initial value is ω_i, which represent the parameter values K_I, K_P, K_D at the previous time unit, and it is also assigned to Sum_i and Sum_{all} simultaneously in rule $\mathrm{Pr}_{6,accumulate_i}$. $x_i[e_i]$ is the values of the three types errors in (22).

The innermost membrane $ComputeQuadraticofControlIncrement$ has two variable, $S_{um}[0]$, which is the sum of three error weighting variables $S_{um}[k]$ in (28), $E_{i2}[e_{\max}]$ is an enzyme, which has the initial value $e_{\max}$,. When the value E_{i2} is zero, the rule $\mathrm{Pr}_{1,wight_i}$ will be executed, otherwise, the rule $\mathrm{Pr}_{2,wight_i}$ will be performed. For example, if the rule $\mathrm{Pr}_{2,control}$ is executed in the inner membrane $ComputeControlValue$, E_2 maintains zero, thus, the rules $\mathrm{Pr}_{1,wight_2}$ and $\mathrm{Pr}_{2,wight_2}$ will not be executed and consequently the rules $\mathrm{Pr}_{1,accumulate_2}$, $\mathrm{Pr}_{5,accumulate_2}$ and $\mathrm{Pr}_{6,accumulate_2}$ in the inner membrane $ComputeQuadraticofOutputError$ will not also be executed, at the same time, the calculation of the weighting factor $\bar{w}_2(k)$ will not be assigned to accumulate the operations Sum_i and Sum_{all}, which indicates that the controller is in the PD model phase. Thus, this control is helpful to improve the controller performance and to reduce the amount of calculation, which actually benefits from the advantage of parallel computing of the membrane system. In rule $\mathrm{Pr}_{3,wight_i}$, the value of E_{i2} is transferred to E_H, which is used to compare with E_D to judge whether the corresponding rules are performed or not. The factor 1.1 is chosen to make E_H slightly larger than E_D. When both the inner membranes $ComputeQuadraticofOutputError$ and $ComputeQuadraticofControlIncrement$ finish their computations, the enzyme E_H has a bigger value than the variable E_D, thus, the corresponding termination rule will be executed. If E_D has an initial value $6 * e_{\max}$ and the rule $\mathrm{Pr}_{2,control}$ and $\mathrm{Pr}_{4,control}$ are not executed, the controller is in the PID model phase. After the inner membranes $ComputeQuadraticofOutputError$ and $ComputeQuadraticofControlIncrement$ finish their computations, E_H has a value $6.6 * e_{\max}$ larger than E_D, so the rule $\mathrm{Pr}_{2,main}$ will be executed and E_T is not zero. So far, the computation of the membrane system finishes. On the other hand, if the rule $\mathrm{Pr}_{2,control}$ or $\mathrm{Pr}_{4,control}$ is executed, E_D becomes $4 * e_{\max}$, which indicates that the controller is in the PD or PI model phase. After the inner membranes *ComputeQuadraticofOutputError* and *ComputeQuadraticofControlIncrement* finish their computations, E_H has a value $4.4 * e_{\max}$ larger than E_D, thus, the rule $\mathrm{Pr}_{2,main}$ will be executed.

4. Experimental Results

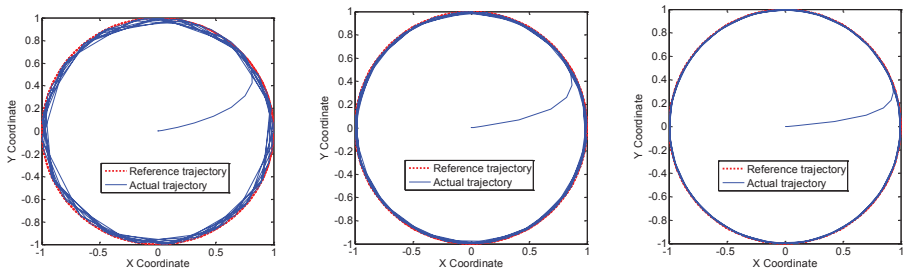
In this section, we first test the performance of the introduced kinematic controller and dynamic controller separately and independently on the software platform MATLAB [28], and then we move to use robots simulation platform Webots to check the presented trajectory tracking control approach. The experiments are conducted on the PC with CPU 2.3GHz, 2GB RAM, and the software platform MATLAB7.4 and Windows XP OS.

