Neurocomputing 272 (2018) 178–188

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Model-free based neural network control with time-delay estimation for lower extremity exoskeleton

Xinyi Zhang[a], Haoping Wang[a,*], Yang Tian[a], Laurent Peyrodie[b], Xikun Wang[a]

[a] *Sino-French Int. Joint Lab. Automatic Control and Signal Processing (LaFCAS), School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China*
[b] *Engineering School-HEI, Lille 59000, France*

ARTICLE INFO

Article history:
Received 3 June 2016
Revised 10 October 2016
Accepted 27 June 2017
Available online 29 June 2017

Communicated by Cheng-Wu Chen

Keywords:
Lower limb exoskeleton
Kinematic model
Neural network
Time-delay estimation
Model-free control

ABSTRACT

A model-free based neural network control with time-delay estimation (TDE-MFNNC) for lower extremity exoskeleton is presented in this paper. The lower limb exoskeleton which has 5 DOFs for each leg is established in Solidworks as a virtual prototype which is used as a platform to build the control system in SimMechanics. In an attempt to get the effective tracking trajectory, a neural network and time-delay estimation technique is added to model-free based iPD controller. The proposed controller is simulated and tested on virtual prototype to comparing with PD controller, neural network and model-free based iPD controller. This comparative study validates TDE-MFNNC as more stable and effective than the traditional controllers. Further, the kinematic model of lower limb exoskeleton added to exoskeleton also proves the stability and effectiveness of the proposed controller.

1. Introduction

A lower limb exoskeleton has been proved to help people with disabled/leg injuries to acquire or recover the walking abilities [1,2]. The exoskeleton is an external electromechanical structure worn by operator whose shape and functions correspond to the human body or limbs. It combines machine power with human intelligence in order to augment the performance of wearer. Some of the systems like BLEEX [3,4], HAL [5,6] and RoboKnee [7] have been designed for a long time and they can accomplish the important daily activities such as walking, upstairs, downstairs, etc.

Even so, the research on exoskeleton is still in the developing or the early stages. There are many problems [8–10] would be studied to average up the research level of exoskeleton, including: (1) the nonlinear and highly coupled mathematical model; (2) intelligent control method for different walking mode that is passive mode, active mode and blending mode; (3) gait planning for different people cause of various disabled or diseases; (4) signal processing and interface between exoskeleton and human body.

As such, a project has been launched to design a wearable exoskeleton to educate movement for those who lack the walking ability. We committed to the modeling and controlling of exoskeleton at the present stage. Fig. 1 illustrates the schema of the exoskeleton intelligent system which contains exoskeleton, human operator and human-machine interface.

As shown in Fig. 1, there is kinematic model in exoskeleton. The kinematic model is used to calculate relationships between terminal trajectory and joint trajectory. Also, model of exoskeleton should be established to be controlled. This paper use the virtual prototype established in Solidworks to replace the mathematical model. This virtual prototype is different from the traditional model which needs to calculate by Lagrange or Newton–Euler equation [11]. It can easily obtain the relationships between joints. The traditional mathematical model is calculated with some simplifications such as: each unit of lower limb exoskeleton is simplified into a linkage while the mass or the barycentre will be inaccuracy. Thus, the virtual prototype needs to pay attention to the details of lower limb exoskeleton and correspond to human leg, avoid simplification.

In addition to the model of exoskeleton, effective control strategy is a crucial issue in the lower limb exoskeleton as well. Position control and direct force control are the most commonly control strategy. Also, the gravity and friction compensation are used to reduce the negative influence in the technical device [12]. The computed torque control (CTC) [13] is based on mathematical model of exoskeleton which is highly complicated and easily affected by friction, disturbances and other factors. The gait trajectory adaption control [14] is also model based and need to calculate the gait trajectory with parameters of human leg. The

* Corresponding author.
E-mail addresses: hp.wang@njust.edu.cn, haoping.wang@ieee.org (H. Wang).

http://dx.doi.org/10.1016/j.neucom.2017.06.055
0925-2312/© 2017 Elsevier B.V. All rights reserved.

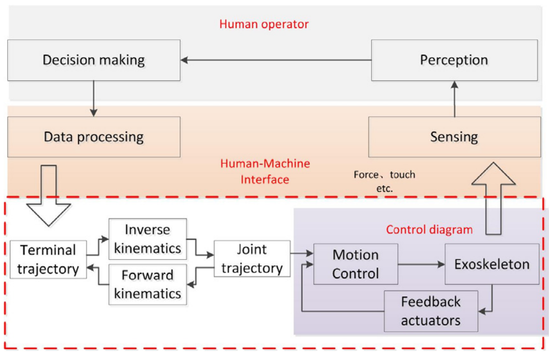

Fig. 1. Schema of the exoskeleton intelligent system.

learning control [15] of lower limb exoskeleton applies forces and learns the impedance parameters of both robot and human. But the parameters should be time-varying with large calculation. The sliding mode control in [16] is to linearize dynamic model of exoskeleton and it is model-free. But the corresponding sliding surface or the Lyapunov function which are able to ensure the desired controller performance are not always easy to choose.

Generally, the control strategies mentioned before are mostly based on the accurate model which has been already known. Although the controllers might be effective to the exoskeleton and can furnish the trajectory tracking, they need to be based on the parameters of dynamic model. While the exoskeleton is interacted/connected with human operators,it would be changeable and indeterminate. In this case, model-based controllers cannot be used to the situations. Thus, the model-free based intelligent proportional-integral-derivative control which is to compensate the unknown or changeable parts is used to control the lower limb exoskeleton in this paper.

Model-free control has an ultra-local model [17,18] to replace the mathematical model and the uncertainties of the model can be estimated by time-delay [19,20] (TDE-MFC). As such, it can be real-time compensation for the uncertainties or the disturbances in model. Usually, the loop is closed by intelligent proportional-integral-derivative controller (iPID) [21]. Based on the idea, this paper will propose a model free based iPID and neural network control with time-delay estimation (TDE-MFNNC) strategy to control the lower limb exoskeleton. This new strategy combines the RBF neural network [22,23] into model free based iPID control to compensate the disturbance and uncertainties. With the virtual prototype and new controller, the torque given to lower limb exoskeleton can be obtained to detect whether it is within tolerable bounds.

This paper is divided into five sections. Section 2 gives the brief introduction about virtual prototyping and kinematic model of exoskeleton. In Section 3, the time-delay estimation with model-free based neural network is described. Section 4 gives comparison of PD controller [24], neural network, TDE-MFC and TDE-MFNNC. Co-simulation results of lower limb exoskeleton with kinematic model are also given in the section. Finally, there is a conclusion and future planning.

2. Exoskeleton and mathematical model

2.1. Virtual prototyping and kinematic model

Usually, the structure of lower limb exoskeleton is considered to be similar to human leg [25]. In this study research, 5 DOFs at each leg is exploited: two at the hip, one at the knee, two at the ankle. Fig. 2 displays the disposition of DOFs and physical prototype of lower limb exoskeleton. Let the *x*-axis point towards to right, the

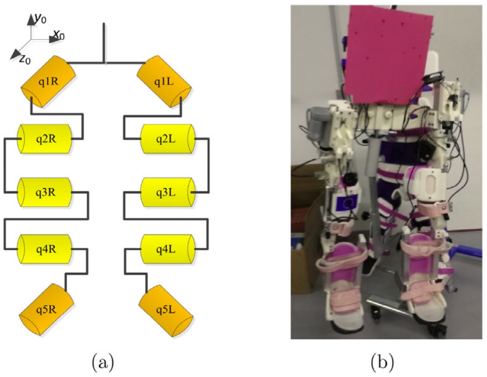

Fig. 2. Structure design of lower limb exoskeleton. (a) Disposition of DOFs. (b) Physical prototype (HEI, Lille).

