J Intell Robot Syst (2007) 49:151–169
DOI 10.1007/s10846-007-9133-1

Robust Recurrent Neural Network Control of Biped Robot

Wu Yilei · Song Qing · Yang Xulei

Received: 10 August 2006 / Accepted: 11 January 2007 /
Published online: 21 March 2007

Abstract In this paper, a recurrent neural network (RNN) control scheme is proposed for a biped robot trajectory tracking system. An adaptive online training algorithm is optimized to improve the transient response of the network via so-called conic sector theorem. Furthermore, L_2-stability of weight estimation error of RNN is guaranteed such that the robustness of the controller is ensured in the presence of uncertainties. In consideration of practical applications, the algorithm is developed in the discrete-time domain. Simulations for a seven-link robot model are presented to justify the advantage of the proposed approach. We give comparisons between the standard PD control and the proposed RNN compensation method.

Keywords Conic sector condition · Discrete robot model · Fault tolerant control · L_2-stability · Online adaptive training · Recurrent neural network

Main Notation List

$\hat{V}(k)$, $\hat{W}(k)$	Estimated weight of RNN output layer and hidden layer, respectively.
$V^*(k)$, $W^*(k)$	Optimal weight of RNN output layer and hidden layer, respectively.
$\hat{y}_{rnn}(k)$, $x(k)$	Output and state vector of RNN, respectively.

Y. Wu (✉) · Q. Song · X. Yang
School of Electrical and Electronic Engineering, Nanyang Technological University,
Nanyang Avenue, Singapore 639798, Singapore
e-mail: pg04474334@ntu.edu.sg

Q. Song
e-mail: eqsong@ntu.edu.sg

$H(\cdot)$, $f(r(k))$	Activation function and cost function of RNN, respectively.
$\alpha(k)$, β	Adaptive learning rate and scaling factor of RNN training algorithm, respectively.
$e(k)$	Tracking error of robot control system.
$r(k)$, $e_v(k)$	Training and estimation error of RNN, respectively.
$\phi(k)$	Feedback signal of closed-loop systems.
$J_i(k)$	Jacobian matrix of error gradient of RNN.
H_1, H_2	Feedforward and feedback operator in closed-loop systems, respectively.
$\theta_d(k)$, $\theta(k)$	Command and actual joint angle of robots, respectively.
K_P, K_D	Coefficients of PD controller.
m, I, l, d	Various parameters of the robot model.
$\tau_0(k)$	Control torque contributed by PD controller.
$\tau_c(k)$	Compensation torque contributed by RNN.
$M(\theta(k))$	Inertia matrix of robots.
$C(\theta(k-1), \theta(k-2))$	Coriolis/centripetal torque matrix of robots.
$G(\theta(k))$	Gravity vector of robots.
$D(\theta(k), \theta(k-1))$	Disturbances including noise, static and dynamic frictions.
$F(\theta(k), \theta(k-1))$	Failure function of robot caused by model uncertainty.
k	Sampling index.

1 Introduction

Recently recurrent neural network (RNN) has been extensively studied in the area of robot control. One of the most remarkable features of RNN is its capability in modelling time-behavior of dynamic processes. This can provide robotic system with strong adaptability to the change of working environments [1, 2]. Moreover, combination of neural network and conventional linear controller is becoming a major direction in robot controller design [3]. The method simultaneously uses a linear controller that provides the basic tracking performance as well as a neural network that compensates various uncertainties, which also named as fault tolerant control (FTC) [4]. For example, in [5, 6] feedforward neural networks were integrated into PD controller to improve the tracking performance. In [7] fuzzy neural network was used and robustness was guaranteed by H_∞ design methods.

Stability is an important aspect of control systems that must be proved before they can be implemented in real applications. Many studies have been carried out on the issue for RNN [8–12] and neural network controlled robotic systems [13–15]. The difficulty of these works consists in the computational complicacy of obtaining the boundary condition of the stability. This is because that a weight adjustment may affect the entire network states evolution due to the inherent feedback structure of RNN.

In this paper, we introduce a generalized framework of robustness analysis for RNN training algorithms. A hybrid controller that consists of PD and RNN is proposed. The design target is to improve the transient response of the system in the presence of system faults. Sufficient conditions on the L_2-stability of training is derived via conic sector theorem such that robustness is ensured by restricting the training parameters within cone conditions. A 7-link robot model is employed as control objective in simulations. Main segments of the model include legs, thighs, trunk and feet. The motion of the robot is constrained on the sagittal plane. As a consequence, the total number of degrees of freedom will be limited enough. The work is developed in the discrete-time domain, which can benefit from direct implementation on digital signal processor (DSP).

The paper is organized as follows: In Section 2 some basics of robotic systems are introduced, including robot structures, walking patterns etc. In Section 3 we present a controller which is constructed by PD and RNN. Then robustness analysis for the online adaptive training algorithm of RNN is derived via conic sector theorem in Section 4. Simulations are demonstrated in Section 5 to verify the effectiveness of theoretical results. Finally conclusion is given in Section 6.

2 Robot Dynamics

The rigid biped robot under consideration is shown in Fig. 1. A detail description of its structure can be found in [7]. For later reference we would like to give an immediate brief review. The robot consists of 7 links, which represent trunk, thigh, shin, metatarsal respectively. The links are joined together at 6 pin joints: two hip joints, two knee joints and two ankle joints. The six joints are ideally rotational. Each of them has one degree of freedom and is driven by independent electric DC motors. For each segment, there are four parameters: the mass of the link–m_i, moment of inertia about the center of gravity (COG)–I_i, the length of the link–l_i, and the distance between the COG and the lower joint–d_i, where $i = 1, \cdots, 6$.

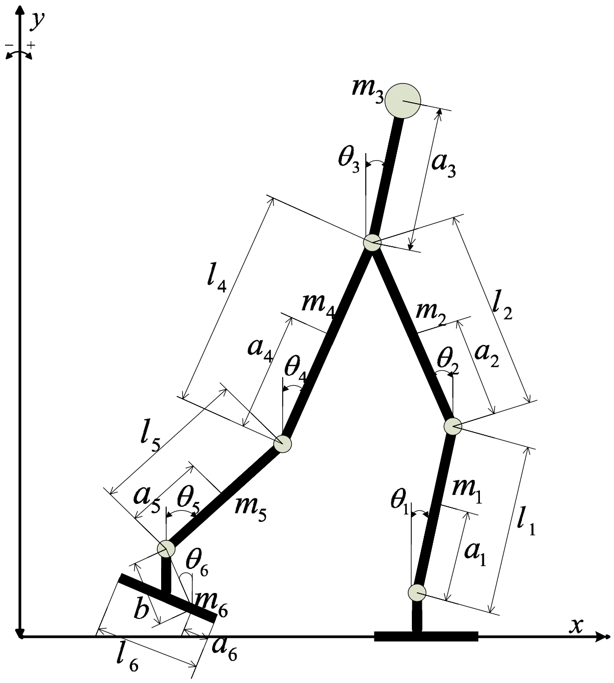

Fig. 1 Structure of a 7-link rigid biped robot

In literature the most popular analytical model of human walking can be described as it is performed so as to have the least expenditure of energy. Under this framework, the walking pattern of a biped robot can be roughly divided into three phases: single support phase, double support phase and transition phase. Among them, the single support phase is a predominant portion that its duration is longer than the other two's. Hence our study mainly focuses on this period. We use the set of the angles θ of each link with the vertical as controlling objectives. In mathematics, the dynamic equations of non-kick action of above described single-leg-supporting phase can be expressed in a analog model by

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) + D(\theta,\dot{\theta}) + F(\theta,\dot{\theta},t) = \tau \tag{1}$$

Where $\dot{\theta}$ and $\ddot{\theta}$ are joint angle velocities and accelerations respectively, $M(\theta) \in R^{6\times6}$ is the inertia matrix, $C(\theta,\dot{\theta}) \in R^{6\times6}$ is the coriolis/centripetal torque matrix, $G(\theta) \in R^{6\times1}$ is gravity vector, $\tau \in R^{6\times1}$ denotes the input torque vector, $D(\theta,\dot{\theta}) \in R^{6\times1}$ is static and dynamic friction and other disturbance torques, and $F(\theta,\dot{\theta},t)$ stands for the unknown fault that occurs in robot manipulator. The following assumptions of biped robot dynamics are necessary [16].