4.1. Performance Tests of Kinematic Controller

The experiments consider two kinds of trajectories, circular line and sine, to test the effectiveness of the introduced kinematic control method. The parameters are set as follows: $\eta_1 = \eta_2 = 0.01$, $\kappa_1 = \kappa_2 = 5$, $\mu_1 = \mu_2 = 0.6$ in (9); $k_1 = k_3 = 4$, $k_2 = 6$ in (14). The circular line trajectory is generated from the uniform motion line velocity $v_R(t) = 1$, and the angular velocity $\omega_R(t) = 1$. The desired posture of the reference trajectory is set as a circumference with radius 1 and the center on the origin of the coordinate. The initial posture of the actual WMR is $p_R(0) = [0,0,0]^T$.

To bring into comparison, we design several experiments applying three kinds of kinematic controllers: (1) classic kinematic control law in many publications [24,11,12]; (2) feed-forward kinematic control law (SMC) in (9), which is designed by backstepping method; (3) the introduced kinematic control law in (10), which integrates feed-forward with feedback parts. The results are shown in Fig.4(a)-(c), respectively, where the red dotted circular line is the reference trajectory; the blue solid circular line is the results of the WMR tracking trajectory in the kinematic model. From the results, we can clearly find that the trajectory tracked by using the presented integration method in Fig.4(c) is closest to the actual trajectory, which verifies its advantage over the other two methods.

In order to further verify the effectiveness of the introduced integration method under different



(a) classic (b) feed-forward (c) Integration of feed-forward and feedback

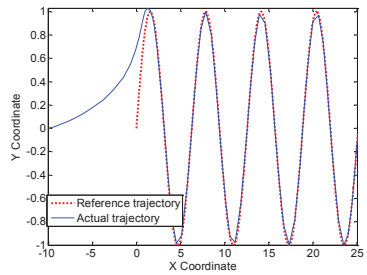
Fig. 4. The circular line trajectory results of three kinds of kinematic controllers

types of trajectories, the reference trajectory is set as sine line. The initial posture of the actual WMR is $p_R(0) = [x_r(0), y_r(0), \theta_r(0)]^T = [10, 0, 0]^T$. The results of the three control methods are shown in Fig.5(a)-(c), respectively. It can be easily observed that the convergence of the introduced integration method is faster than feed-forward method and much faster than classic method from the zoom in section.

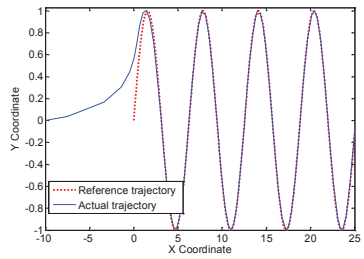
Table 1. Performance index ISE comparisons among difference kinematic controllers

kinematic controller	circular line	sine
classic	1.073	12.734
feed-forward	0.708	5.262
feed-forward and feedback	0.441	3.735

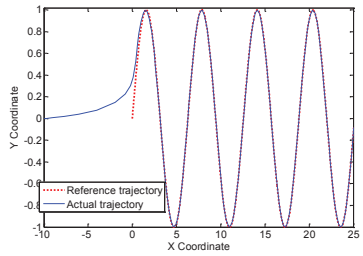
For evaluating the tracking performance of these kinematic controllers, we adopt integral square error (ISE) performance index to make comparisons. ISE is defined as $V_{ISE} = \int_0^{t_e} \left[(x_e)^2 + (y_e)^2 + (\theta_e)^2 \right] dt$, where, x_e, y_e and θ_e represent the lateral, longitudinal and direction errors in (3), respectively. ISE in different controllers in circular line and sine case are indicated in Table 1. It is clear that the ISE of the introduced integration method in both two cases is much smaller than the other two methods, which indicates that the proposed kinematic controller can provide more accurate input velocity for dynamic controller than classic and feed-forward methods.