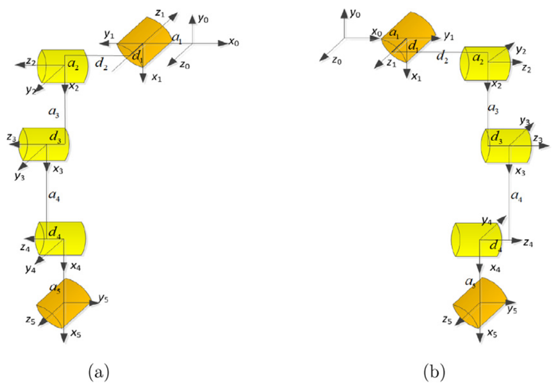

Fig. 3. The kinematics configuration in the lower limb exoskeleton. (a) Right leg. (b) Left leg.

z-axis point forward and the *y*-axis be determined by the right-rule. The transformation between each body follows the rule which is easily to establish kinematic model. The degrees of freedoms are in Fig. 2(a). q1 indicates the abduction/adduction at hip. q2 indicates the flexion/extension at hip in left leg. q3 shows the flexion/extension at knee. q4 shows the flexion/extension at ankle. q5 express the eversion/inversion 2(b) shows the physical prototype of lower limb exoskeleton. While the exoskeleton is connected to human body, there is a bandage on exoskeleton which is to hold human body. There are totally 10 DOFs in the lower limb exoskeleton while there are two legs in an exoskeleton. For simulation, virtual prototype is more useful than physical prototype and the kinematic model is used to give joints trajectory to the virtual prototype.

Robot kinematics, the basic analysis of motion control, describes the transformation between the spatial pose and joints. Denavit–Hartenberg (D–H) matrix is the most commonly method to derive the forward kinematics. Despite the 10 DOFs, the forward kinematics can be de-constructed because the two kinematic chains are completely independently. Fig. 2 displays the kinematics configuration of left leg Fig. 3(a) and right leg 3(b) in the lower limb exoskeleton which follows the D–H standard form and the transformation in virtual prototype. The transformation of kinematics follows the rule of D–H model.

The D–H parameters of each leg are listed in Table 1. In this paper, we just display the corresponding transformation matrix of left leg in Eq. (1). Readers can calculate the transformation matrix of right leg following the parameters in Table 1. Lets define the abbreviation by s_1 for $\sin(\theta_1)$, and c_1 for $\cos(\theta_1)$. Further, s_{23} represents $\sin(\theta_2+\theta_3)$ and s_{234} represents $\sin(\theta_2+\theta_3+\theta_4)$. c_{23} is the abbreviation of $\cos(\theta_2+\theta_3)$ and c_{234} represents $\cos(\theta_2+\theta_3+\theta_4)$.

Table 1
D–H parameters of each leg.

	Left leg				Right leg			
Joint$i$	α_{i-1}(rad)	a_{i-1}	d_i	θ_i(rad)	α_{i-1}(rad)	a_{i-1}	d_i	θ_i(rad)
1	0	a_1	d_1	$\theta_1-\pi/2$	π	$-a_1$	$-d_1$	$\theta_1+\pi/2$
2	$-\pi/2$	a_2	d_2	θ_2	$-\pi/2$	a_2	d_2	θ_2
3	0	a_3	d_3	θ_3	0	a_3	d_3	θ_3
4	0	a_4	d_4	θ_4	π	a_4	$-d_4$	θ_4
5	$\pi/2$	a_5	0	θ_5	$\pi/2$	a_5	0	$\theta_5+\pi$

$$
{}^0_1T=\begin{bmatrix} s_1 & c_1 & 0 & a_1\\ -c_1 & s_1 & 0 & 0\\ 0 & 0 & 1 & d_1\\ 0&0&0&1\end{bmatrix},\quad {}^1_2T=\begin{bmatrix} c_2 & -s_2 & 0 & a_2\\ 0 & 0 & 1 & d_2\\ -s_2 & -c_2 & 0 & 0\\ 0&0&0&1\end{bmatrix},
$$
$$
{}^2_3T=\begin{bmatrix} c_3 & -s_3 & 0 & a_3\\ s_3 & c_3 & 0 & 0\\ 0 & 0 & 1 & d_3\\ 0&0&0&1\end{bmatrix},\quad {}^3_4T=\begin{bmatrix} c_4 & -s_4 & 0 & a_4\\ s_4 & c_4 & 0 & 0\\ 0 & 0 & 1 & d_4\\ 0&0&0&1\end{bmatrix},
$$
$$
{}^4_5T=\begin{bmatrix} c_5 & -s_5 & 0 & a_5\\ 0 & 0 & -1 & 0\\ s_5 & c_5 & 0 & 0\\ 0&0&0&1\end{bmatrix} \tag{1}
$$

According to Eq. (1), the forward kinematics model of lower limb exoskeleton can be obtained in Eq. (2), which describes left leg, by matrix multiplication:

$$
\begin{aligned}
{}^0_5T &= {}^0_1T\,{}^1_2T\,{}^2_3T\,{}^3_4T\,{}^4_5T\\
&=\begin{bmatrix} c_1s_5+s_1c_5c_{234} & c_1c_5-s_1s_5c_{234} & s_1s_{234} & p_x\\ s_1s_5-c_1c_5c_{234} & s_1c_5+s_5c_1c_{234} & -c_1s_{234} & p_y\\ -s_{234}c_5 & s_{234}s_5 & c_{234} & p_z\\ 0&0&0&1\end{bmatrix}
\end{aligned} \tag{2}
$$

$$
p_x=a_1+(d_2+d_3+d_4)c_1+a_2s_1+a_3c_2s_1+a_4s_1c_{23}+a_5s_1c_{234} \tag{3}
$$

$$
p_y=(d_2+d_3+d_4)s_1-a_2c_1-a_3c_2c_1-a_4c_1c_{23}-a_5c_1c_{234} \tag{4}
$$

$$
p_z=d_1-a_3s_2-a_4s_{23}-a_5s_{234} \tag{5}
$$

While p_x, p_y, p_z represents the respectively position of x, y, z axis to the end of exoskeleton. The kinematic model can be used to test workspace of robot.

Conversely, the inverse kinematics is more important than kinematic model in engineering application [26]. However, the solution of inverse trigonometric function is not the only one, so it leads to the multiple solutions of robot inverse kinematic. To solve the problem, it usually combines with the structural characteristics of the robot and the actual pose. It is well-known that the inverse kinematics can be obtained analytically if the chain has five or less DOF. Therefore, we obtain the angle of each joint by algebraic method. Also, there is constraint conditions: (1) when the exoskeleton is walking, the body which links two hip joints is always in a horizontal state and the exoskeleton is always vertical upward, (2) the contact between foot of exoskeleton and the ground is completely exposed. Then, a feasible analytical solution of the left leg at lower limb exoskeleton is presented in Eqs. (6)–(10). Corresponding results in right leg can be obtained in the same way.

$$
\theta_1=\arctan 2\left(\frac{d_2+d_3+d_4}{\rho},-\sqrt{1-\left(\frac{d_2+d_3+d_4}{\rho}\right)^2}\right)-\arctan 2(p_x-a_1,p_y) \tag{6}
$$

$$
\begin{gathered}
\theta_2=\arctan 2\left(\frac{k}{\sqrt{k_1^2+k_2^2}},-\sqrt{1-\left(\frac{k}{\sqrt{k_1^2+k_2^2}}\right)^2}\right)-\arctan 2(k_1,k_2)\\
\begin{cases} k_1=2a_3s_1(p_x-a_1)-2a_3p_yc_1-2a_2a_3\\ k_2=2a_3(p_z-d_1)\\ k=s_1^2(p_x-a_1)^2+c_1p_y^2+(p_z-d_1)^2+a_2^2\\ \quad +a_3^2-a_4^2-2s_1(p_x-a_1)(c_1p_y-a_2)+2a_2c_1p_y\end{cases}
\end{gathered} \tag{7}
$$

$$
\theta_3=\arccos\frac{\left\{\begin{matrix}(p_x-a_1)^2+p_y^2+(p_z-d_1)^2-2a_2S_1(p_x-a_1)\\ +2a_2C_1p_y-(d_2+d_3+d_4)^2+a_2^2-a_3^2-a_4^2\end{matrix}\right\}}{2a_3a_4} \tag{8}
$$

$$
\theta_4=pi/2-\theta_2-\theta_3 \tag{9}
$$

$$
\theta_5=-\theta_1 \tag{10}
$$

Thus, the kinematic model can provide joint trajectory for dynamic model.

2.2. Dynamic model in MATLAB-SimMechanics

The dynamic model differs from kinematic model. It gives relationships between torque and joint trajectory. Based on kinematic model, the joint trajectory can be obtained. Lagrange equation is commonly used to gain dynamic model and its general form is:

$$
\frac{d}{dt}\frac{\partial L}{\partial \dot q}-\frac{\partial L}{\partial q}=\tau \tag{11}
$$

where $L=T-V$. T and V represent kinetic energy and potential energy respectively.τ represents the variable of generalized coordinates. represents the torque of system. With Eq. (11), the dynamic model of lower limb exoskeleton can be expressed as:

$$
M(q)\ddot q+C(q,\dot q)+G(q)=\tau \tag{12}
$$

where $M(q)$ is the inertia matrix which is symmetric, positive definite. $C(q,\dot q)$ is Coriolis term and $G(q)$ is the gravity term.