Assumption 1

(a) *The inertia matrix $M(\theta)$ is symmetric, positive definite and both $M(\theta)$ and $M^{-1}(\theta)$ are uniformly upper and lower bounded.*
(b) *The inertia matrix $M(\theta)$ is also kinetic energy matrix of the manipulator and the kinetic energy can be written as $\dot{\theta}^T M(\theta)\dot{\theta}/2$, where superscript T denotes the transpose of the vectors and matrices.*
(c) *The matrix $\dot{M}(\theta) - 2C(\theta,\dot{\theta})$ is skew-symmetric.*
(d) *The unknown system uncertainty is bounded.*

In this paper, a hybrid controller that integrated PD and RNN is proposed. In functionality, the PD is utilized as basic feedback control for trajectory following and the RNN facilitates to compensate the nonlinearity and system faults [17, 18]. Suppose in a fault free condition, robot dynamics of Eq. 1 can be transformed into a nominal form as

$$\ddot{\theta} = M^{-1}(\theta)[\tau - C(\theta,\dot{\theta})\dot{\theta} - G(\theta) - D(\theta,\dot{\theta}) - F(\theta,\dot{\theta},t)] \tag{2}$$

If the robot parameters $M(\theta)$, $C(\theta,\dot{\theta})$ and $G(\theta)$ are exactly known, then with a PD controller $y_{pd} = K_P e + K_D \dot{e}$, the computed torque τ_0 can be designed as

$$\tau_0 = M(\theta)(\ddot{\theta}_d + K_P e + K_D \dot{e}) + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) \tag{3}$$

Where $e = \theta_d - \theta$ is the tracking error vector, K_P, K_D are scalars. Substitute the computed torque (3) into the nominal system (2), we obtain the following error dynamics

$$\ddot{e} + K_D\dot{e} + K_P e - M^{-1}(\theta)D(\theta,\dot{\theta}) - M^{-1}(\theta)F(\theta,\dot{\theta},t) = 0 \tag{4}$$

However there may not exist available value for K_P, K_D with which the tracking error vector converges to zero. Furthermore when a fault $F(\theta,\dot{\theta},t)$ occurs, the PD controller may not be able to response as fast as required. The situation will be even worse in case that an integrator is introduced to reduce offset. Hence a feedforward

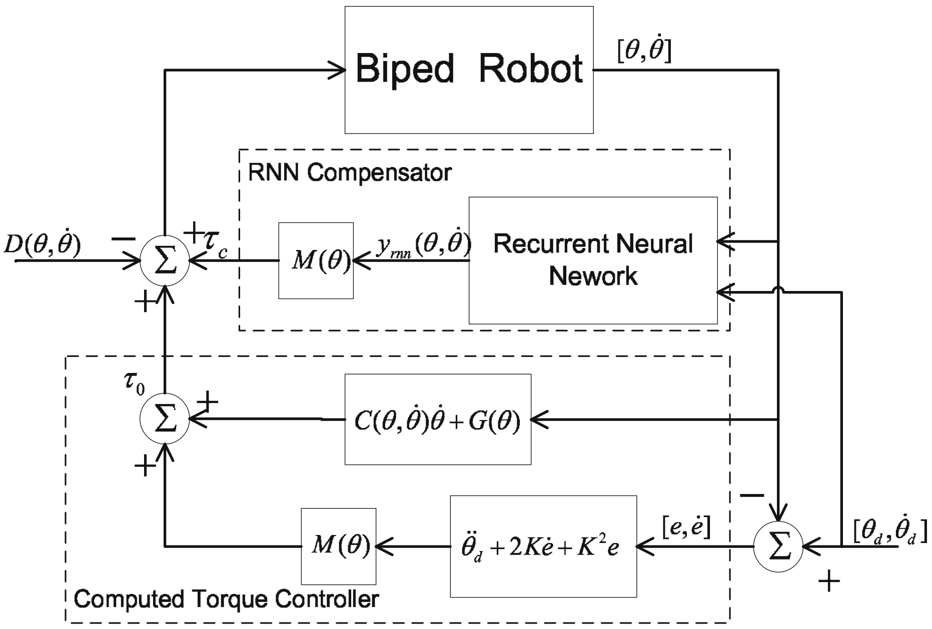

Fig. 2 Block diagram of RNN control scheme

controller is obviously desirable to compensate the error and disturbance terms, and in turn to improve system response speed and reduce the steady state error. Because RNN is specially powerful in identifying dynamic nonlinear processes, thus we employ RNN in robot system to recover the tracking performance in the occurrence of fault. Hereby the computed torque output should be reconfigured as

$$\tau = \tau_0 + \tau_c(\hat{y}_{rnn}) \tag{5}$$

The block diagram of the whole control system is displayed in Fig. 2.

3 RNN Compensator

Now consider a RNN with 24 input nodes, 6 output nodes, and 50 hidden layer neurons. The output and hidden layer weights of the RNN can be expressed (in matrix form) by $V \in R^{6\times 50}$ and $W \in R^{50\times 24}$, respectively. Upon an input vector $u(k)$, the corresponding RNN output can be expressed as

$$\hat{y}_{rnn}(k) = \hat{V}(k)H(\hat{W}(k)\cdot x(k)) \tag{6}$$

where $H(\cdot)$ is the nonlinear activation function, and $x(k)$ is the state vector that consists of 12 external input entries and 12 output feedback entries

$$x(k) = [u^1(k), \cdots, u^6(k), u^1(k-1), \cdots, u^6(k-1), \hat{y}^1_{rnn}(k-1), \cdots, \\ \hat{y}^6_{rnn}(k-1), \hat{y}^1_{rnn}(k-2), \cdots, \hat{y}^6_{rnn}(k-2)]^T \tag{7}$$

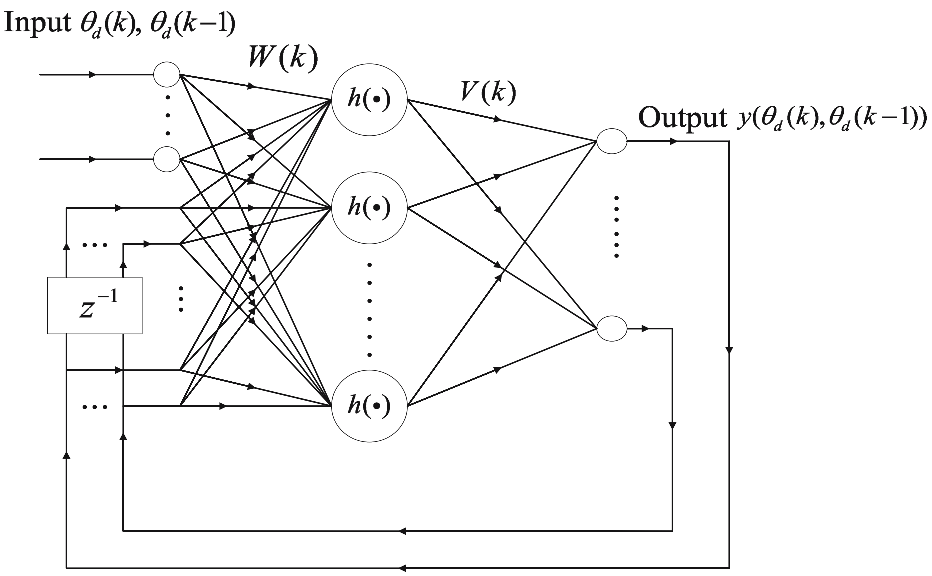
Input $\theta_d(k)$, $\theta_d(k-1)$
$W(k)$
$h(\bullet)$
$h(\bullet)$
$h(\bullet)$
$V(k)$
Output $y(\theta_d(k), \theta_d(k-1))$
z^{-1}

Fig. 3 Structure of an external feedback recurrent neural network

Superscript *i* denotes the *i*th entry of each vector. In estimating a desired signal of N samples $\{d(0), d(1), \cdots, d(N)\}$, the RNN estimation error $r(k)$ is defined by

$$r(k) = d(k) - \hat{y}_{rnn}(k) \tag{8}$$

The diagram of the RNN is shown in Fig. 3.