(a) classic



(b) feed-forward



(c) Integration of feed-forward and feedback

Fig. 5. The sine line trajectory results of three kinematic controllers

4.2. Performance Tests of Proportional-Integral -Derivative based Membrane Controller

In the tests, the key controller parameters in (28) are first discussed. The scale factor K of neurons is very important. The larger the value of K is, the faster the response of PIDMC becomes, but it may cause large overshoot and make the system unstable. On the contrary, too small K may cause bad response. Since PIDMC uses the standardized learning algorithm and K is not too larger, the learning rates ζ_1, ζ_2 and ζ_3 can be larger. So the scale factor $K = 0.02$, the learning rate $\zeta_1 = 12$ of proportional coefficients, integral coefficients $\zeta_2 = 3$ and derivative coefficients $\zeta_3 = 6$, respectively; weighting factors $P = 2$ and $Q = 1$; the initial values of the weight are $w_1(0) = 0.34$, $w_2(0) = 0.32$, $w_3(0) = 0.33$; the proper parameters of PIDMC are obtained by the evolutionary algorithms [46] or trial-and-error through experiments. To test PIDMC performance, three experiments are considered: (1) the step and sawtooth tracking are used to compare with the neural network (NN)-based PID controller in [5,48]; (2) the servo controller experiment considering the actual friction model is performed; (3) the cosine curve tracking is used to compare the introduced method with the robust NN-based sliding mode controller (NNSMC). The physical parameters for control system of WMR are shown in Table. 2.

Table 2. The values of the control system's parameters

Parameter	Value	Description
m_R	$2(kg)$	Mass of the WMR
I	$0.5(kg.m^2)$	Moment of inertia
R	$0.1(m)$	Radius of the WMR
r	$0.04(m)$	Radius of each wheel

The details of the three types of tracking are as follows:

(1) Step tracking: the reference position response of the WMR system is generated from the desired step signal $rin_{step}(k) = 1$, where k is the discrete time; the sample time $S_{time} = 0.001s$. The results of PID step tracking are shown in Fig.6(a), where the red line is the desired step signal; $d_1(k)$ is a duration of disturbing signal, $d_1(k) = 0.7$ while $200 < k < 400$; $d_2(k)$ is a transient disturbing signal, $d_2(700) = 0.57$; $d_1(k)$ and $d_2(k)$ are assumed to be the uncertainties in the WMR parameters, such as the mass or inertia changes in real applications. As shown in Fig.6(a), PIDMC controller reaches the response surface much faster than normal PID method and the methods in [5,48] both at the beginning time and during the duration of disturbing time. Note that at the transient disturbing occurring time $k = 700$ from the zoom in section in Fig.6(a), the methods in [5,48] has an overshoot phenomenon, the normal PID has an big overshoot and slight oscillation phenomenon, but PIDMC avoids it.

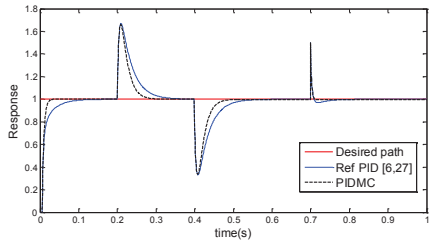
Sawtooth tracking: to verify the adaptability of PIDMC in the more complex case, the reference position response is generated from the desired sawtooth signal $rin_{saw}(k)$. The results of PID sawtooth tracking are shown in Fig.6(b). We can find that the normal PID has the worst performance since the oscillation appears at the turning point. Similar to the results in Fig.6(a), the convergence of the PIDMC response curve in sawtooth tracking is also better than the benchmark methods. From both of the simulation results, we can easily conclude that PIDMC has better response tracking performances than the methods in [5,48] and much better than the normal one.

Furthermore, for evaluating the tracking performance of these PID controllers, we make comparisons use the following performance index in Table. 3, such as: ISE, the integral of the time multiplied by the square error (ITSE), the integral of the absolute error (IAE), and the integral of the time multiplied by the absolute value of the error (ITAE). It is clear that all the performance indices of the PIDMC are much smaller than the other methods, since PIDMC has much better tracking ability.

