01 Jan 2001

Adaptive Critic Based Neuro-Observer

Xin Liu
Missouri University of Science and Technology, xinliu@mst.edu

S. N. Balakrishnan
Missouri University of Science and Technology, bala@mst.edu

Recommended Citation

X. Liu and S. N. Balakrishnan, "Adaptive Critic Based Neuro-Observer," *Proceedings of the 2001 American Control Conference, 2001*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2001.
The definitive version is available at https://doi.org/10.1109/ACC.2001.945958

Proceedings of the American Control Conference
Arlington, VA June 25-27, 2001

Paper ID: ACC01-IEEE1297

Adaptive Critic Based Neuro-Observer

Xin Liu[1], S. N. Balakrishnan[2]
Department of Mechanical and Aerospace Engineering and Engineering Mechanics
University of Missouri-Rolla
Rolla, MO 65409-0050
email: bala@umr.edu, xliu@umr.edu

Abstract

A new Neural Network (NN) based observer design method for nonlinear systems represented by nonlinear dynamics and linear/nonlinear measurement is proposed in this paper. In this new approach, as the first step, the observer design problem is changed into a 'controller' design problem by establishing the error dynamics, and then the Adaptive Critic (AC) based approach is applied on this error dynamics to design a 'controller', such that the errors are driven to zero. The resulting observer has inherent robustness from the AC based design approach. Some simulations are presented to illustrate the effectiveness of this approach.

1. Introduction:

State estimation is a critical issue in control and has been used widely in real-time applications, such as [1]. Although the solutions for linear systems can be obtained successfully by using Kalman filter [2] and Luenberger observer [3], the solutions for more important and complicated nonlinear problems are still under investigation. In [4], a stable adaptive observer for Single Input Single Output (SISO) observable nonlinear time varying system is introduced, which is based on the transformation of the system to an observable canonical form. A nonlinear observer design method is proposed by Yaolong [5] for a class of nonlinear systems. Those nonlinear systems are affine functions of the derivative of the measured output, with coefficients that are smooth nonlinear functions of the measured output. Krishna, K. Busawon[6] proposed a constant gain nonlinear observer design method based on linearization. All these efforts attempted to find the solutions for some class of nonlinear systems and as such the resulting design approaches are limited to be used on certain class of nonlinear system or not accurate when the states are far from the linearization point.

In order to deal with control systems with complex and unknown nonlinear dynamics, NNs are introduced to approximate a part or the whole nonlinear dynamics [7-14], which is based on the premise that NNs have the capability of approximating any nonlinear continuous function to arbitrary accuracy. Thus, the observer design problem is transformed to how to adjust the weights of NNs to find the desired solution. In [7], a multi-layer Recurrent Neural Network (RNN) is used to determine the output gain matrix of a Luenberger state observer in real time. General Regression Neural Networks (GRNN) are used in [8] and [9] to design observers for a class of nonlinear systems with known linear part and single unknown nonlinear part. In [10], C. L. Hwang proposes an estimator design method for a class of nonlinear dynamic systems based on Radial Basis Function Networks (RBFN). A B-Spline NN is used in [11], in which the nonlinear observer consists of a linear observer and a nonlinear compensation term. The compensation term is determined first and then a B-Spline NN is used to model the compensation term. Some of the attempts to address general class of nonlinear systems were presented in [13] and [14]. State estimators are designed using Dynamic Recurrent Neural Networks (DRNN) for a general class of single output nonlinear system. However, the nonlinear dynamics should contain a known linear part, and the Strictly Positive Real (SPR) requirement is very hard to satisfy sometimes. Of all those observer design methods, NNs are used to approximate the unknown dynamics and little work has been done to use the NNs to approximate the observer gains. In this paper, a new observer design method is presented to address a more general class of nonlinear systems, and Multi-Layer Perceptrons (MLPs) are used to approximate the observer gain in the designed observers.