It is obvious that the dynamic model is nonlinear and highly coupled with large computation. There are also some assumptions such as the distribution of shape and quality is uniform and regular. Also the friction and series of uncertainties are not taken into account. Thus, the traditional dynamics may be different from actual model. This paper uses the dynamic model in MATLAB-SimMechanics to replace the traditional dynamics.

SimMechanics is a simulation toolbox in MATLAB which is designed for mechanical system to set up an environment for rigid body and its movement. The block diagram is also used in SimMechanics to build model which is more close to actual. The virtual prototype firstly established in Solidworks and we can use SimMechanicsLink to automatically generate the dynamic model in SimMechanics which is used to accomplish the control system for walking and to verify the effective of TDE-MFNNC. The dynamic model in SimMechanics is illustrated in Fig. 4. There are two subsystems corresponding to left leg and right leg. In this paper, the prototype serves as a platform to confirm the performance of controller.

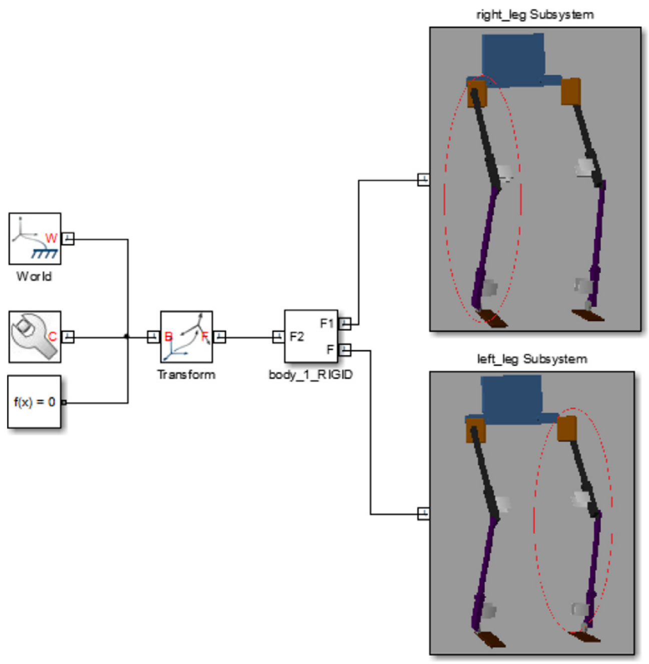

Fig. 4. Schema of the exoskeleton intelligent system.

3. TDE-MFNNC design

Considering of the virtual prototype, a model-free based iPID Control (MFC) is introduced into lower limb exoskeleton. To improve the robustness and eliminate tracking error, a neural network is added to model free controller thus obtain the model-free based neural network controller (MFNNC). For uncertainties and disturbances, time-delay technique is used for estimation in this paper.

3.1. TDE-MFC design

3.1.1. Model-free control

The model-free Control replaced the complex mathematical model by an ultra-local model. Combining to the lower limb exoskeleton, the dynamic model can be yield in Eq. (13).

$$q^{(\nu)}(t) = F(t) + \alpha\tau(t) \tag{13}$$

where $q^{(\nu)}(t)$ is νth derivative order of the output $q(t)$, which is the joint angle. The value of v is practically frequently chosen as 1 or seldom as 2, $\tau(t)$ is the control input which is torque given to the exoskeleton in this paper, α is a non-physical parameter selected by the practitioner, F denotes the total lumped unknown dynamics of the system, friction, disturbances and other uncertainties.

Close the loop via the intelligent proportional-integral-derivative controller (iPID) which is defined as:

$$\tau(t) = -\frac{F(t) - q_d^{(\nu)}(t) - (K_P e(t) + K_I \int e(t) + K_D \dot{e}(t))}{\alpha} \tag{14}$$

where $q_d^{(\nu)}(t)$ is the νth derivative order of the desired joint trajectory. $e(t) = q_d(t) - q(t)$ is the trajectory tracking error.

Normally to avoid the high derivative order of output, one sets $K_I = 0$ and obtains the corresponding iPD controller as follow:

$$\tau(t) = -\frac{F(t) - q_d^{(\nu)}(t) - (K_P e(t) + K_D \dot{e}(t))}{\alpha} \tag{15}$$

Combining Eqs. (13) and (15) yields the error function:

$$e^{(\nu)}(t) + K_D \dot{e}(t) + K_P e(t) = 0 \tag{16}$$

In Eq. (16), there are none of the unknown parts and uncertainties of the plant, i.e. F is eliminated. It is thus left with a linear equation. When $\nu = 2$, it is a 3-order equation while $\nu = 3$ is 2-order. As such, the task turns into tuning parameters of K_P and K_D.

In reality for real application, the calculated control torque in (15) cannot be applied directly because of F which is lumped unknown dynamics and needed to be estimated online.

3.1.2. Time-delay estimation

The estimation of F is a crucial term for the practical implementation. A time delay estimation technique which possesses excellent robustness in view of uncertain system, unknown disturbance and parameters variations [27–29] is introduced into the MFC for estimation of F.

According to Eq. (13), the unknown value of F can be computed as:

$$F(t) = q^{(\nu)}(t) - \alpha\tau(t) \tag{17}$$

But it cannot be estimated directly because of the absence of control information at current instant t, i.e. $\tau(t)$ is not available for the estimation of $F(t)$. The essential idea of the time delay estimation technique is to voluntarily introduce a small delay t_0 in the control design, and then to use past observations regarding both control input and system response to compensate the unknown dynamics, unexpected disturbances and other uncertainties simultaneously. Assuming that $F(t)$ is continuous and the delay t_0 is sufficiently small, then the lumped unknown dynamics can be estimated approximately as:

$$F(t) \cong \hat{F}(t) = F(t - t_0) = q^{(\nu)}(t) - \alpha\tau(t - t_0) \tag{18}$$

The proposed control torque can be realized and implemented as:

$$\tau(t) = -\frac{\hat{F} - q_d^{(\nu)}(t) - (K_P e(t) + K_D \dot{e}(t))}{\alpha} \tag{19}$$

Thus, Eq. (19) is the control rate of time-delay estimation based model-free controller that we proposed to be used for lower limb exoskeleton. Noting that there is a simple structure and only parameter need to be chosen by practitioner, K_P and K_D can be tuned with pole placement method.

Define the error as $\xi(t) = \hat{F}(t) - F(t) = F(t - t_0) - F(t)$, and combining (13) and (19) yield:

$$e^{(\nu)}(t) + K_D \dot{e}(t) + K_P e(t) = \xi(t) \tag{20}$$

If the time delay t_0 is sufficiently small, the error will be close to zero, i.e. $\xi(t) \to 0$. It is thus similar to Eq. (16) and the closed loop stability can be guaranteed. But from practical engineering implementation point of view, $e(t)$ is not necessary to be zero due to the time delay t_0. With the appropriate selected K_P and K_D, $e(t)$ is just under bounded. To compensate the estimation error, a RBF neural network is combined with model-free control in the following.

3.2. TDE-MFNNC design

To eliminate $\xi(t)$ which is the error of estimation, an additional RBF neural network [30–32] is added to time-delay estimation with model-free based iPD controller (TDE-MFNNC). Control torque now is:

$$\tau(t) = -\frac{\hat{F} - q_d^{(\nu)}(t) - (K_P e(t) + K_D \dot{e}(t))}{\alpha} + \tau_{NN}(t) \tag{21}$$

Set $\nu = 1$ and the error function comes to:

$$\dot{e}(t) + K_D \dot{e}(t) + K_P e(t) = \xi(t) + \alpha\tau_{NN}(t) \tag{22}$$

$$\dot{e}(t) = -Ke(t) + f(t) + \mu_{NN}(t) \tag{23}$$

where $K = \frac{K_P}{1+K_D}$, $f(t) = \frac{\xi(t)}{1+K_D}$, $\mu_{NN}(t) = \frac{\alpha}{1+K_D}\tau_{NN}(t)$.

An ideal approximation of neural network is introduced into model-free controller in this paper, thus it is given as:

$$\hat{f}(x|W^*) = {W^*}^T h(x) \tag{24}$$

And the optimal weight W^* should be satisfied with $W^* = \arg\min[f(t)]$. Where $h(x) = \exp(-\frac{\|x-c\|^2}{2b^2})$ is the Gaussian function for the hidden layer, with $c = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{m1} & \cdots & c_{mn} \end{pmatrix}$ the center vector of Gaussian function and $b = (b_1, \dots, b_n)^T$ the width of Gaussian function.