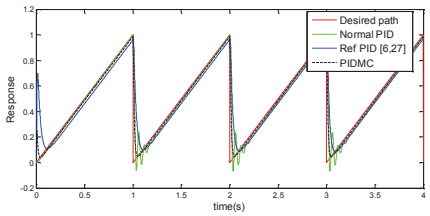
Table 3. Performance comparisons among existing PID controls and PIDMC for step and sawtooth tracking

	step		sawtooth	
	ISE	ITSE	IAE	ITAE
Normal PID	37.78	10.21	209.03	315.57
Ref PID [5,48]	29.90	7.31	141.72	247.80
PIDMC	20.21	5.10	95.47	168.12

(2) Consideration of the actual friction model: the friction is common in the servo system and has a com-



(a) Step tracking



(b) Sawtooth tracking

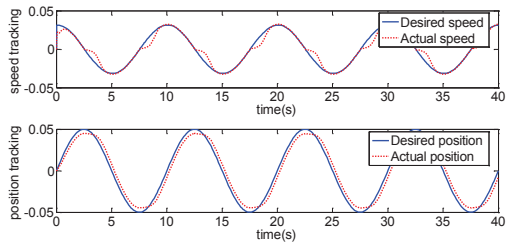
Fig. 6. The step and sawtooth tracking results of two kinds of PID controllers

plex, non-linear and uncertainty characteristics. We choose the LuGre model in [45] as the friction model. Differential equation $I\ddot{\omega} = u - F$ is the dynamic function for a simpler servo system, where I is the moment of inertia and set $I = 1$ in this case; ω is the angle of rotation; u and F are the control torque and friction torque, respectively. F can be described by the following LuGre model:

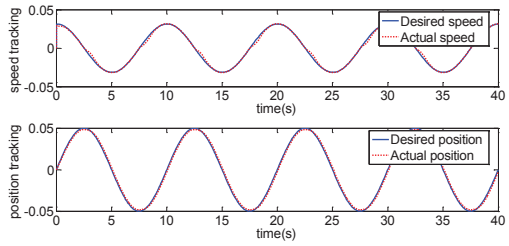
$$\begin{cases} F = \lambda_1 s + \lambda_2 \dot{s} + \beta\,\dot{\omega} \\ \dot{s} = \dot{\omega} - \dfrac{\lambda_0 |\dot{\omega}| s}{F_c + (F_s - F_c) e^{-\left(\dot{\omega}/V_s\right)^2} + \beta\,\dot{\omega}} \end{cases} \tag{31}$$

where the state variable s represents the average deformation of the contact surface; λ_1 and λ_2 are the dynamic friction parameters and set $\lambda_1 = 230, \lambda_2 = 2$; F_c, F_s, β and V_s are the static friction parameters; β is the viscous friction coefficient and set $\beta = 0.025$; V_s is the switching speed and set $V_s = 0.03$; F_c is the Coulomb's friction and set $F_c = 0.24$,F_s is the static friction and set $F_s = 0.37$.

The simulation test was carried out by using sinusoidal waveforms $0.05\sin(2\pi t)$ as the desired input trajectory. We compare the control performance comparisons between the proposed PIDMC and the conventional PID controller, where the parameters of conventional PID controller were obtained by trial and error tests and set to be $K_P = 25$, $K_I = 4$, $K_D = 7$. The initial values of the PIDMC weight, (w_1, w_2 and w_3), were set to be the same as those of conventional PID. The simulation results are shown in Fig. 7.



(a) Tracking results of conventional PID controller



(b) Tracking results of proposed PIDMC

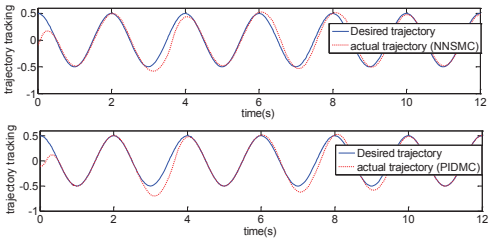
Fig. 7. The position and speed tracking results of two kinds of PID controllers considering the friction model

Fig. 7(a) shows that the conventional PID controller has the speed tracking deviation at the beginning and has the dead zone phenomenon, and the position tracking has the flat top phenomenon. While in Fig. 7(b), the response of PIDMC is in good agreement with that of position tracking. There is also a little bit deviation in speed tracking due to friction, but the results are much better than the conventional PID controller.

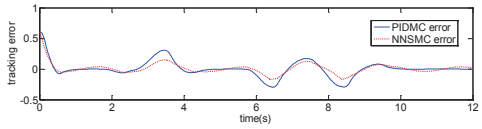
(3) Comparisons of proposed control method with the related control methods: sliding mode control is a special discontinuous control with strong robustness to external disturbances and system uncertainties. In [21], NNSMC was proposed to overcome the inherent deficiency of SMC with fixed larger upper boundedness. In order to accelerate the NN learning efficiency, a partitioned NN (two NN) structure was applied.