2. Background: Adaptive Critic Based Optimal Control (ACBOC)

Optimal controller designs seek to accomplish some desired objectives by minimizing a pre-defined cost functional, and simultaneously, satisfying some boundary conditions and constraints. The cost functional or performance index is expressed by a mathematical expression in terms of the system variables and controls. For the problems discussed in this paper, the cost functional is chosen to be in a quadratic form which is used in most applications:

$$I = \sum_{k=0}^{\infty} (x_k^T Q x_k + u_k^T R u_k) \qquad (1)$$

where x is the state vector, u represents the control vector and k is the time index. In Eq. (1), Q is a symmetric positive semi-definite matrix and R is a symmetric positive definite matrix. The choice of these matrices is a design decision to give different degrees of importance to the state trajectory and the control effort. The optimal control problem can now be stated as - minimize the cost function I given in Eq. (1) with the differential constraints of the state equation:

$$x_{k+1} = f(x_k, u_k) \qquad (2)$$

when the initial state x_0 is given and final time $t_f \to \infty$. In our study this problem is solved through approximate dynamic programming formulation.

2.1 Approximate Dynamic Programming Method

Dynamic programming provides a computational technique to apply the principle of optimality to sequence of decisions which define an optimal control policy. A general mathematical description of the optimality conditions obtained as a direct convergence of the "Principle of Optimality" is the Hamilton-Jacobi Bellman (HJB) equation [16]. The HJB equation for a discrete-time system is given below:

$$J(x_k) = \min_{u(x_k)}\{U(x_k, u(x_k)) + < J(f(x_k, u(x_k)))>\} \qquad (3$$

where state at time step k is given by x_k and the control by $u(x_k$. $J(x_k)$ represents the cost-to-go from time instant k to final time N $U(x_k, u(x_k))$ is the utility function denoting the one step cost an $<J(x_{k+1})>$ is the optimized cost-to-go from time instant $k+1$ final time N. For the adaptive critic method used in this study, *co-state* (or Lagrangian multiplier) $\lambda(x_k)$ is defined

$\lambda(x_k) = \frac{\partial J(x_k)}{\partial x_k}$, also $\lambda(x_{k+1}) = \frac{\partial J(x_{k+1})}{\partial x_{k+1}}$ and the *co-st*

equation can be derived by differentiating Eq. (3) with respect to x_k:

$$\lambda(x_k) = \frac{\partial U(x_k,u_k)}{\partial x_k} + \frac{\partial J(f(x_k,u(x_k)))}{\partial x_k}$$
$$= \frac{\partial U(x_k,u_k)}{\partial x_k} + \left(\frac{\partial u(x_k)}{\partial x_k}\right)^T \frac{\partial U(x_k,u_k)}{\partial u_k} \quad (4)$$
$$+ \left(\frac{\partial x_{k+1}}{\partial x_k} + \frac{\partial x_{k+1}}{\partial u_k}\frac{\partial u(x_k)}{\partial x_k}\right)^T \lambda(x_{k+1})$$

It can be seen that the co-state equation develops backwards in time. The Bellman's optimality equation can be derived by differentiating Eq. (3) with respect to u_k:

$$0 = \frac{\partial J(x_k)}{\partial u_k} = \frac{\partial U(x_k,u_k)}{\partial u_k} + \frac{\partial J(x_{k+1})}{\partial u_k} \quad (5)$$

Dynamic programming requires the system model and its derivatives in (4) and (5). These equations are used iteratively to solve for a optimal control policy.

2.2 Adaptive Critic and General Training Procedure

Adaptive critic methods have been proposed by Werbos [17] as a new optimization tool combining together concepts of reinforcement learning and approximate dynamic programming. The adaptive critic method used in this paper consists of two neural networks: one outputs the control u_k and the other outputs Lagrangian multiplier λ_k. Inputs to both are the states x_k at time instant k. The adaptive critic technique finds the control which minimizes the cost in Eq. (1) by solving Eq. (2) and Eq. (4) with the use of the stationary Eq.(5) and the known initial states.