Define the approximate error as:

$$\sigma = f(t) - \hat{f}(x|W^*) \tag{25}$$

Then the error function will be given as:

$$\dot{e}(t) = -Ke(t) + (\hat{f}(x|W^*) + \sigma) + \mu_{NN}(t) \tag{26}$$

To compensate the estimation error, i.e. to let $\lim_{t\to\infty} e(t) = 0$. The following equation should be satisfied:

$$\mu_{NN}(t) = -f(x|\hat{W}) = -\hat{W}^T h(x) \tag{27}$$

whose adaptive rate of weight is proposed as [33]:

$$\dot{\hat{W}} = -\gamma e^T P h(x) \tag{28}$$

The error function then given into:

$$\dot{e}(t) = -Ke(t) + (W^* - \hat{W})^T h(x) + \sigma \tag{29}$$

Design the Lyapunov function as:

$$V = \frac{1}{2}e^T P e + \frac{1}{2\gamma}(W^* - \hat{W})^T (W^* - \hat{W}) \tag{30}$$

where γ is a positive constant and P is a positive definite matrix which satisfy:

$$K^T P + PK = -Q \tag{31}$$

Take the derivative of V, then comes to:

$$\begin{aligned}
\dot{V} &= \frac{1}{2}\dot{e}^T P e + \frac{1}{2}e^T P \dot{e} + \frac{1}{\gamma}(W^* - \hat{W})^T \dot{\hat{W}} \\
&= \frac{1}{2}((W^* - \hat{W})^T h(x) + \sigma - Ke(t))^T P e + \frac{1}{\gamma}(W^* - \hat{W})^T \dot{\hat{W}} \\
&\quad + \frac{1}{2}e^T P((W^* - \hat{W})^T h(x) + \sigma - Ke(t)) \\
&= -\frac{1}{2}e^T(t)\left(K^T P + PK\right)e(t) + (W^* - \hat{W})^T e^T(t) P h(x) \\
&\quad + \frac{1}{\gamma}(W^* - \hat{W})^T \dot{\hat{W}} \\
&= -\frac{1}{2}e^T(t) Q e(t) + e^T(t) P \sigma \\
&\quad + \frac{1}{\gamma}(W^* - \hat{W})^T [\dot{\hat{W}} + \gamma e^T(t) P h(x)]
\end{aligned} \tag{32}$$

Combining the adaptive rate equation (28) into (32):

$$\dot{V} = -\frac{1}{2}e^T(t) Q e(t) + e^T(t) P \sigma \tag{33}$$

In equation, $-\frac{1}{2}e^T(t)Qe(t) \le 0$, therefore, the approximate error σ can be sufficiently small by RBF neural network then make $\dot{V} \le 0$.

For the convenience of readers, the control diagram of TDE-MFNNC is shown in Fig. 5.

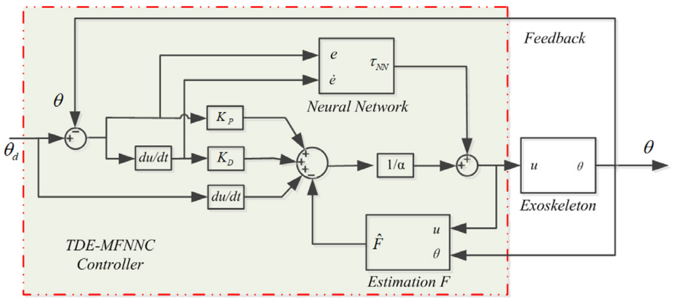

Fig. 5. Block diagram of the TDE-MFNNC.

Table 2
Parameters of different control method.

Control method	Parameters		
	K_P	K_D	α
PD	[200;300;300;300;200]	[20;20;20;20;10]	
TDE-MFC	[200;300;300;300;200]	[20;20;20;20;10]	[10;10;10;10;10]

4. Co-simulation results

In this section, we will compare the proposed method to PD controller, neural networks and model-free based iPD controller to validate the performance of it. Then combing the kinematic model to lower limb exoskeleton to create the experiment platform for practical use.

4.1. Comparison results without kinematics

In the modern rehabilitation therapy, training for those who do not have the ability to walk is always in passive mode. The passive mode is about following the given trajectory. For people who has ability to walk is under the active mode. The exoskeleton provides power to the wearer by contact with each other. The torque which is given to the exoskeleton should be within the range that human can bear. So the output of controller which is the torque should not be too large.

To demonstrate the performance of the proposed method, we start from PD controller and neural network. The desired trajectory for each joint is:

Joint1: $0.1\sin(t) - pi/2$, Joint2: $\sin(t) + pi/2$, Joint3: $\sin(t)$, Joint4: $\sin(t)$, Joint5: $pi/2$.

The parameters of PD controller, TDE-MFC are in Table 2. For neural network, the construction of RBF neural network is 2-5-1 and the parameters of Gaussian function is $b = 0.2$ and $c = [-2\ -1\ 0\ 1\ 2]$. Parameters in TDE-MFNNC are the same as TDE-MFC and neural network (NN).

Fig. 6 demonstrates trajectory tracking error of each joint with different time delay in TDE-MFNNC. From the diagram, the tracking error of each joint has a large floating in the first 1 s.

With comparison, it is obviously that the error is small when time delay is small. As such, this paper gives $t_0 = 0.001$ s.

Figs. 7–11 illustrates the simulation results of joint 1–5 with comparison of four controllers. The results represent the angle, error and torque of joint 1–5 respectively in different controller.

As observed in Figs. 7–11, the tracking error is extremely small of each joint while the proposed method, TDE-MFNNC, has a best tracking performance. Tracking error with TDE-MFNNC of each joint is in $\pm$ 0.01*rad*. Control torque of each joint is also in the acceptable range of human body.

The simulation results indicate the convergence performance of TDE-MFNNC. It shows that the model-free based neural network with time-delay estimation technique can control the lower

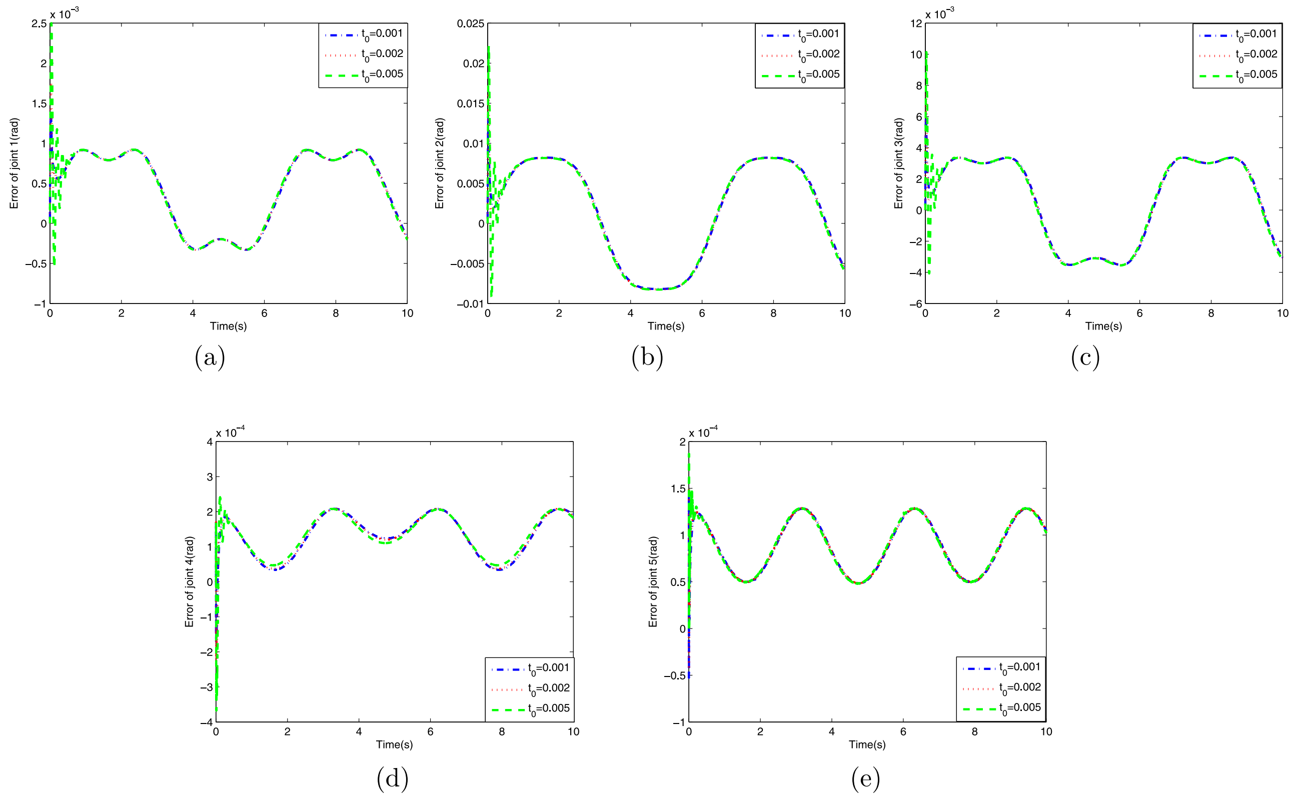

Fig. 6. Tracking error with different time delay of each joint in TDE-MFNNC.