However the RNN (6) is a digital system while the robot (2) is modelled in continuous-time domain. We must digitize the robot model before carrying out formal analysis. For this purpose, a sampler is placed on θ to define

$$\theta(k) = \theta(kT_s) \in R^{6\times 1} \tag{9}$$

Where T_s represents the sampling period. One of the most popular approaches to digitize the nonlinear dynamics in Eq. 2 is to use Euler's rule. By this method, the discrete-time model of the angular speed and angular acceleration of joint angles can be obtained as

$$\begin{cases} \dot{\theta}(k)T_s = \theta(k) - \theta(k-1) \\ \ddot{\theta}(k)T_s = \dot{\theta}(k) - \dot{\theta}(k-1) \end{cases} \tag{10}$$

Substitute Eq. 10 into Eq. 2, then robot dynamics is transformed to an input-output framework

$$\begin{aligned} \theta(k+1) = 2\theta(k) - \theta(k-1) + T_s^2 M^{-1}(\theta(k))[\tau_0(k) + \tau_c(k) \\ - C(\theta(k), \theta(k-1))\frac{\theta(k) - \theta(k-1)}{T_s} \\ - G(\theta(k), \theta(k-1)) - D(\theta(k), \theta(k-1)) - F(\theta(k), \theta(k-1))] \end{aligned} \tag{11}$$

With the feedback signal $e(k) = \theta_d(k) - \theta(k)$, the output of PD controller is

$$y_{pd}(k) = \left(K_P + K_D \frac{z-1}{z}\right) e(k) \tag{12}$$

Thus the computed torque in discrete-time domain can be expressed by

$$\begin{aligned}\tau_0(k) = {} & M(\theta(k)) T_s^{-2} [\theta_d(k+1) - 2\theta_d(k) + \theta_d(k-1) + K_P e(k) \\ & + K_D(e(k) - e(k-1))] + C(\theta(k), \theta(k-1)) \frac{\theta(k) - \theta(k-1)}{T_s} \\ & + G(\theta(k))\end{aligned} \tag{13}$$

$$\tau_c(k) = M(\theta(k)) T_s^{-2} \hat{y}_{rnn}(k) \tag{14}$$

Where $\tau_c(k)$ is the RNN estimation of uncertainties $D(\theta(k), \theta(k-1))$ and $F(\theta(k), \theta(k-1))$. Substitute Eq. 13, 14 into Eq. 11, closed loop error dynamics of robot control system become

$$\begin{aligned}& T_s^2 M^{-1}(\theta(k))[D(\theta(k), \theta(k-1)) + F(\theta(k), \theta(k-1))] - \hat{y}_{rnn}(k) \\ & = e(k+1) - 2e(k) + e(k-1) + K_P e(k) + K_D(e(k) - e(k-1)) \\ & = e(k+1) * [1 + (K_P + K_D - 2) z^{-1} + (1 - K_D) z^{-2}]\end{aligned} \tag{15}$$

In practice, the modelling error $r(k) = d(k) - \hat{y}_{rnn}(k)$ in training algorithm may not be directly measurable. We would utilize the relationship (15) to obtain $r(k)$ through $e(k)$. Define the operator

$$G(z) = 1 + (K_P + K_D - 2) z^{-1} + (1 - K_D) z^{-2} \tag{16}$$

Because $G(z)$ is an FIR filter such that all the poles located in the unit circle in complex z-plane. Hence $G(z)$ is open loop stable and we need only to ensure $r(k) \in L_2$ in order to guarantee BIBO of closed loop control systems. Along this thought, we will study the L_2 stability of RNN training algorithm via conic sector theorem in the next section.

4 Robust Adaptive Training Algorithm

During the training phase of RNN, the weights are updated recursively to make the output best fit into the training data set. The target is to find the optimal weight that minimizes the following cost function

$$f = \frac{r(k)^T r(k)}{2} \tag{17}$$

In an environment of time-varying signal statistics, one of the frequently used method is the random gradient search algorithm that iteratively reduces $f(r(k))$ by estimating the weight vector at each time instant

$$\hat{V}_i(k+1) = \hat{V}_i(k) - \alpha \cdot \frac{\partial f}{\partial \hat{V}_i(k)} \qquad \hat{W}_i(k+1) = \hat{W}_i(k) - \alpha \cdot \frac{\partial f}{\partial \hat{W}_i(k)} \tag{18}$$

where step-size α is the so-called learning rate and subscription i represents the ith row of the matrix. Obviously the hidden layer output is always bounded due to the threshold properties of the activation function. Therefore influence of output layer weights on the whole system stability will be predominant. Hence we concentrate on the analysis of RNN output layer in this paper.

It is well known that in order to achieve a better convergence speed, a larger learning rate α is required, however, big steady state error may be resulted or even a unstable training. On the contrary, small step-size may lead to excessive number of iterations needed to reach the minimum, ie, a slow convergence speed. Thus an optimal learning rate is desirable to achieve a tradeoff between convergence speed and stability. In this section, we present a robustness analysis for the training and derive an online adaptive learning rate based upon the input–output approach from nonlinear system theory. We start by introducing the conic sector theorem. Considering following feedback system

$$\begin{cases} r(k) &= \varepsilon(k) - \phi(k) \\ e_v(k) &= H_1 r(k) \\ \phi(k) &= H_2 e_v(k) \end{cases} \tag{19}$$

Where operators $H_1, H_2 : L_{2e} \to L_{2e}$, and discrete time signal $r(k), e_v(k), \phi(k) \in L_{2e}$ and $\varepsilon(k) \in L_2$.

Theorem 1 (Conic Sector Theorem) *If $H_1 : r(k) \to e_v(k)$ and $H_2 : e_v(k) \to \phi(k)$ satisfy the following two inequalities for some σ, γ, η*

$$\text{(a)} \quad \sum_{k=1}^{N} [r(k)e_v(k) + \tfrac{1}{2}\sigma r^2(k)] \geq -\gamma$$

$$\text{(b)} \quad \sum_{k=1}^{N} [\tfrac{1}{2}\sigma \phi^2(k) - \phi(k)e_v(k)] \leq -\eta \, \|\phi(k), e_v(k)\|_N^2$$

Then the closed loop system is stable in sense of $r(k), e_v(k) \in L_2$.