The training procedure consists of two training cycles: *critic's* and *action's*. The action neural network outputs the control $u(x_k)$ for the input state x_k. The output of the plant serves as input to the critic neural network that is trained to estimate the derivative of the cost-to-go J. Thus the critic neural network contains information about the function to be minimized. The optimal control can be obtained by training action neural network and critic neural network successively. MLPs are known for their ability to approximate any nonlinear system to any degree of accuracy, and can also adapt to new sets of input-output pairs. This makes them ideal in adapting to an optimal control policy by adjusting the weights using a standard gradient descent algorithm. Two MLPs are chosen to act as the action and critic networks. The training of both of the neural networks is illustrated in Fig. 1, where state x_k is chosen randomly in both training and the weights of both neural networks are initialized randomly. For the optimal controller design problem, the iterative procedure between the training of the action NN and critic NN is continued till the *optimal controller $u^*(k)$ is obtained.*

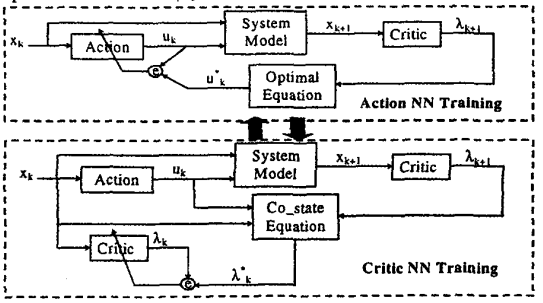

Fig. 1 General training procedure

3 Adaptive Critic Based Neuro-Observer Design

The nonlinear system in our study is given in the following discrete form:

$$x_{k+1} = f(x_k,u_k) \quad (6)$$
$$y_k = h(x_k,u_k) \quad (7)$$

Assume:

1. Dynamics of the nonlinear system $x_{k+1} = f(x_k,u_k)$ is known and is locally Hurwitz, where $x_k \in \Re^n$, $u_k \in \Re^m$ is the input, $k \in Z^+$ indicates time instant.
2. Measurement $y_k \in \Re^p$ is given, where $p < n$ and function $h(x_k,u_k)$ is known.
3. System is locally observable.

For a nonlinear system given in Eq. (6) and Eq. (7), a neuro-observer in the following form is proposed in this study:

$$\hat{x}_{k+1} = f(\hat{x}_k,u_k) + bNN(y_k - \hat{y}_k) \quad (8)$$
$$\hat{y}_k = h(\hat{x}_k) \quad (9)$$

where $\hat{x}_k$ represents the estimate state of the nominal system, y_k the given measurement, b a constant, $NN(y_k - \hat{y}_k)$ a neural network with $y_k - \hat{y}_k$ as input. It is easy to see that Luenberger like observer is just a sub-class of this kind of observer, where the mapping of the neural network is linear. Define the error state $e_k = x_k - \hat{x}_k$, then the error dynamics of the system can be obtained by subtracting observer Eq. (8) from nominal system Eq. (6):

$$\begin{aligned} e_{k+1} &= f(x_k,u_k) - f(\hat{x}_k,u_k) - bNN(y_k - \hat{y}_k) \\ &= f(\hat{x}_k + e_k,u_k) - f(\hat{x}_k,u_k) - bNN(h(\hat{x}_k + e_k) - h(\hat{x}_k)) \\ &= f(\hat{x}_k + e_k,u_k) - f(\hat{x}_k,u_k) - b\bar{u}_k(e_k,\hat{x}_k) \\ &= F(e_k,\hat{x}_k,u_k,\bar{u}_k) \end{aligned} \quad (10)$$