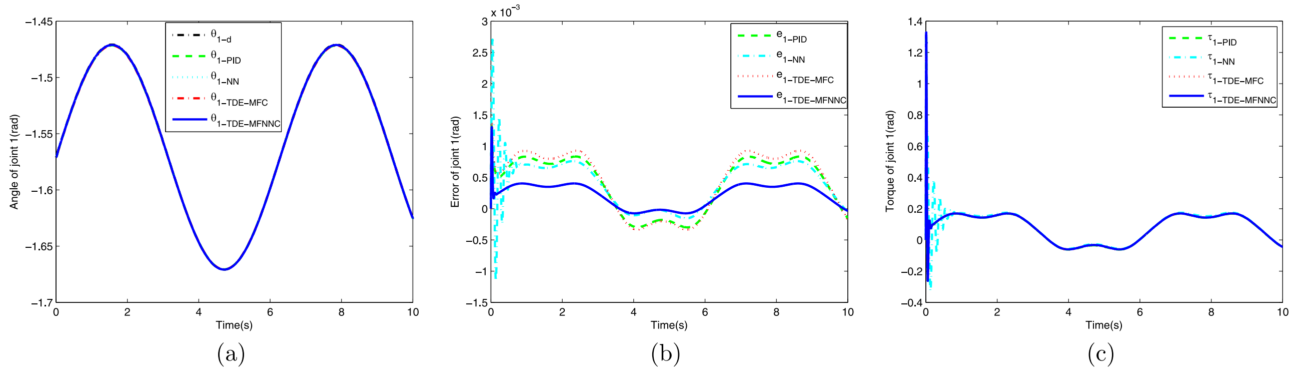

Fig. 7. Simulation results of joint 1 with four different controllers. (a) Trajectory tracking. (b) Trajectory error. (c) Control torque.

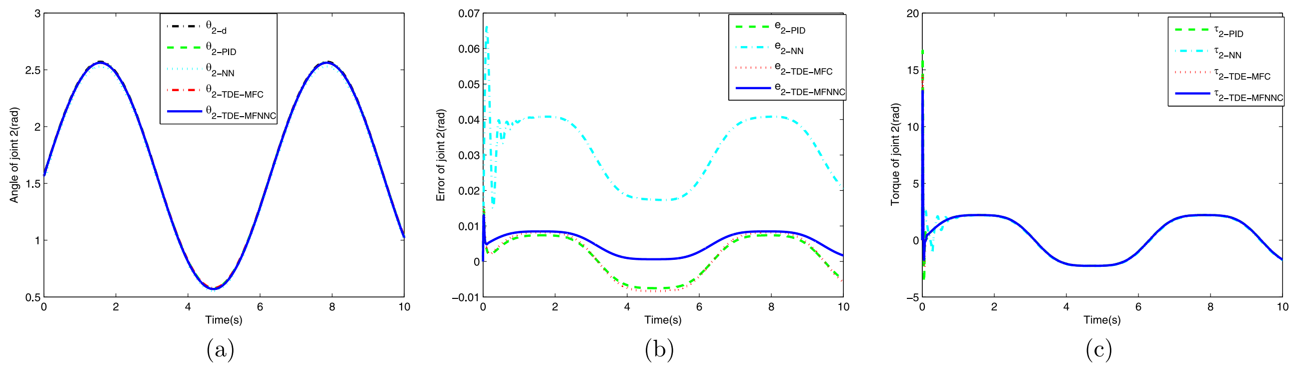

Fig. 8. Simulation results of joint 2 with four different controllers. (a) Trajectory tracking. (b) Trajectory error. (c) Control torque.

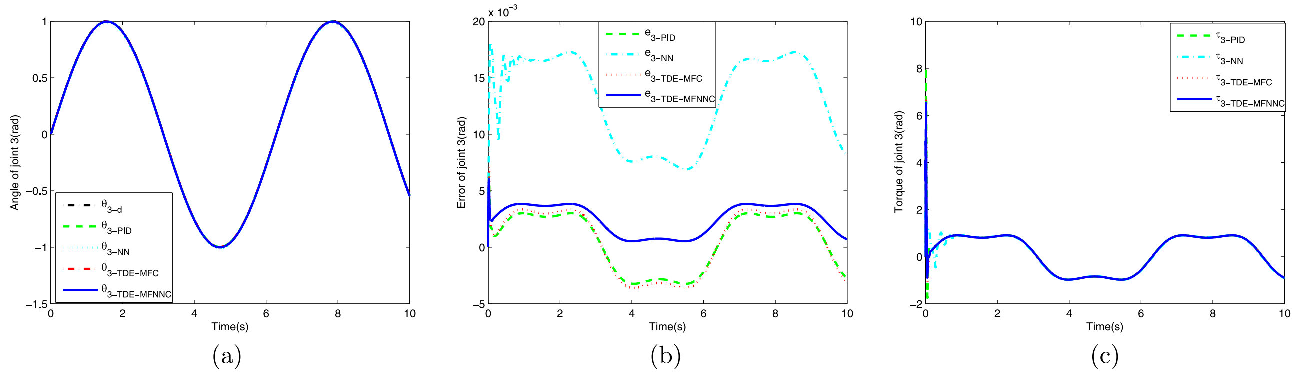

Fig. 9. Simulation results of joint 3 with four different controllers. (a) Trajectory tracking. (b) Trajectory error. (c) Control torque.

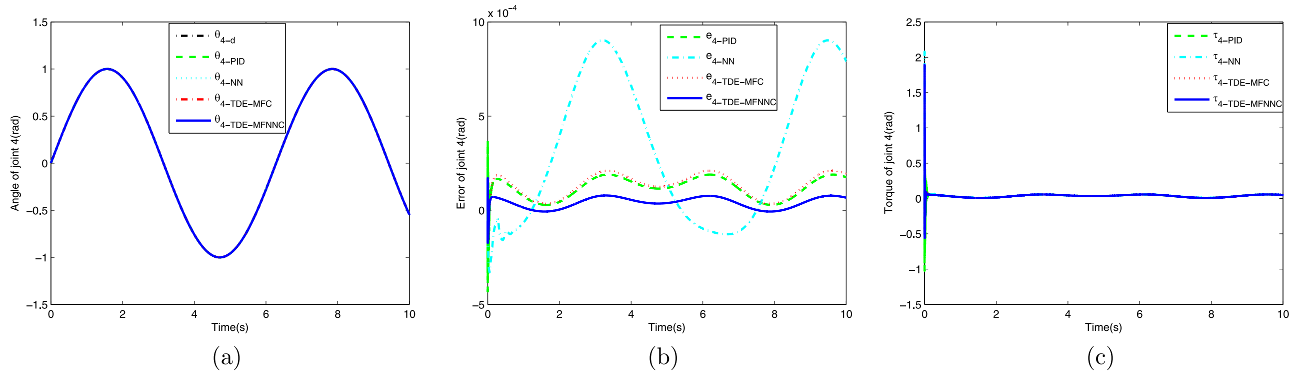

Fig. 10. Simulation results of joint 4 with four different controllers. (a) Trajectory tracking. (b) Trajectory error. (c) Control torque.