Proof See corollary 8.1 in [19]. □

Remark 1 In Fig. 4, operator H_1 represents the nonlinear mapping and H_2 is dynamic linear operator. Indeed when condition (a) and (b) are satisfied, H_1 will be dissipative and H_2^{-1} will be strictly inside the cone $(1, (1-\sigma_{LS})^{1/2})$, which is equivalent to H_2 being strictly inside the cone $(\sigma_{LS}^{-1}, \sigma_{LS}^{-1}(1-\sigma_{LS})^{1/2})$ [19]. For the

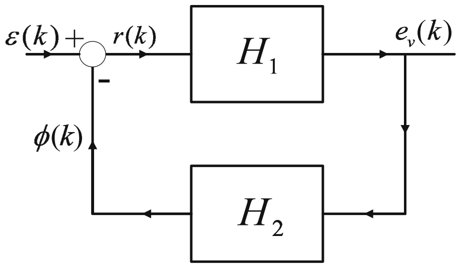

Fig. 4 Closed loop feedback system: H_1–static nonlinear, H_2–dynamic linear

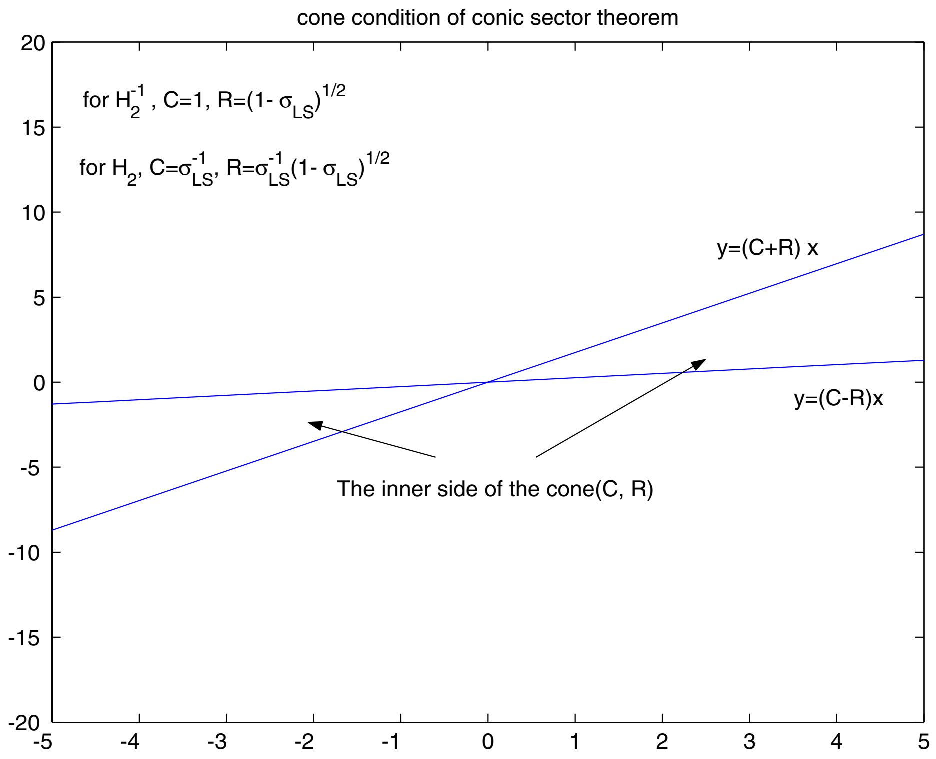

Fig. 5 Cone conditions for H_2^{-1} and H_2 to be passive

case that no dynamics are present, eg, $H_2 = 1$, then H_2 will be strictly inside any cone as long as $\sigma_{LS} < 1$ holds. This relation is illustrated in Fig. 5.

The first step of training RNN is calculating the error gradient. However, because $r(k)$ is a vector of 6×1-dimensional and $\hat{V}(k)$ is a matrix of 6×50-dimensional in this work, there is no convenient way to directly calculate the derivative of $r(k)$ against $\hat{V}(k)$. We can only derive it row by row. Define the Jocabian matrix

$$J_i(k) = \frac{\partial x(k)}{\partial \hat{V}_i(k)} \in R^{24\times 50} \tag{20}$$

Where $\hat{V}_i(k)$ is the ith row of $\hat{V}(k)$ and $i = 1, \cdots, 6$. Subsequently the error gradient can be obtained as

$$\begin{aligned}\frac{\partial f(r)}{\partial \hat{V}_i(k)} &= -\sum_{n=1}^{6} r_n(k)\left[\frac{\partial \hat{y}_{rnn,n}(k)}{\partial \hat{V}_i(k)} + \frac{\partial \hat{y}_{rnn,n}(k)}{\partial x(k)} \cdot \frac{\partial x(k)}{\partial \hat{V}_i(k)}\right] \\ &= -\sum_{n=1}^{6} r_n(k)\frac{\partial \hat{y}_{rnn,n}(k)}{\partial \hat{V}_i(k)} - \sum_{n=1}^{p} r_n(k)\hat{V}_n(k) diag\{H'(\cdot)\}\hat{W}(k)J_i(k) \\ &= -\, r_i(k)H(\hat{W}(k)x(k))^T - r^T(k)\hat{V}(k) diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\end{aligned} \tag{21}$$

Theorem 2 *If RNN is trained by the following normalized gradient based algorithm*

$$\hat{V}_i(k+1) = \hat{V}_i(k) - \beta\alpha(k)\frac{\partial f(r(k))}{\partial \hat{V}_i(k)} \tag{22}$$

where β is a scaling factor in the interval $(0, 1)$, $\frac{\partial f(r(k))}{\partial \hat{V}_i(k)}$ is defined in Eq. 21, and $\alpha(k)$ is adaptive learning rate determined by

$$\alpha(k) = \frac{2\left\{1 + \left\|\hat{V}(k) diag\{H'(\cdot)\}\hat{W}(k)\sum_{n=1}^{6} J_n(k)\right\|^2 \sum_{n=1}^{6}\left[\hat{V}_n(k) diag\{H'(\cdot)\}\hat{W}(k)J_n(k)H(\hat{W}(k)x(k))\right]^{-1}\right\}}{\left\|H(\hat{W}(k)x(k))\right\|^2 + \sum_{i=1}^{6}\left\|\hat{V}(k) diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\right\|^2} \tag{23}$$

Then the training will be L_2-stable in the sense of $e, e_v \in L_2$

Proof Before applying the conic sector stability theorem, we need to restructure the adaptation law (22) into an equivalent error feedback system. Moreover the weight estimation error should be referred as output signal of the closed loop.

(1) Construct dynamic linear operator H_2

Firstly, the $r(k)$ is decomposed as follows

$$\begin{aligned} r(k) &= d(k) - \hat{y}_{rnn}(k) \\ &= V^*H(W^*x(k)) - \hat{V}(k)H(\hat{W}(k)x(k)) \\ &= [V^*H(W^*x(k)) - V^*H(\hat{W}(k)x(k))] - [\hat{V}(k)H(\hat{W}(k)x(k)) - V^*H(\hat{W}(k)x(k))] \end{aligned} \tag{24}$$

Where V^* and W^* are ideal values of $\hat{V}(k)$ and $\hat{W}(k)$ respectively. Due to the threshold activation function, $V^*H(W^*x(k)) - V^*H(\hat{W}(k)x(k)$ will always be bounded and has no influence on output layer stability. Thus we can put it into disturbance term $\tilde{\varepsilon}(k)$. On the other hand, the $\hat{V}(k)H(\hat{W}(k)x(k)) - V^*H(\hat{W}(k)x(k))$ can be regarded as an independent term, which is related to the weight estimation error in a explicit manner. Define

$$e_v(k) = \hat{V}(k)H(\hat{W}(k)x(k)) - V^*(k)H(\hat{W}(k)x(k)) = \tilde{V}(k)H(\hat{W}(k)x(k)) \tag{25}$$

Then Eq. 24 can be simplified to

$$r(k) = \tilde{\varepsilon}(k) - e_v(k) \tag{26}$$

Above equation establishes the relationship between the disturbance $\tilde{\varepsilon}(k)$ and the posterior estimation error $e_v(k)$, where the linear feedback gain $H_2 = 1$.