where $\hat{x}_k$ can be calculated directly from observer Eq. (8). In fact, observer Eq. (8) and error dynamics (10) are coupled during the training. The design of the observer is then changed to find a nonlinear function $\bar{u}_k(e_k,\hat{x}_k) = NN\left(h(\hat{x}_k + e_k) - h(\hat{x}_k)\right)$, such that the error state e_k is driven to zero with time. Note that this can be considered as a control design problem. There are many ways to design controllers for different nonlinear systems, such as feedback linearization [18], State Dependent Riccati Equation (SDRE) [19] and so on. In order to use those approaches, we have to know all the error states in Eq. (10), and this is actually our design objective in this study. Since the AC design is based on the powerful dynamic programming concept and the fact that the convergence of the neural networks and the algorithm have also been proved in our previous studies [20], it is the basis for the observer design in this study.

With error dynamics Eq. (10), a quadratic cost function is chosen:

$$I = \frac{1}{2}\sum_{k=0}^{\infty} U(e_k,\bar{u}(e_k,\hat{x}_k)) = \frac{1}{2}\sum_{k=0}^{\infty}\left\{e_k^T Q e_k + \bar{u}_k^T R \bar{u}_k\right\} \quad (11)$$

The discrete HJB equation is now given by:

$$\begin{aligned} J(e_k,\hat{x}_k) = \min_{\bar{u}(e_k,\hat{x}_k)}\{&U(e_k,\bar{u}(e_k,\hat{x}_k)) \\ &+ < J(F(e_k,\hat{x}_k,u_k,\bar{u}_k)) >\} \end{aligned} \quad (12)$$

where $\bar{u}_k = \bar{u}(e_k,\hat{x}_k)$. The co-state equation for the error dynamics can then be obtained from Eq. (4):

$$\lambda^*(e_k,\hat{x}_k)=\frac{\partial U(e_k,\bar{u}_k)}{\partial e_k}+\left(\frac{\partial \bar{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T\frac{\partial U(e_k,\bar{u}_k)}{\partial \bar{u}_k}$$
$$+\left(\frac{\partial e_{k+1}}{\partial e_k}+\frac{\partial e_{k+1}}{\partial \bar{u}_k}\frac{\partial \bar{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T\lambda(e_{k+1},\hat{x}_{k+1})$$
$$=Qe_k+\left(\frac{\partial \bar{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T R\bar{u}_k \tag{13}$$
$$+\left(\frac{\partial F(e_k,\hat{x}_k,u_k,\bar{u}_k)}{\partial e_k}-b\frac{\partial \bar{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T\times\lambda(e_{k+1},\hat{x}_{k+1})$$

and the stationary equation is given by:

$$0=\frac{\partial J(e_k,\hat{x}_k)}{\partial \bar{u}_k}=\frac{\partial U(e_k,\bar{u}_k)}{\partial \bar{u}_k}+\frac{\partial J(e_{k+1},\hat{x}_{k+1})}{\partial \bar{u}_k}$$
$$=R\bar{u}_k+\frac{\partial F(e_k,\hat{x}_k,u_k,\bar{u}_k)}{\partial \bar{u}_k}\lambda(e_{k+1},\hat{x}_{k+1})$$
$$\bar{u}_k^*=R^{-1}b\lambda(e_{k+1},\hat{x}_{k+1}) \tag{14}$$

Where λ^* and $\bar{u}_k^*$ are targets of critic neural network and action neural network respectively. Then the training of the neural networks can start with choosing e_k randomly [21]. The neural networks in our study are MLPs where the number of layer and the number of neuron in each hidden layer should be determined by the complexity of the problem.