In the simulations, cosine $0.5\cos(2\pi t)$ is adopted as the S-typed trajectories to test the trajectory tracking effect. We consider parameter variations as the uncertainties and the external disturbances to the control system of WMR at the third and sixth second, where the parameter variations of the inertia and mass of WMR are described as follows:

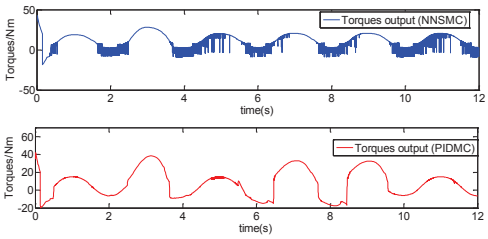
$$\begin{cases} I = 0.5 + 0.35 \ \ kg.m^2 \ \ t = 3(s) \\ m = 2 \pm 1.25 \ \ kg \ \ \ 6 < t < 9(s) \end{cases} \tag{32}$$



(a) Tracking results of NNSMC and PIDMC



(b) Tracking errors of NNSMC and PIDMC



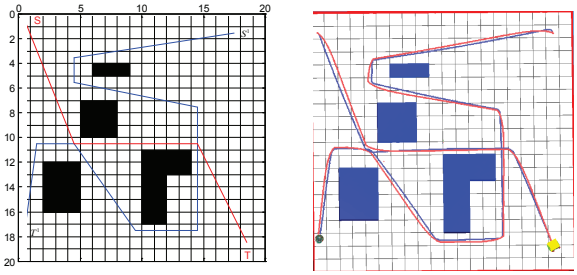
(c) Output torques of NNSMC and PIDMC

Fig. 8. Comparison of PIDMC and SMC with neural networks

Fig. 8(a) indicates that both NNSMC and PIDMC guarantee their robustness in the presence of large disturbance at different time with different types of external disturbances due to the self-learning ability of NN. PIDMC has a slightly larger tracking error than NNSMC, but it has a slightly faster tracking speed and slightly trajectory tracking stability better than NNSMC, as shown in Fig .8(b). Fig. 8(c) shows that PIDMC has a smoother control output than NNSMC since the chattering phenomenon is the the inherent deficiency of SMC.

4.3. Performance Tests of Trajectory Tracking Control Approach for Wheeled Mobile Robots

To test the feasibility and effectiveness of the proposed approach integrating outer kinematic controller with inner dynamic controller, we conduct the experiments on the mobile robot simulation software Webots 6.2.1 [14]. The experiments on the simulated robot with differential wheels were performed in an environment described in our work [46]. We have found the optimal path in the environment for WMR by using a path planning algorithm, which is presented by the red line (from S to T) in Fig.9(a). The same area with obstacles in the Webots robotics simulator is shown in Fig.9(b).



(a) path planning results in [46] (b) trajectory tracking results

Fig. 9. The results of the proposed controllers for simulated robot trajectory

In the experiments, the comparison of PIDMC and the controller in [5,48]) is considered. The grid step $S_g = 1$. The local error of the simulated robot is set to $e_l = 0.5$. The starting position of the reference path is located at grid $(0,0)$. Both of the simulated robots are located at grid $(1,0)$, where there is a small distance error between them. When trajectory tracking begins, it can be clearly found that the robot with the controller in [5,48] (red line) takes more time to find the desired path than the robot with PIDMC (blue line), and both of them walk along the reference path and go across the lane between two obstacles, and finally both of them reach the target grid. Fortunately,

PIDMC is closer to the reference path than the other one.

Furthermore, we found another sub-optimal path from S^1 to T^1 in Fig.9(a), which is described by the blue line. E-puck [14] is selected as the simulated robot and the diameter is 0.7. It is clear that the robot with PIDMC (blue line) in Fig.9(b) can also keep up with the desired path with more fewer tracking errors and much better tracking ability than the robot with the controller in [5,48] (red line). The comparisons with respect to the two performance indexes, ISE and ITSE, are shown in Table. 4. Results indicate that PIDMC achieves much smaller or better performance than the controller in [5,48].