(2) Construct static nonlinear operator H_1

We proceed to establish the nonlinear forward path which maps output error $r(k)$ into parameter estimation error $e_v(k)$. Substitute Eq. 21 into Eq. 22, then training algorithm of RNN can be expanded as

$$\tilde{V}_i(k+1) = \tilde{V}_i(k) + \beta \cdot \alpha(k)[r_i(k)H(\hat{W}(k)x(k))^T + r^T(k)\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)] \tag{27}$$

Note the above equation is actually an modified least mean square algorithm, which uses a filtered tracking error $r(k)$ instead of $e(k)$ in adaptation. Square both sides of Eq. 27 and rearrange the terms

$$\begin{aligned}\left\|\tilde{V}_i(k+1)\right\|^2 - \left\|\tilde{V}_i(k)\right\|^2 = {} & 2\beta\alpha(k)\tilde{V}_i(k)[r_i(k)H(\hat{W}(k)x(k))^T \\ & + r^T(k)\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)]^T \\ & + \beta^2\alpha^2(k)\left\|r_i(k)H(\hat{W}(k)x(k))^T\right. \\ & \left. + r^T(k)\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)]\ \right\|^2\end{aligned} \tag{28}$$

Because of the fact that $2\alpha(k)r(k)^T\tilde{V}(k)H(\hat{W}(k)x(k)) = 2\alpha(k)r(k)^Te_v(k)$, summing up Eq. 28 of index i, the following equation can be obtained

$$\begin{aligned}\sum_{i=1}^{6}\left\|\tilde{V}_i(k+1)\right\|^2 - \sum_{i=1}^{6}\left\|\tilde{V}_i(k)\right\|^2 = {} & 2\beta\alpha(k)r(k)^Te_v(k) \\ & \times\left[1+\frac{\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)\sum_{n=1}^{6}J_n(k)\right\|^2}{\sum_{n=1}^{6}[\hat{V}_n(k)diag\{H'(\cdot)\}\hat{W}(k)J_n(k)H(\hat{W}(k)x(k))]}\right] \\ & + \beta^2\alpha^2(k)\sum_{i=1}^{6}\|r_i(k)\ H(\hat{W}(k)x(k))^T \\ & + r^T(k)\hat{V}(k)diag\{H'(\cdot)\}\ \hat{W}(k)J_i(k)\Big\|^2\end{aligned} \tag{29}$$

Furthermore, we have

$$\begin{aligned}&\sum_{i=1}^{6}\left\|r_i(k)H(\hat{W}(k)x(k))^T + r^T(k)\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\right\|^2 \\ &\leq 2\sum_{i=1}^{6}r_i^2(k)\left\|H(\hat{W}(k)x(k))\right\|^2 + 2\|r(k)\|^2\sum_{i=1}^{6}\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)]\right\|^2 \\ &= 2\|r(k)\|^2\left[\left\|H(\hat{W}(k)x(k))\right\|^2 + \sum_{i=1}^{6}\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)]\right\|^2\right]\end{aligned} \tag{30}$$

Substitute Eq. 30 into Eq. 29

$$\begin{aligned}\sum_{i=1}^{6}\left\|\tilde{V}_i(k+1)\right\|^2-\sum_{i=1}^{6}\left\|\tilde{V}_i(k)\right\|^2 &\le 2\beta\alpha(k)r(k)^T e_v(k)\\ &\times\left[1+\frac{\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)\sum_{n=1}^{6}J_n(k)\right\|^2}{\sum_{n=1}^{6}[\hat{V}_n(k)diag\{H'(\cdot)\}\hat{W}(k)J_n(k)H(\hat{W}(k)x(k))]}\right]\\ &+2\beta^2\alpha^2(k)\left\|r(k)\right\|^2\left[\left\|H(\hat{W}(k)x(k))\right\|^2+\sum_{i=1}^{6}\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\right\|^2\right]\end{aligned} \tag{31}$$

Summing up inequality (31) of sampling index k

$$\begin{aligned}\sum_{i=1}^{6}\left\|\tilde{V}_i(N+1)\right\|^2-\sum_{i=1}^{6}\left\|\tilde{V}_i(0)\right\|^2 &\le \sum_{k=0}^{N}2\beta\alpha(k)e_v(k)r(k)\\ &\times\left[1+\frac{\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)\sum_{n=1}^{6}J_n(k)\right\|^2}{\sum_{n=1}^{6}[\hat{V}_n(k)diag\{H'(\cdot)\}\hat{W}(k)J_n(k)H(\hat{W}(k)x(k))]}\right]\\ &+\sum_{k=0}^{N}\left\{2\beta^2\alpha^2(k)\left\|r(k)\right\|^2\left[\left\|H(\hat{W}(k)x(k))\right\|^2+\sum_{i=1}^{6}\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\right\|^2\right]\right\}\end{aligned} \tag{32}$$

By this step, we are able to form the closed loop through Eq. 26 and Eq. 32. Thus conic sector theorem can be applied straight forwardly.

(3) Sufficient conditions of L_2-stability

To simply the presentation, we define the following notation

$$\rho=2\beta\alpha^2(k)\left[\left\|H(\hat{W}(k)x(k))\right\|^2+\sum_{i=1}^{6}\left\|\hat{V}(k)diag\{H'(\cdot)\}\hat{W}(k)J_i(k)\right\|^2\right] \tag{33}$$

Substitute $\alpha(k)$ of Eq. 23 into Eq. 32, the following inequality can be derived

$$\begin{aligned}&\sum_{k=0}^{N}\left[e_v(k)r(k)+\frac{1}{2}\left\|r(k)\right\|^2\right]\\ &\quad\ge\sum_{k=0}^{N}\left[e_v(k)r(k)+\frac{1}{2}\beta\left\|r(k)\right\|^2\right]\\ &\quad\ge\rho_{min}\sum_{i=1}^{6}\left\|\tilde{V}_i(N+1)\right\|^2-\rho_{max}\sum_{i=1}^{6}\left\|\tilde{V}_i(0)\right\|^2\end{aligned} \tag{34}$$

Table 1 Parameters values of the robot model

	Length (m)	Mass (kg)	Mass center (m)	Inertia (kg*m^2)
Coxa(link 3)	0.204	5.900	0.070	0.010
Thigh(link 2,4)	0.412	13.20	0.210	0.067
Calf(link 1,5)	0.385	7.700	0.223	0.010
Foot(link 6,7)	0.290	8.200	0.140	0.028

Now by the cone condition (a)(b) of Theorem 1, we conclude the proof. Note the condition (b) of theorem 1 can be treated as positive real function, which $H_2 = 1$ already satisfies. □

Remark 2 The algorithm can be explained that at the starting point, the neural network is trained by the gradient algorithm. With the time step moves on, the estimation error tends to drift outside the specified cone, ie, the estimation error energy is amplified. Then we utilize the normalized learning rate to push the weight estimation error within the bounds of the cone, which is equivalent to make the feedback system satisfy the conic stability. As for β in theorem 2, the theoretical ideal value is 1. In the practical design, we suggest to make $\beta = 0.8$ to maintain both fast transient response while marginal stability. The entire procedure to synthesize the robust adaptive training algorithm for RNN can be summarized as follows:

Step 1: Calculate estimation error $r(k)$ of RNN by the measurements of $e(k)$ via Eq. 15;