Up to this point the AC based neuro-observer design method is almost the same as that of the controller design method. There are some differences, however:

1. Since only measurements are known for feedback and the measurement vector has a smaller dimension than that of the state vector, the number of inputs to the action neural network is therefore less than the number of the states.
2. Where the measurements are nonlinear, two approaches can be used to train the action neural network. First some kind of nonlinear transformation can be used to get corresponding state from the nonlinear measurement before training the action neural network; second the nonlinear measurement can be used directly as input to train the action neural network. The first approach is pretty straightforward; the second approach is also applicable because of the universal approximation capability of the neural networks. Some examples will be given later to illustrate these two approaches.
3. Consider $\bar{u}_k$ is not a real controller, and optimal controller is not the objective of the design, the iteration process of the algorithm can be simplified such that only one or a few iterations are used to find $\bar{u}_k$. Thus the time used for the training of the neural network can be tremendously reduced.
4. The meaning of matrix R in the cost functional (11) should be redefined. It can be seen from the following simulations that matrix R is an important parameter to adjust the time for the state trajectories of the observer to track the desired state trajectories. While in ACBOC method it is used to adjust the magnitude of the controller such that the control effort is limited in a desired range.
5. Robustness is another important issue in observer design, since the measurements are always combinations of useful signals and noises. A robust observer should be able to keep the tracking error bounded in a small range given noisy measurements. The design approach introduced in this section also shows good robustness as evidenced in the simulations.

4. Illustrations:

A second order nonlinear system is used in this section to illustrate the neuro-observer design approach. The effect of matrix R in the cost functional (11) on the tracking of the desired trajectories of the system is studied, the results while the measurement is nonlinear are plotted, and robustness of the observer given noisy measurement is also illustrated in the following sections.

A single-link robot arm rotating in a vertical plane

The equation of motion is:

$$M\ddot{q}+\frac{1}{2}mgl\sin q=u \tag{15}$$
$$y=q \tag{16}$$

In which q is the angle, u the input torque, M the moment of inertia, g the gravity constant, m and l are the mass and the length of the link, and the measurement is the angle. The robot parameters are (in *SI* units): $m = 1$, $l = 1$, $M = 0.5$, and $g = 9.8$. Letting $x_1 = q$ and $x_2 = \dot{q}$, the state space representation of the system given in Eq. (15) and Eq. (16) can be written as follow:

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix}=\begin{pmatrix}0 & 1\\ 0 & 0\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix}+\begin{pmatrix}0\\ 1\end{pmatrix}\left(u-\frac{1}{2}mgl\sin x_1\right)/M \tag{17}$$
$$y=\begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} \tag{18}$$

Where input $u=\sin(2t)+\cos(20t)$ is given.

According to the previous section, we can design a neuro-observer for this system:

$$\begin{pmatrix}\dot{\hat{x}}_1\\ \dot{\hat{x}}_2\end{pmatrix}=\begin{pmatrix}0 & 1\\ 0 & 0\end{pmatrix}\begin{pmatrix}\hat{x}_1\\ \hat{x}_2\end{pmatrix}+\frac{1}{M}\begin{pmatrix}0\\ 1\end{pmatrix}\left(u-\frac{1}{2}mgl\sin\hat{x}_1\right)-NN(y_m-\hat{y}) \tag{19}$$
$$\hat{y}=\begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}\hat{x}_1\\ \hat{x}_2\end{pmatrix} \tag{20}$$

The error dynamics are:

$$\begin{pmatrix}\dot{e}_1\\ \dot{e}_2\end{pmatrix}=\begin{pmatrix}0 & 1\\ 0 & 0\end{pmatrix}\begin{pmatrix}e_1\\ e_2\end{pmatrix}+\frac{1}{M}\begin{pmatrix}0\\ 1\end{pmatrix}\left(-\frac{1}{2}mgl(\sin(e_1+\hat{x}_1)-\sin(\hat{x}_1))\right) \tag{21}$$
$$+NN(y_m-\hat{y})$$

Where y_m is the measurement. The observer (19) and the error dynamics (21) are then discretized using a time step $\Delta t = 0.01\sec$.