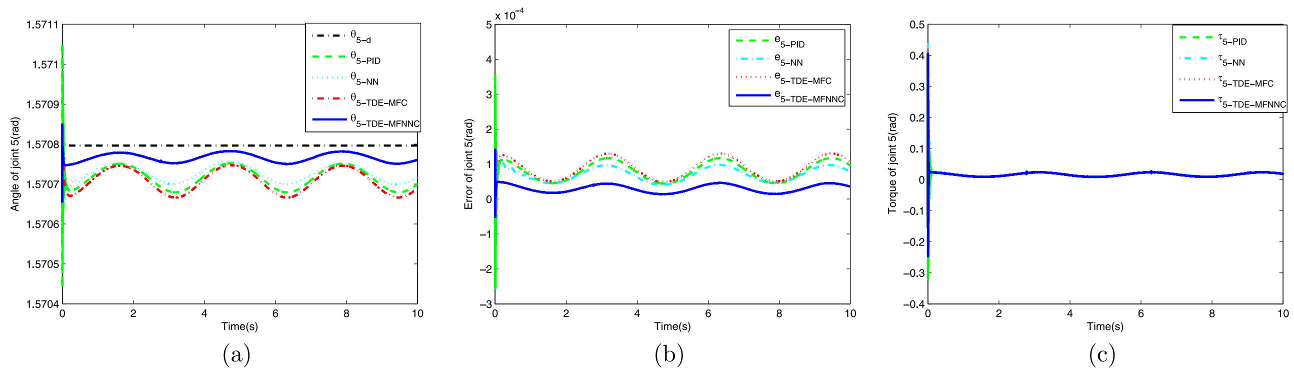

Fig. 11. Simulation results of joint 5 with four different controllers. (a) Trajectory tracking. (b) Trajectory error. (c) Control torque.

limb exoskeleton and has an excellent control. Comparing to the traditional neural network, PD control and TDE-MFC, the proposed method, TDE-MFNNC, has more stable inputs and trajectory tracking performance without big fluctuation in the beginning.

4.2. Simulation with kinematics

In the practical application, the target points or the trajectory of exoskeleton will be known instead of the angle in exoskeleton. Therefore, we should use the inverse kinematics to achieve the reference trajectory of the joint space as shown in Fig. 1. A simulation platform with kinematics is in Fig. 12.

Parameters in kinematic model are corresponding to the virtual prototype. In this paper, we give the parameters below:

$a_1 = 0.195\,\text{m}, a_2 = 0.045\,\text{m}, a_3 = 0.410\text{m}, a_4 = 0.463\,\text{m}, a_5 = 0.024\,\text{m}$

$d_1 = 0.090\,\text{m}, d_2 = 0.005\,\text{m}, d_3 = 0.015\,\text{m}, d_4 = -0.010\,\text{m}$

The trajectory in the task space and the parameters of the TDE-MFNNC controller are listed in Table 3 for left and right leg. The controller parameters used in left leg and right leg is the same. The corresponding simulation results in consideration of the kinematics are shown in Fig. 13(left leg) and Fig. 14 (right leg).

According to Fig. 13, the former 5 pictures is the joint angle of 5 links. The performance of trajectory is well. The last picture is the trajectory of task space, i.e. it is the trajectory of left foot. It is the same in Fig. 14. The error and torque in each leg are in Fig. 15 and Fig. 16.

With kinematic model, the exoskeleton works on the basis of terminal trajectory. By calculating with Eqs. (6)–(10), angle of each

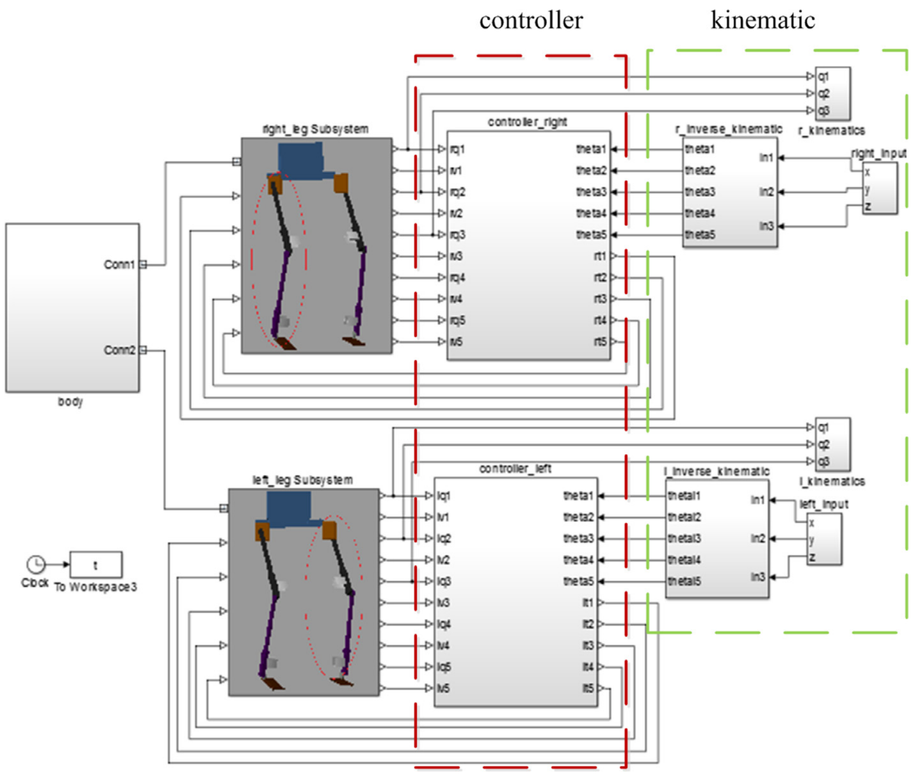

Fig. 12. Simulation platform with kinematics.

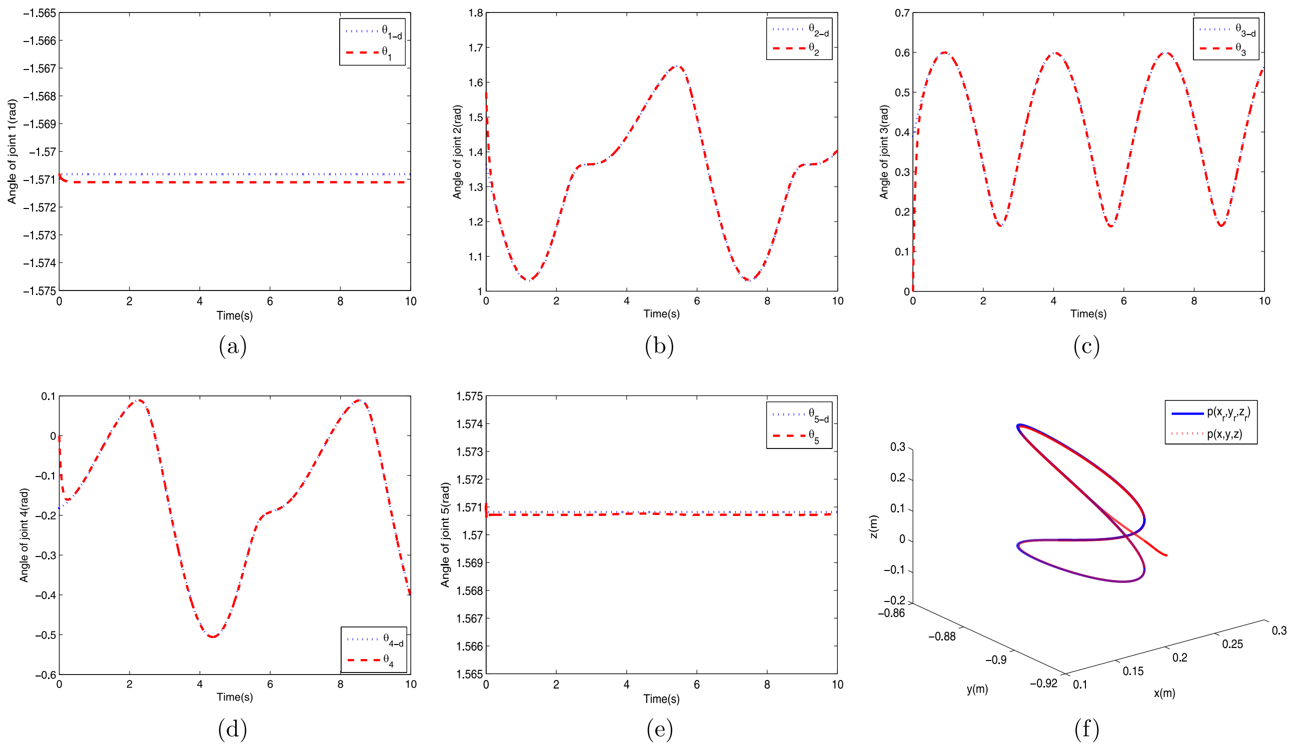

Fig. 13. The simulation results of each joint with kinematics of left leg.

joint can be obtained to the controller. Then torque is given to drive the exoskeleton. Also, the control torque is in the acceptable range of human body and the controller has a satisfactory tracking performance.