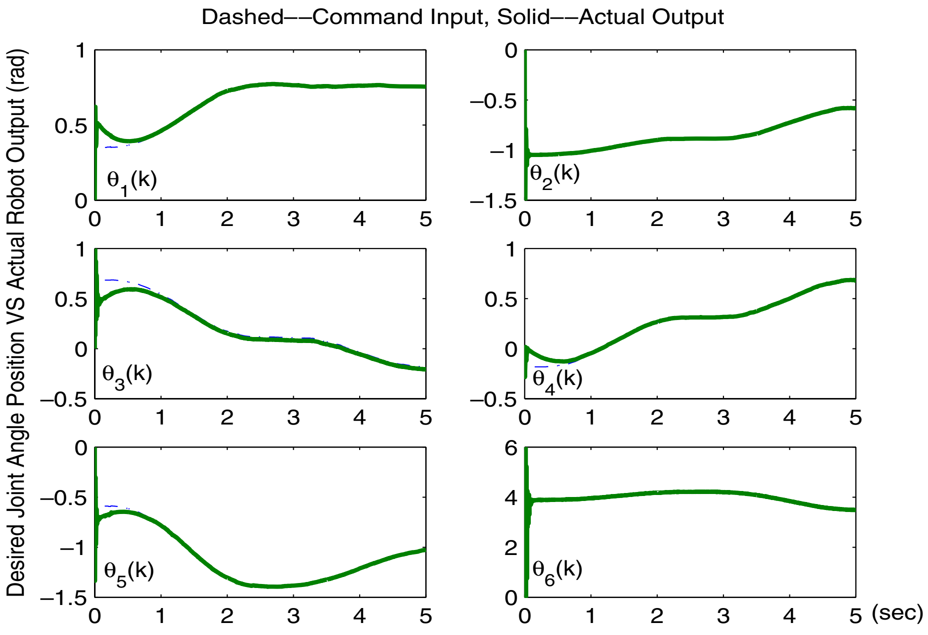

Fig. 6 Actual and desired joint angle positions in case of no fault

Step 2: Calculate the gradient of $f(r(k))$ with respect to weight parameter $\hat{V}(k)$, based upon Eq. 21;
Step 3: Choose the optimal learning rate $\alpha(k)$ according to Eq. 23 and update the weight parameters of each layer of RNN;
Step 4: Calculate the computed torque $\tau_0(k)$ and RNN compensation torque $\tau_c(k)$ according to the trained weight;
Step 5: With the torque input, we measure the angular position $\theta(k+1)$ of robot joint angle, and compare with command signal $\theta_d(k+1)$ to obtain $e(k+1)$;
Step 6: Go back to Step 1 to continue iteration.

5 Simulations

In this section, the proposed robust adaptive training algorithm and the hybrid controller are investigated through computer simulations. The control objective is to make joint positions of the biped robot follow the reference trajectories. Moreover, when fault occurs, we expect to recover the control performance by RNN compensation scheme. The RNN is constructed with 50 hidden neurons and 24 input nodes. Both of the hidden and output layer weights of RNN are initialized by uniformly distributed number between -1 and 1. Sigmoid function $H(x) = 1/(1+e^{-\lambda x})$ is chosen as activation function. The sampling period $T = 0.005$ s. Every simulation is running $1,000$ steps, ie, 5 s. In the model setup of simulations, the lower limb of the robot is divided into two identical parts: left and right, including coxa, thigh, calf, and foot. Formulas of various parameters of robot dynamics are given in Appendix. The nominal values of the coefficients in these formulas are presented in Table 1.

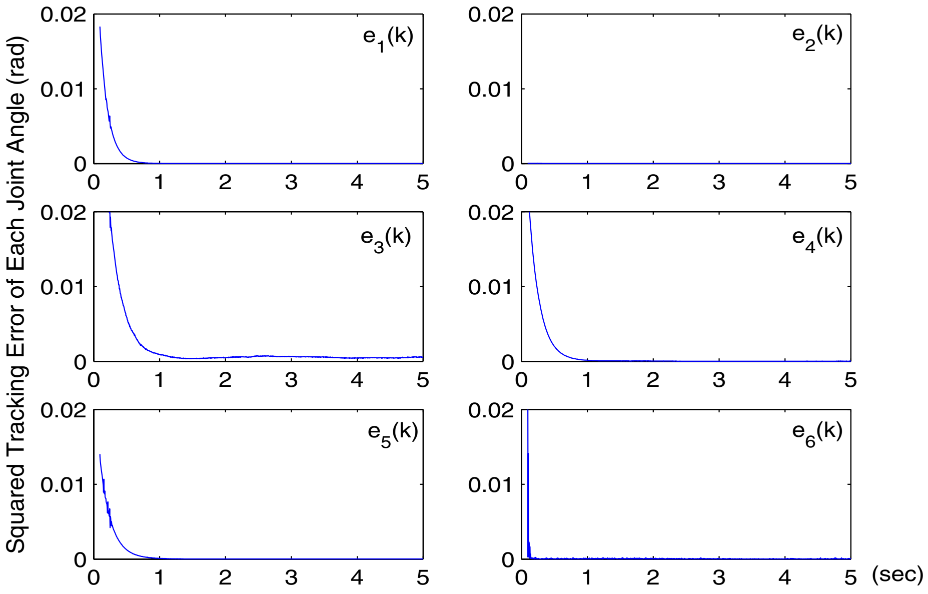

Fig. 7 Squared tracking error in case of no fault

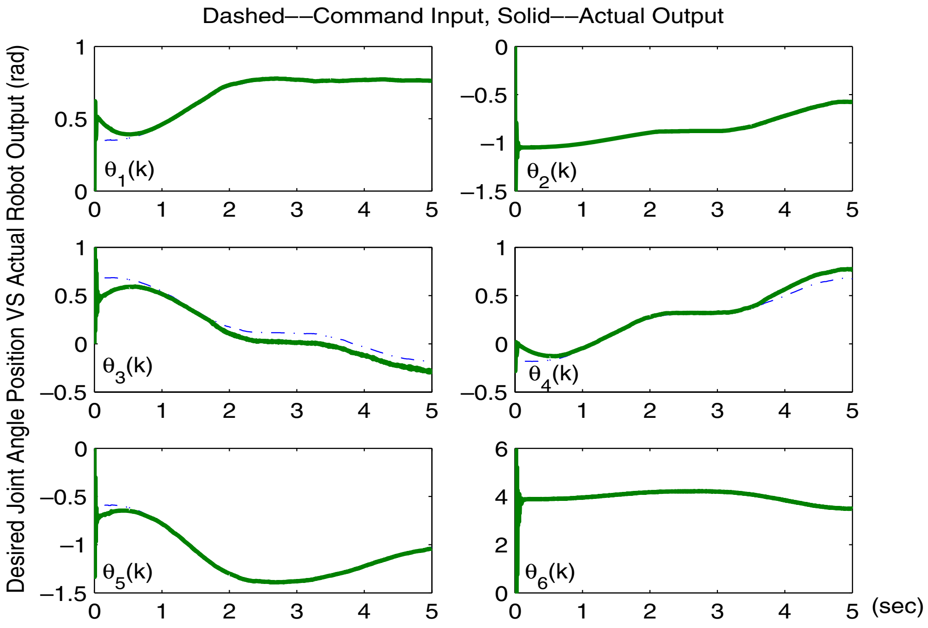

Fig. 8 Actual and desired joint angle positions in case of fault occurring

The desired trajectories θ_d of joint angles are chosen to use the effects of gravity in a way that the angular momentum is increased in the single support phase. Two