$$\begin{pmatrix}\hat{x}_{k+1,1}\\ \hat{x}_{k+1,2}\end{pmatrix}=\begin{pmatrix}1 & \Delta t\\ 0 & 1\end{pmatrix}\begin{pmatrix}\hat{x}_{k,1}\\ \hat{x}_{k,2}\end{pmatrix}+\frac{1}{M}\begin{pmatrix}0\\ \Delta t\end{pmatrix}\left(u-\frac{1}{2}mgl\sin\hat{x}_1\right) \tag{22}$$
$$+\Delta tNN(y_{mk}-\hat{y}_k)$$

$$\begin{pmatrix}e_{k+1,1}\\ e_{k+1,2}\end{pmatrix}=\begin{pmatrix}1 & \Delta t\\ 0 & 1\end{pmatrix}\begin{pmatrix}e_{k,1}\\ e_{k,2}\end{pmatrix}$$
$$+\frac{1}{M}\begin{pmatrix}0\\ \Delta t\end{pmatrix}\left(-\frac{1}{2}mgl(\sin(e_{k,1}+\hat{x}_{k,1})-\sin(\hat{x}_{k,1}))\right) \tag{23}$$
$$+\Delta tNN(y_{mk}-\hat{y}_k)$$

where $e_k=\begin{pmatrix}e_{k,1} & e_{k,2}\end{pmatrix}^T$, $x_k=\begin{pmatrix}x_{k,1} & x_{k,2}\end{pmatrix}^T$, the first index k denote the time index, the second index tells us whether the state is the first state or the second state, $\bar{u}_k(e_k,\hat{x}_k)=NN(y_{mk}-\hat{y}_k)$, y_{mk} is the discrete measurement. The cost functional is chosen to be in a quadratic form the same as Eq. (11), then the co-state equation and the stationary equation can be obtained from Eq. (13) and Eq. (14 separately.

$$\lambda(e_k,\hat{x}_k)=Q\begin{pmatrix}e_{k1}\\e_{k2}\end{pmatrix}+\left(\frac{\partial\overline{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T R\overline{u}_k$$
$$+\left(\begin{pmatrix}1 & \Delta t\\ \frac{-\Delta t m g l}{2M}\cos(e_{k1}+\hat{x}_{k1}) & 1\end{pmatrix}+\Delta t\frac{\partial\overline{u}(e_k,\hat{x}_k)}{\partial e_k}\right)^T \quad (24)$$
$$\lambda(e_{k+1},\hat{x}_{k+1})$$

$$\overline{u}_k^* = -R^{-1}\Delta t\lambda(e_{k+1},\hat{x}_{k+1}) \quad (25)$$

The Q matrix is chosen to be an identity matrix and matrix R = *2diag{1 1}* (a two by two matrix with all diagonal elements to be 2) in the cost functional (11). Two MLPs with two hidden layers and one output layer are chosen to act as *Action NN* and *Critic NN*. There are eight neurons in each of the hidden layer and two neurons in the output layer. The activation functions in the hidden layers are chosen to be tangent sigmoid functions and pure linear functions are used in the output layers.

Fig. 2(a) and Fig. 2(b) show the state trajectories when the initial state is $\hat{x}_0=[-0.6 \quad 0.6]^T$ for the observer. The dashed lines denote the exact state trajectories of the nominal system when the initial state is $x_0=[0 \quad 0]^T$. Fig. 2(c) shows the state trajectory errors for those different initial states.

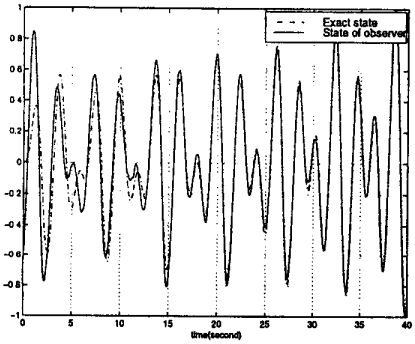

Fig. 2(a) Trajectory of state $x_1, \hat{x}_1$ for $\hat{x}_0=[-0.6 \quad 0.6]^T$