Table 4. Performance comparisons among PIDMC controller and controller in [5,48] for trajectory tracking

	from S to T		from S^1 to T^1	
	ISE	ITSE	ISE	ITSE
Controller in [5,48]	25.37	7.43	113.62	28.48
PIDMC	4.85	2.21	20.54	6.31

5. Conclusions Remarks

This paper introduced a two-stage WMR trajectory tracking controller: inner dynamic controller and outer kinematic controller. The kinematic controller is designed by integrating feed-forward and feedback controls to improve the precision of the desired input velocity for dynamic controller. The dynamic controller is designed by introducing PIDMC which can merge a variety of ideas from various advanced PID algorithms (neural networks and experts' knowledge considered in this paper) together in a simple and easy way by means of the inter communication and inherent parallelism of membrane systems. PIDMC has good performances such as simple structure, easy use, adaptive learning, fault tolerance, powerful computing, and is appropriate for the control applications with disturbances, nonlinearity and uncertainties. Extensive experiments conducted with different conditions show the feasibility, effectiveness and robustness of the proposed trajectory tracking controller.

Following the trajectory tracking controller design of WMR in this study, our investigation may move forward to the following issues: how to extend the proposed method to design membrane controllers for more complex robotic behaviours; how to combine numerical P systems with other advanced control strategies for more control engineering applications; and how to use symbolic and numerical P systems to construct cognitive and executive architectures (planning, learning, memory or execution) of autonomous robots [8].

Acknowledgements

The work of XW and GZ is supported by the National Natural Science Foundation of China (61170016, 61373047). The work of MG, FI and RL was supported by a grant of the Romanian National Authority for Scientific Research, CNCS-UEFISCDI (project number: PN-II-ID-PCE-2011-3-0688).

References

[1] O. Arsene, C. Buiu and N. Popescu, Snups - a simulator for numerical membrane computing, *Intern. J. of Innovative Computing, Information and Control*, **7**(6) (2011), 283–295.

[2] H. Adeli and H. S. Park, A Neural Dynamics Model for Structural Optimization - Theory, *Computers and Structures* **57**(3) (1995), 383–390.

[3] H. Adeli and H. S. Park, Optimization of Space Structures by Neural Dynamics, *Neural Networks* **8**(5) (1995), 769–781.

[4] S. Ahmad, N. H. Siddique, M. Osman Tokhi, Modelling and simulation of double-link scenario in a two-wheeled wheelchair, *Integrated Computer Aided Engineering*, **21** (2) (2014), 119–132.

[5] K. K. Ahn and D. C. Thanh, Nonlinear PID control to improve the control performance of 2 axes pneumatic artificial muscle manipulator using neural network, *Mechatronics*, **16**(9) (2006), 577–587.

[6] F. Amini, and M. Zabihi-Samani, A Wavelet-based adaptive pole assignment method for structural control, *Computer-Aided Civil and Infrastructure Engineering*, **29**(6) (2014), 464–477.

[7] S. Blazic, A novel trajectory-tracking control law for wheeled mobile robots, *Robotics and Autonomous Systems*, **59**(11) (2011), 1001–1007.

[8] C. Buiu, C. I. Vasile, and O. Arsene, Development of membrane controllers for mobile robots, *Information Sciences*, **187** (2012), 33–51.

[9] A. Caponio, G. L. Cascella, F. Neri, N. Salvatore, M. Sumner, A Fast Adaptive Memetic Algorithm for Off-line and On-line Control Design of PMSM Drives, *IEEE Trans. Syst., Man, Cybern. B* **37** (1) (2007), 28–41.

[10] D. Chwa, Fuzzy adaptive tracking control of wheeled mobile robots with state-dependent kinematic and dynamic disturbances, *IEEE Transactions on Fuzzy Systems*, **20**(3) (2012), 587–593.

[11] C. Chen, T. Li, Y. Yeh and C. Chang, Design and implementation of an adaptive sliding-mode dynamic controller for wheeled mobile robots, *Mechatronics*, **19**(2) (2009), 156–166.

[12] C. Chen, T. Li and Y. Yeh, EP-based kinematic control and adaptive fuzzy sliding-mode dynamic control for wheeled mobile robots, *Information Sciences*, **179**(2) (2009), 180–195.

[13] M.A. Colomer, A. Montorib, E. Garcia and C. Fondevilla, Using a bioinspired model to determine the extinction risk of Calotriton asper populations as a result of an increase in extreme rainfall in a scenario of climatic change, *Ecological Modelling*, **81** (2014), 1–14.