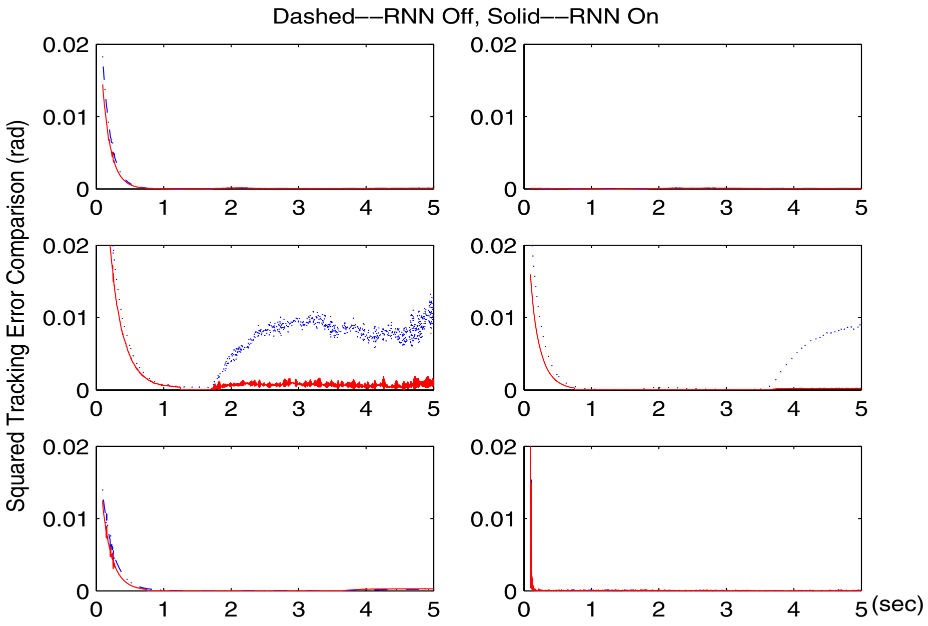

Fig. 9 Squared tracking error in case of fault occurring

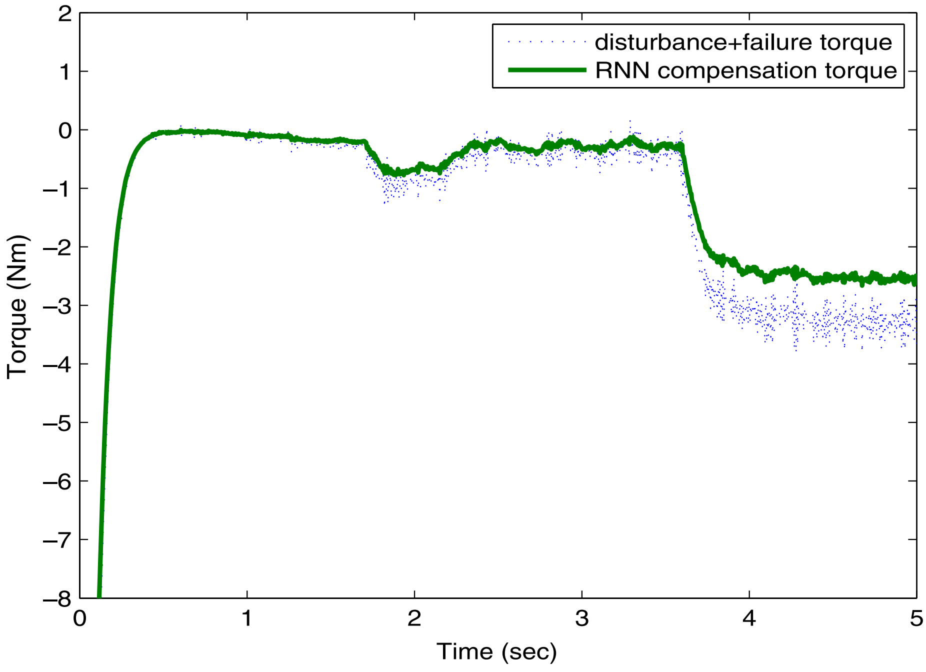

Fig. 10 Compensation output and estimation error of RNN

simulation examples are synthesized to evaluate the robustness of the controller. In both cases, the initial joint positions and their velocities are given by

$$\begin{aligned}\theta(0) &= [\theta_1(0) \quad \theta_2(0) \quad \theta_3(0) \quad \theta_4(0) \quad \theta_5(0) \quad \theta_6(0)]^T \\ &= [0.37 \quad -1 \quad 0.75 \quad -0.15 \quad 0.56 \quad 3.85]^T rad \\ \dot{\theta}(0) &= [\dot{\theta}_1(0) \quad \dot{\theta}_2(0) \quad \dot{\theta}_3(0) \quad \dot{\theta}_4(0) \quad \dot{\theta}_5(0) \quad \dot{\theta}_6(0)]^T \\ &= [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T rad/\sec\end{aligned}$$

Case 1 Only external disturbances, no fault

In the first simulation, only system uncertainty is considered. We employ PD controller and turn off the RNN compensator. The disturbances, including noise, static and dynamic frictions are described as

$$\begin{cases} D(\theta, \dot{\theta}) &= 0.5sign(\dot{\theta}) + 2\dot{\theta} \\ F(\theta, \dot{\theta}, t) &= 0 \end{cases} \tag{35}$$

The plant output and squared tracking error are displayed in Figs. 6 and 7. The simulation results indicate that in a fault-free environment, PD controller is capable to provide a satisfactory tracking performance and no necessary to insert any compensator.

Case 2 Two faults, occurs at 1.7th and 3.6th s, respectively.

In the second simulation, we take into account the model uncertainty as well as system uncertainty. The first fault of 50% increase in the mass of link 2 and 4 occurs at 1.7th s. The second fault of nonlinearity changes in link 4 and 5 occurs at 3.6 s, where the failure function is expressed as

$$\begin{cases} D(\theta,\dot{\theta}) &= 0.5sign(\dot{\theta}) + 2\dot{\theta} \\ F(\theta,\dot{\theta},t) &= M(\theta)[0, \quad 0, \quad 0, \quad 0.6\theta_4^2\dot{\theta}_5^2, \quad 0.5\dot{\theta}_4\theta_5, \quad 0]^T, \qquad t = 3.6 \end{cases} \tag{36}$$

To provide a comparative idea, firstly we turn off RNN compensator and try to use PD controller only. The simulation results are shown in Fig. 8. It can be seen that the joint angles cannot catch up with the command signal in link 3 and 4 due to the model uncertainty.

Then we turn on the RNN compensator. The tracking errors of the two control scheme are put together in Fig. 9. In addition, the failure function and RNN compensation effort are displayed in Fig. 10. To avoid over lengthy plot, only the information of link 4 are presented because of its representative characteristics in both faults. From the plots, we found that before faults occur, the steady state error of hybrid controller is almost the same as that of pure PD controller setup. This means both the controllers can stabilize the biped robot dynamics in a fault-free condition. In contrast, after the occurrence of nonlinear faults and modelling uncertainties, the trajectory error of PD control method deteriorates considerably, especially in link 3 and 4. While the control performance is apparently enhanced in the configuration of the hybrid controller with RNN compensator.

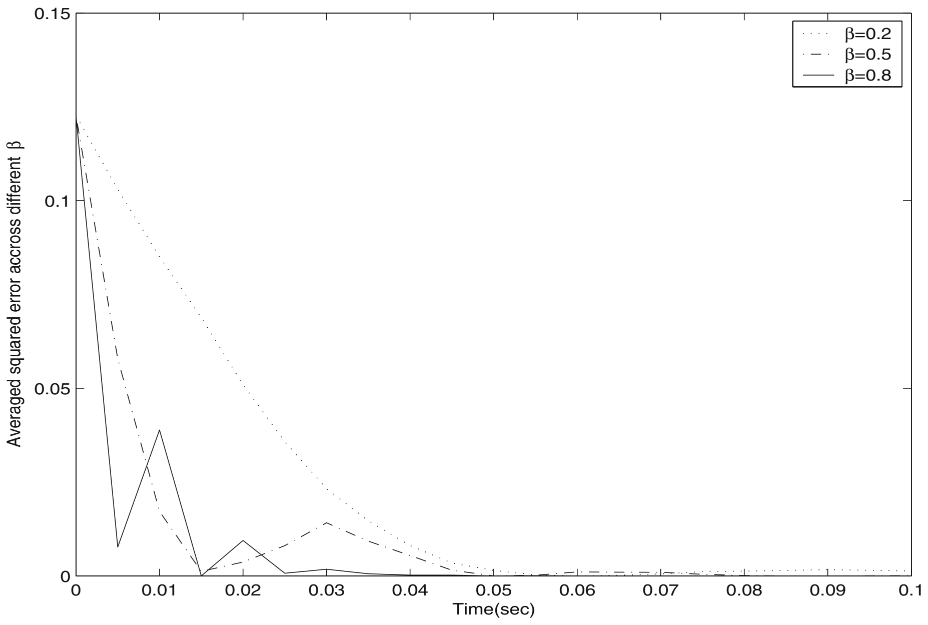

Fig. 11 Tracking error of RNN compensation with different scaling factor β

The effects of different choices of β on the derived training algorithm (22) are also studied. It turns out the convergence speed is getting faster as β increases, as expected. On the other hand, the jitter in the training error becomes series when β approaches 1, which may lead to unstable training. We suggest to make $\beta = 0.8$ in design to ensure both transient performance and marginal stability. The simulation result is displayed in the next figure. The first 20 steps (0.1 s) of averaged MSE of 6 joint angle tracking results is shown (see Fig. 11).