5. Conclusion

A model-free based neural network with time-delay estimation (TDE-MFNNC) is proposed to control the lower limb exoskeleton in

Table 3
Desired trajectory and parameters for simulation with kinematics.

Trajectory	Left leg		Right leg
	$p_{x_r} = 0.205$ $p_{y_r} = 0.025\sin(2t - \pi/4) - 0.885$ $p_{z_r} = 0.21\sin(t) + 0.09$		$p_{x_r} = -0.205$ $p_{y_r} = 0.025\sin(2t - \pi/4) - 0.885$ $p_{z_r} = 0.21\sin(t + \pi) + 0.09$
Parameters	K_P [200;300;300;300;200]	K_D [20;20;20;20;10]	α [10;10;10;10;10]

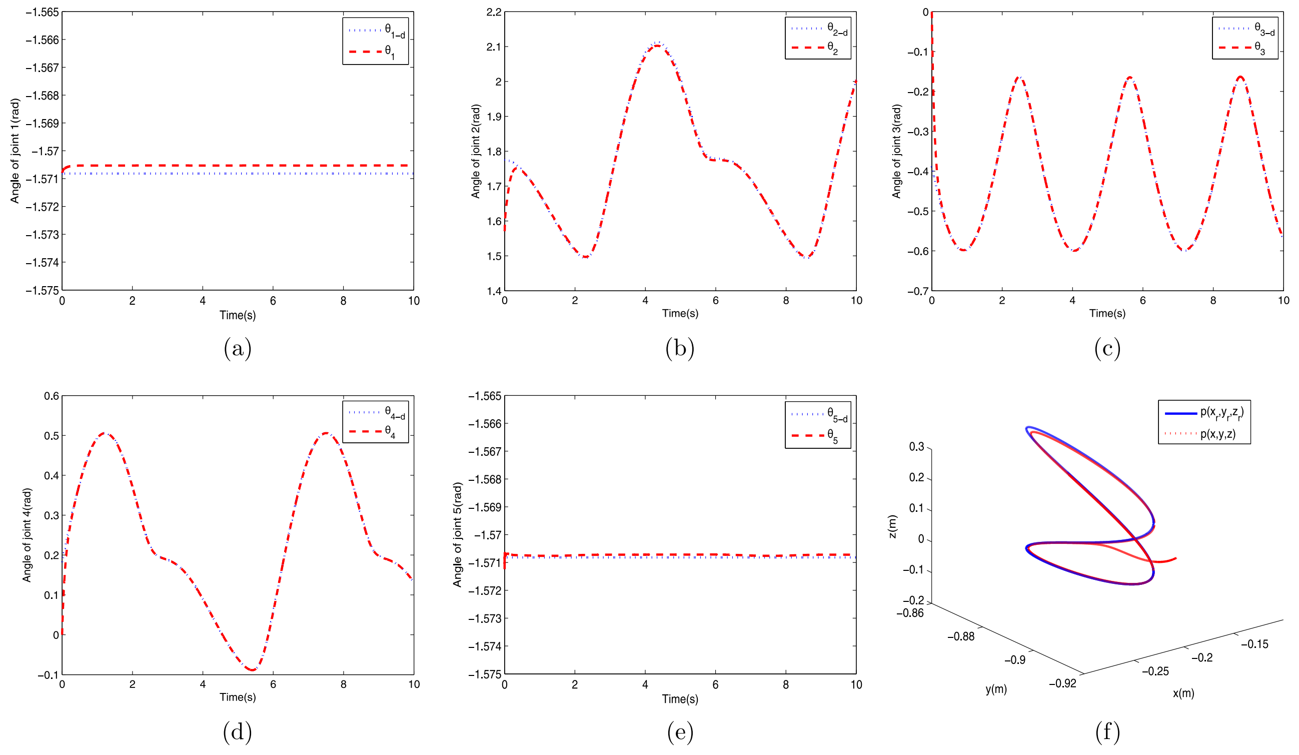

Fig. 14. The simulation results of each joint with kinematics of right leg.

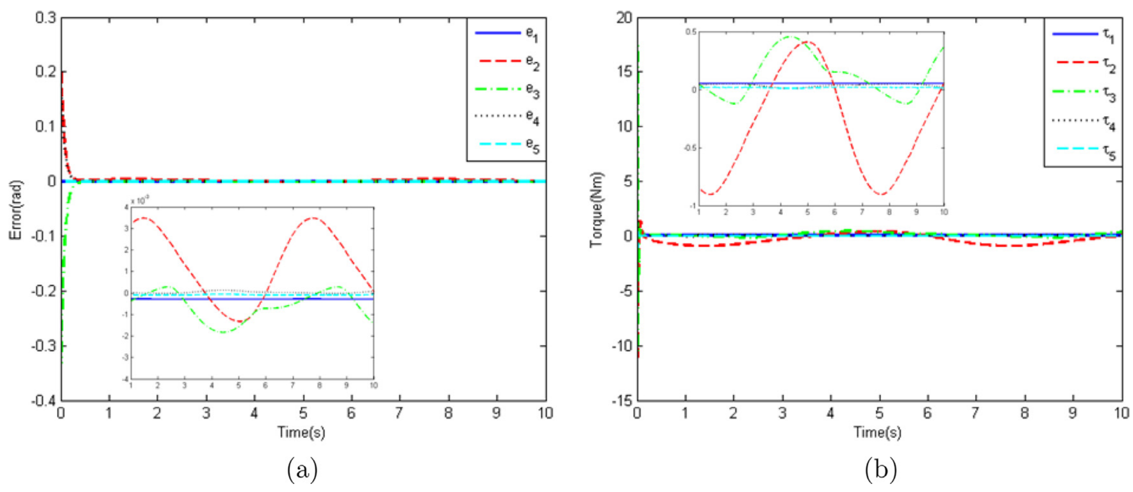

Fig. 15. the error and torque results of left leg. (a) error. (b) torque.

this paper. The virtual prototype established in Solidworks serves as a platform in order to build the control system and test the performance. In comparison with PD controller, neural network and model-free based iPD controller, the TDE-MFNNC performs well in trajectory tracking and the control torque is more stable. There are also simulation results which are co-simulated with kinematic model to verify and test the proposed controller. In the future, the project will focus on mechanical design of lower limb exoskeleton to establish an actual model and the controllers will be tested in the actual model with human operator.

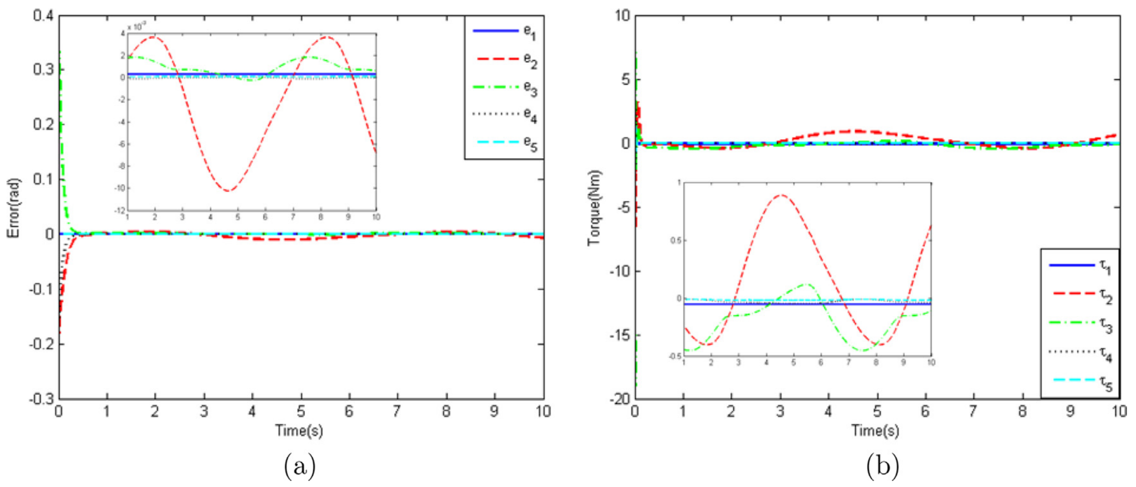

Fig. 16. The error and torque results of right leg. (a) error. (b) torque.