[14] Cyberbotics, professional mobile robot simulation website, *www.cyberbotics.com* (2014).

[15] X. Duan, H. Deng and H. Li, A Saturation-Based Tuning Method for Fuzzy PID Controller, *IEEE Transactions on Industrial Electronics*, **60**(11) (2013), 5177–5185.

[16] R. Fierro and F. L. Lewis, Control of a nonholonomic mobile robot: back-stepping kinematics into dynamics, *IEEE Conference on Decision and Control*, (1995), 3805–3810.

[17] S. Ghosh-Dastidar, H. Adeli, A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection, *Neural Networks*, **22**(10) (2009), 1419–1431.

[18] S. Ghosh-Dastidar, H. Adeli, Improved spiking neural networks for EEG classification and epilepsy and seizure detection *Integrated Computer-Aided Engineering* **14** (3) (2007), 187–212.

[19] S. Ghosh-Dastidar, H. Adeli, Spiking neural networks, *International Journal of Neural System*, **19**(4) (2009), 295–308.

[20] E. Harinath and G. K. I. Mann, Design and tuning of standard additive model based fuzzy PID controllers for multivariable process systems, *IEEE Trans. Syst., Man, Cybern. B, Cybern*, **38**(3) (2008), 667–674.

[21] H. Hu and P. Y. Woo, Fuzzy supervisory sliding-mode and neuralnetwork control for robotic manipulators, *IEEE Trans. Ind. Electron*, **53**(3) (2006), 929–940.

[22] J. W. Jung, V. Q. Leu and T. D. Do, Adaptive PID Speed Control Design for Permanent Magnet Synchronous Motor Drives, *IEEE Transactions on Power Electronics*, **30**(2) (2015), 900–908.

[23] Y. Kanayama, Y. Kimura, F. Miyazaki and T. Noguchi, A stable tracking control method for an autonomous mobile robot, *Proceedings of the IEEE Conference Robotics and Automation*, Cincinnati (1990), 384–389.

[24] T. Kukao, H. Nakagawa and N. Adachi, Adaptive tracking control of nonholonomic mobile robot, *IEEE Transactions on Robotics and Automation*, **16**(6) (2000), 609–615.

[25] J. Li, H. Ji, L. Cao, D. Zang, R. Gu, B. Xia, Q. Wu, Evaluation and Application of a Hybrid Brain Computer Interface for Real Wheelchair Parallel Control with Multi-Degree of Freedom, *International Journal of Neural Systems*, **24**(4) (2014), Article No. 1450014.

[26] Y. Li, C. Liu, J. X. Gao, W. Shen, An integrated feature-based dynamic control system for on-line machining, inspection and monitoring, *Integrated Computer Aided Engineering* **22** (2) (2015), 187–200.

[27] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko and B. Potocnik, Time optimal path planning considering acceleration limits, *Robotics and Autonomous Systems*, **45**(3-4) (2003), 199–210.

[28] MathWorks, Matlab and Simulink for Technical Computing, *www.mathworks.com* (2015).

[29] L. Ma and S. Srinivasan, Synthetic Population Generation with Multi-level Controls: A Fitness Based Synthesis Approach and Validations, *Computer-Aided Civil and Infrastructure Engineering*, **30**(2) (2015), 135–150.

[30] A. B. Pavel, Membrane controllers for cognitive robots. *Master's thesis*, Politehnica University of Bucharest, Romania (2011).

[31] A. B. Pavel and C. Buiu, Using enzymatic numerical P systems for modeling mobile robot controllers, *Natural Computing*, **11**(3) (2012), 387–393.

[32] Gh. Păun, G. Rozenberg and A. Salomaa, *The Oxford Handbook of Membrane Computing*. NY, USA: Oxford University Press, (2010).

[33] R. C. Panda, C. C. Yu and H. Huang, PID tuning rules for SOPDT systems: Review and some new results, *ISA Transactions*, **43**(2) (2004), 283–295.

[34] G. G. Rigatos, Adaptive fuzzy control for differentially flat MIMO nonlinear dynamical systems, *Integrated Computer Aided Engineering* **20** (2) (2013), 111–126.

[35] T. J. Ren, T. C. Chen and C. J. Chen, Motion control for a two-wheeled vehicle using a self-tuning PID controller, *Control Engineering Practice*, **16**(3) (2008), 365–375.