6 Conclusion

In this work, we present a RNN compensation scheme for robot trajectory tracking system. The convergence speed of gradient-type training algorithm is optimized by using adaptive learning rate. Robustness is analyzed based upon conic sector theorem. Simulation results show that standard PD controller most likely suffers from poor transient response when system faults occur. While with the assistance of RNN, the performance is prevented from deteriorating successfully. Therefore we conclude that the proposed RNN and training algorithm can provide a favorable compensation of nonlinear faults and lead to an improvement of whole system robustness.

Acknowledgements We thank the two anonymous referees who helped to improve the article greatly.

Appendix

The parameters M, C, G of robot dynamics in Eq. 11 can be calculated by

$$M(\theta(k)) = \{c_{ij}\cos(\theta_i(k) - \theta_j(k))\}$$
$$C(\theta(k)) = \{c_{ij}\sin(\theta_i(k) - \theta_j(k))\}$$
$$G(\theta(k)) = \{-h_i \sin\theta_i(k)\}$$

Where

$$h_1 = (m_1a_1 + m_2l_1 + m_3l_1 + m_4l_1 + m_5l_1 + m_6l_1)g \qquad h_2 = (m_2a_2 + m_3l_2 + m_4l_2 + m_5l_2 + m_6l_2)g$$
$$h_3 = m_3a_3g \qquad h_4 = (m_4a_4 - m_4l_4 - m_5l_4 - m_6l_4)g$$
$$h_5 = (m_5a_5 - m_5l_5 - m_6l_5)g \qquad h_6 = -m_6bg$$
$$c_{11} = m_1a_1^2 + (m_2 + m_3 + m_4 + m_5 + m_6)l_1^2 + I_1 \qquad c_{22} = m_2a_2^2 + (m_3 + m_4 + m_5 + m_6)l_2^2 + I_2$$
$$c_{33} = m_3a_3^2 + I_3 \qquad c_{44} = m_4(l_4 - a_4)^2 + (m_5 + m_6)a_4^2 + I_4$$
$$c_{55} = m_5(l_5 - a_5)^2 + m_6l_5^2 + I_5 \qquad c_{66} = m_6b^2 + I_5$$
$$c_{12} = m_2l_1a_2 + (m_3 + m_4 + m_5 + m_6)l_1l_2 \qquad c_{13} = m_3l_1a_3$$
$$c_{14} = -m_4l_1(l_4 - a_4) - (m_5 + m_6)l_1l_4 \qquad c_{15} = -m_5l_1(l_5 - a_5) - m_6l_1l_5$$

$$c_{16} = -m_6 l_1 b \qquad c_{23} = m_3 l_2 a_3$$
$$c_{24} = -m_4 l_2 (l_4 - a_4) - (m_5 + m_6) l_2 l_4 \qquad c_{25} = -m_5 l_2 (l_5 - a_5) - m_6 l_2 l_5$$
$$c_{26} = -m_6 l_2 b \qquad c_{34} = c_{35} = c_{36} = 0$$
$$c_{45} = m_5 l_4 (l_5 - a_5) + m_6 l_4 l_5$$
$$c_{46} = m_6 l_5 b$$
$$c_{ij} = c_{ji}$$

References

1. Yildirim, S.: Design of adaptive robot control system using recurrent neural network. J. Intell. Robot. Syst. **44**(3), 247–261 (2005)
2. Loo, C.K., Mandava, R., Rao, M.V.C.: A hybrid intelligent active force controller for articulated robot arms using dynamic structure neural network. J. Intell. Robot. Syst. **40**(2), 113–145 (2004)
3. Song, Q., Hu, W.J., Soh, Y.C.: Robust adaptive dead zone technology for fault-tolerant control of robot manipulators using neural networks. J. Intell. Robot. Syst. **33**(2), 113–137 (2002)
4. Blanke, M.: Fault-tolerant control systems – a history view. Control Eng. Pract. **5**(5), 693–702 (1997)
5. Lewis, F.L., Liu, K., Yesildirek, A.: Neural net robot controller with guaranteed tracking performance. IEEE Trans. Neural Netw. **6**(3), 703–715 (1995)
6. Gu, D.B., Hu, H.S.: Neural predictive control for a car-like mobile robot. Robot. Auton. Syst. **39**(2), 73–86 (2002)
7. Liu, Z., Li, C.W.: Fuzzy neural network quadratic stabilization output feedback control for biped robots via H infinity approach. IEEE Trans. Syst. Man Cybern., Part B, Cybern. **33**(1), 67–84 (2003)
8. Rupp, M., Sayed, A.H.: Supervised learning of perceptron and output feedback dynamic networks: a feedback analysis via the small gain theorem. IEEE Trans. Neural Netw. **8**(3), 612–622 (1997)
9. Mak, M.W., Ku, K.W., Lu, Y.L.: On the improvement of the real time recurrent learning algorithm for recurrent neural networks. Neurocomputing **24**(1), 13–36 (1999)
10. Haykin, S.: Neural Networks. Prentice-Hall, Upper saddle River, NJ (1999)
11. Atiya, A.F., Parlos, A.G.: New results on recurrent network training: unifying the algorithms and accelerating convergence. IEEE Trans. Neural Netw. **11**(3), 697–709 (2000)
12. Mandic, D.P., Chambers, J.A.: Recurrent Neural Networks for Prediction: Learning Algorithms, Architecture and Stability. Wiley, Singapore (2001)
13. Sun, F.C., Sun, Z.Q., Woo, P.Y.: Neural network-based adaptive controller design of robotic manipulators with an observer. IEEE Trans. Neural Netw. **12**(1), 54–67 (2001)
14. Song, Q., Xiao, J., Soh, Y.C.: Robust back-propagation training algorithm for multi-layered neural tracking controller. IEEE Trans. Neural Netw. **10**(5), 1133–1141 (1999)
15. Lin, C.M., Chen, C.H., Hus, C.F., Fan, W.Z.: Robust fault-tolerant control for robotic system using recurrent cerebellar model articulation controller. IEEE Int. Conf. Industrial Technology **7**, 1006–1011 (2005)
16. Song, Q., Grimble, M.J.: Robust neural network/proportional tracking controller with guaranteed global stability. Proc. Int. Symp. Intelligent Control **15**, 34–39 (2003)
17. Lewis, F.L.: Control of Robot Manipulators. Macmillan, New York (1993)
18. Tzafestas, S.G., Krikochoritis, A.E., Tzafestas, C.S.: A robust-adaptive locomotion controller for 9-link with rapidly varying unkown parameters. Proc. Mediterranean Conf. Control and Systems **5**, 21–23 (1997)
19. Cluett, W.R., Shah, L., Fisher, D.G.: Robustness analysis of discrete-time adaptive control systems using input-output stability theory: a tutorial. IEE Proc. Part D. **135**(2), 133–141 (1988)