Acknowledgment

This work was partially supported by International Science & Technology Cooperation Program of China (2015DFA01710), by the National Natural Science Foundation of China (61304077), and by the 11th Jiangsu Province Six talent peaks of high level talents (2014_ZBZZ_005)

References

[1] K. Anam, A.A. Al-Jumaily, Active exoskeleton control systems: state of the art, Proc. Eng. 41 (2012) 988–994.
[2] C.J. Yang, J.F. Zhang, Y. Chen, Y.M. Dong, Y. Zhang, A review of exoskeleton-type systems and their key technologies, proceedings of the institution of mechanical engineers, Part C: J. Mech. Eng. Sci. 222 (8) (2008) 1599–1612.
[3] A.B. Zoss, H. Kazerooni, A. Chu, Biomechanical design of the berkeley lower extremity exoskeleton (BLEEX), IEEE/ASME Trans. Mechatron. 11 (2) (2006) 128–138.
[4] J. Ghan, R. Steger, H. Kazerooni, Control and system identification for the berkeley lower extremity exoskeleton (BLEEX), Adv. Robot. 20 (9) (2012) 989–1014.
[5] A. Tsukahara, Y. Hasegawa, Y. Sankai, Gait support for complete spinal cord injury patient by synchronized leg-swing with HAL, in: IEEE/RSJ International Conference on Intelligent Robots & Systems, 2011, pp. 1737–1742.
[6] H. Kawamoto, Y. Sankai, Power assist method based on phase sequence and muscle force condition for HAL, Adv. Robot. 19 (7) (2012) 717–734.
[7] J.E. Pratt, B.T. Krupp, C.J. Morse, S.H. Collins, The roboknee: an exoskeleton for enhancing strength and endurance during walking, in: IEEE International Conference on Robotics & Automation, 2004, pp. 2430–2435.
[8] T. Yan, M. Cempini, C.M. Oddo, Review of assistive strategies in powered lower-limb orthoses and exoskeletons, Robot. Auton. Syst. 64 (2015) 120–136.
[9] W. Koh, B.H. Mccormick, Brain microstructure database system: an exoskeleton to 3d reconstruction and modeling, Neurocomputing 44C46 (3) (2002) 1099–1105.
[10] S. Mefoued, A second order sliding mode control and a neural network to drive a knee joint actuated orthosis, Neurocomputing 155 (C) (2015) 71–79.
[11] X.Y. Zhang, H.P. Wang, Y. Tian, Z.F. Wang, L. Peyrodie, Modeling, simulation & control of human lower extremity exoskeleton, in: 34th Chinese IEEE Control Conference (CCC), 2015, pp. 6066–6071.
[12] Z. Li, C.Y. Su, L. Wang, Nonlinear disturbance observer-based control design for a robotic exoskeleton incorporating fuzzy approximation, IEEE Trans. Ind. Electron. 62 (9) (2015) 5763–5775.
[13] P. Masarati, Computed torque control of redundant manipulators using general-purpose software in real-time, Multibody Syst. Dyn. 32 (4) (2014) 403–428.
[14] J.F. Zhang, Y.M. Dong, C.J. Yang, 5-link model based gait trajectory adaption control strategies of the gait rehabilitation exoskeleton for post-stroke patients, Mechatronics 20 (3) (2010) 368–376.
[15] R. Lu, Z. Li, C.Y. Su, Development and learning control of a human limb with a rehabilitation exoskeleton, IEEE Trans. Ind. Electron. 61 (7) (2014) 3776–3785.
[16] X. Wang, X. Li, J. Wang, Data-driven model-free adaptive sliding mode control for the multi degree-of-freedom robotic exoskeleton, Inf. Sci. 327 (2015) 246–257.
[17] M. Fliess, C. Join, Model-free control, Int. J. Control 86 (12) (2013) 2228–2252.
[18] M. Fliess, C. Join, Stability margins and model-free control: a first look, in: European IEEE Control Conference (ECC), 2014, pp. 454–459. 2014
[19] J. Na, X. Ren, Y. Xia, Adaptive parameter identification of linear SISO systems with unknown time-delay, Syst. Control Lett. 66 (1) (2014) 43C50.
[20] H.P. Wang, Y. Tian, S.Y. Ni, Intelligent proportional trajectory tracking controllers: using ultra-local model and time delay estimation techniques, in: Control and Decision Conference IEEE, 2015.
[21] M. Fliess, C. Join, Model-free control and intelligent PID controllers: towards a possible trivialization of nonlinear control, Syst. Identif. 15 (1) (2009) 1531–1550.
[22] L.J. Herrera, H. Pomares, I. Rojas, Global and local modelling in RBF networks, Neurocomputing 74 (16) (2011) 2594–2602.
[23] A. Saranli, B. Baykal, Complexity reduction in radial basis function (RBF) networks by using radial b-spline functions, Neurocomputing 12 (1) (2010) 183–194.
[24] K.J. Astrom, T. Hagglund, PID controllers: Theory, design and tuning, Instrument Society of America, Research Triangle Park NC, 1995.
[25] A.M. Dollar, H. Herr, Lower extremity exoskeletons and active orthoses: challenges and state-of-the-art, IEEE Trans. Robot. 24 (1) (2008) 144–158.
[26] Z.F. Wang, L. Peyrodie, H. Cao, O. Agnani, H.P. Wang, Slow walking model for children with multiple disabilities via an application of humanoid robot, Mech. Syst. Signal Process. 68 (2016).
[27] Z. Wu, H. Su, J. Chu, State estimation for discrete markovian jumping neural networks with time delay, Neurocomputing 73 (10C12) (2010) 2247–2254.
[28] K. Youcef-Toumi, S.-T. Wu, Input/output linearization using time delay control, J. Dyn. Syst. Meas. Control 114 (1) (1991) 2601–2606.
[29] G.R. Cho, P.H. Chang, S.H. Park, M. Jin, Robust tracking under nonlinear friction using time-delay control with internal model, IEEE Trans. Control Syst. Technol. 17 (6) (2009) 1406–1414.
[30] H. Jafarnejadsani, J. Pieper, J. Ehlers, Adaptive control of a variable-speed variable-pitch wind turbine using radial-basis function neural network, IEEE Trans. Control Syst. Technol. 21 (6) (2013) 2264–2272.
[31] C.F. Hsu, Adaptive dynamic RBF neural controller design for a class of nonlinear systems, Appl. Soft Comput. 11 (8) (2011) 4607–4613.
[32] K. Meng, Z.Y. Dong, D.H. Wang, A self-adaptive RBF neural network classifier for transformer fault analysis, IEEE Trans. Power Syst. 25 (3) (2010) 1350–1360.
[33] L. Cheng, Z.G. Hou, M. Tan, Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model, Automatica 45 (10) (2009) 2312–2318.

Xinyi Zhang is a graduate student at Automation School of NJUST, China. Her research interests include lower limb exoskeleton and model-free based neural network control with time-delay estimation.

Haoping Wang (M'09, SM'13) received the Ph.D. degree in Automatic Control from Lille University of Science and Technology (LUST), France, in 2008. He is currently Professor at Automation School, Deputy Director of Sino-French Engineering School, and Scientific and Executive director of Sino-French Int. Joint Laboratory of Automatic Control and Signal Processing, Nanjing University of Science and Technology, China. He was research fellows at MIS Laboratory of Picardie University and at LAGIS of LUST, France. His research interests include the theory and applications of hybrid systems, visual servo control, observation design, exoskeleton robotics, friction modeling and compensation, modeling and control of diesel engines, biotechnological processes and wind turbine systems.

Yang Tian received the Ph.D. degree in Automatic Control from Ecole Centrale de Lille, France, in 2010. She is currently an Associate Professor at Automation School, NJUST, China. Her research interests include nonlinear and hybrid systems theory and applications, and algebraic-differential methods in control and estimation theory.

Laurent Peyrodie is Chairman-Professor of Biomedical Signal Processing Unit and co-Chairman of Medical Engineering and Health at High study of engineering School (Lille). His research focuses on biomedical processing. He is interested in EEG signal, particularly on EEG signal filtering and epilepsy seizure detection. Work on the EEG has been extended to other physiological signals including Center of Pressure (COP). He is working with hospital from Groupement Hospitalier de l?université Cahtolique de Lille (GHICL). He is actually the leader project of exoskeleton for children with multiples disabilities in North of France.

Xikun Wang is the graduate student at Automation School of Nanjing University of Science & Technology, his research is in lower extremity of exoskeleton. In the research team, his part is stable gait planning, co-simulation through ADAMS a nd MATLAB and test on the NAO.