[36] J. L. Rosselló, V. Canals, A. Oliver and A. Morro, Studying the Role of Synchronized and Chaotic Spiking Neural Ensembles in Neural Information Processing, *International Journal of Neural Systems*, **24**(5) (2014), Article No. 1440003.

[37] N. Salvatore, A. Caponio, F. Neri, S. Stasi, G. L. Cascella, Optimization of Delayed-State Kalman Filter-based Algorithm via Differential Evolution for Sensorless Control of Induction Motors, *IEEE Transactions on Industrial Electronics*, **57** (1) (2010), 385–394.

[38] S. Sun, Designing approach on trajectory-tracking control of mobile robot, *Robotics Computer-Integrated Manufacturing*, **21** (2005), 81–85.

[39] S. Shapero, M. Zhu, J. Hasler, C. J. Rozell, Optimal Sparse Approximation with Integrate and Fire Neurons, *International Journal of Neural Systems*, **24**(5) (2014), Article No. 1440001.

[40] C. I. Vasile, A. B. Pavel, I. Dumitrache and G. Păun, On the power of enzymatic numerical P systems, *Acta Informatica*, **49**(6) (2012), 395–412.

[41] N. Wang, H. Adeli, Self-constructing wavelet neural network algorithm for nonlinear control of large structures, *Engineering Applications of Artificial Intelligence*, **41** (2015), 249–

258.

[42] W. Wu, H. Chen and Y. Wang, Global trajectory tracking control of mobile robots, *ACTA Automatic Sinica*, **27**(3) (2001), 325–331.

[43] J. Wang, S. Eben Li, Y. Zheng, X.-Y. Lu Longitudinal collision mitigation via coordinated braking of multiple vehicles using model predictive control, *Integrated Computer Aided Engineering* **22** (2) (2015), 171–185

[44] Z. Wang, L. Guo, M. Adjouadi, A generalized leaky Integrate-and-Fire Neuron Model with Fast Implementation Method, *International Journal of Neural Systems*, **24**(5) (2014), Article No. 1440004.

[45] C. C. D. Wit, H. Olsson, K. J. Astrom and P. Lischinsk, A new model for control of systems with friction, *IEEE Transactions on Automatic Control*, **40**(3) (1995), 419–425.

[46] X. Wang, G. Zhang, J. Zhao, H. Rong, F. Ipate and R. Lefticaru, A modified membrane-inspired algorithm based on particle swarm optimization for mobile robot path planning, *International Journal of Computers, Communications and Control*, **10**(5) (2015), 655–668.

[47] J. Xiao, Y. Huang, Z. Cheng, J. He and Y. Niu, A hybrid membrane evolutionary algorithm for solving constrained optimization problems, *Optik*, **125**(2) (2014), 897-902.

[48] J. Ye, Adaptive control of nonlinear PID-based analog neural networks for a nonholonomic mobile robot, *Neurocomputing*, **71**(7-9) (2008), 1561–1565.

[49] S. J. Yoo, Y. H. Choi and J. B. Park, Generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: adaptive learning rates approach, *IEEE Transactions on Circuits and Systems I: Fundamental Theroy and Applications*, **53**(6) (2006), 1381–1394.

[50] G. Zhang, J. Cheng, T. Wang, X. Wang, J. Zhu, *Membrane Computing:Theory & Applications*, Science Press, Beijing, China, (2015).

[51] G. Zhang, M. Gheorghe, L. Pan, M.J. Pérez-Jiménez. Evolutionary membrane computing: a comprehensive survey and new results. *Information Sciences*, **279** (2014), 528–551.

[52] W. Zhang, J. Ning, M. Zhang, Y. Pei, B. Liu, B. Sun, Multiresolution streamline placement based on control grids, *Integrated Computer Aided Engineering* **21** (1) (2014), 47–57.

[53] G. Zhang, H. Rong, F. Neri, M.J. Pérez-Jiménez. An optimization spiking neural P system for approximately solving combinatorial optimization problems, *International Journal of Neural Systems*, **24**(5) (2014), Article No. 1440006.

[54] G. Zhang, F. Zhou, X. Huang, J. Cheng, M. Gheorghe, F. Ipate and R. Lefticaru, A novel membrane algorithm based on particle swarm optimization for solving broadcasting problems, *Journal of Universal Computer Science* **18**(13) (2012), 1821–1841.