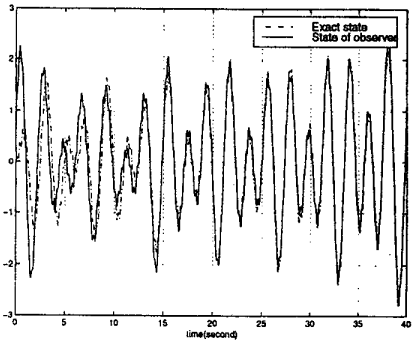

Fig. 2(b) Trajectory of state $x_2, \hat{x}_2$ for $\hat{x}_0=[-0.6 \quad 0.6]^T$

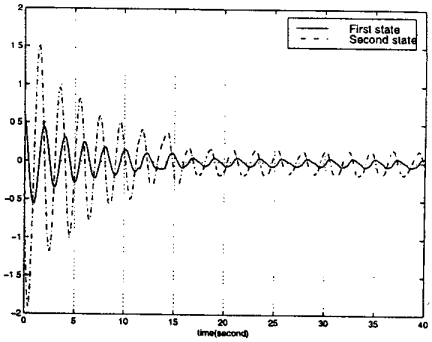

Fig. 2(c) Tracking error for $\hat{x}_0=[-0.6 \quad 0.6]^T$

In order to demonstrate the affection of matrix R on the tracking of the state trajectories of the observer to the desired state trajectories, it is changed to be R =*0.2diag{1 1}*, and the state trajectories are plotted in Fig. 3(a) and Fig. 3(b). The tracking errors are plotted in Fig. 3(c). It is obvious that the tracking speed and accuracy are increased significantly using a smaller R matrix

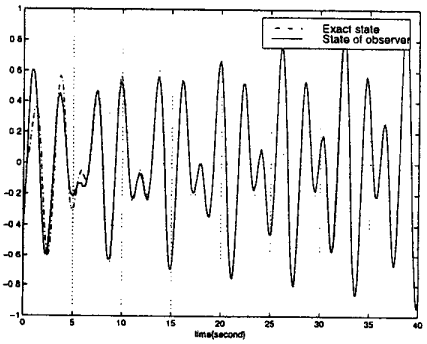

Fig. 3(a) Trajectory of state $x_1, \hat{x}_1$

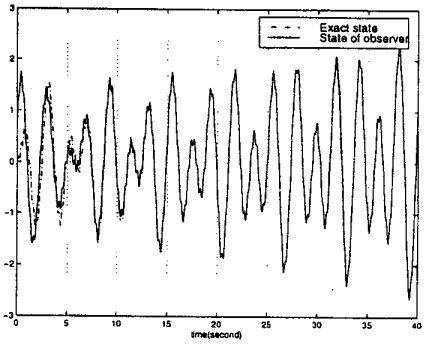

Fig. 3(b) Trajectory of state $x_2, \hat{x}_2$

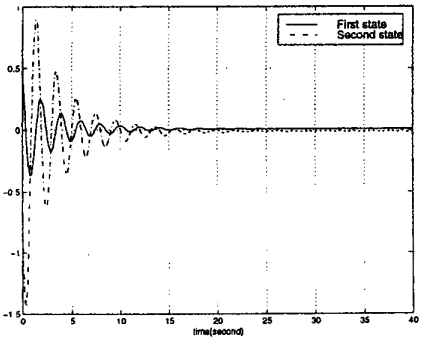

Fig. 3(c) Tracking error

The measurements used in the above examples are all linear measurements. When nonlinear measurements are used, this design method can still generate the desired results. Let $y = x_1^3$ and $\hat{y} = \hat{x}_1^3$ in the above example. Basically, a nonlinear transformation $x_1 = \sqrt[3]{y}$ can be used to calculate the state x_1 in this example, but when the measurement contains noise, this transformation approach is not proper. Hence, we consider using the nonlinear measurements as inputs to the action neural network directly. The state trajectories of the observer and the nominal system are plotted in Fig. 4(a) and Fig. 4(b) and the tracking error is plotted in Fig. 4(c). It's obvious the results are reasonable.

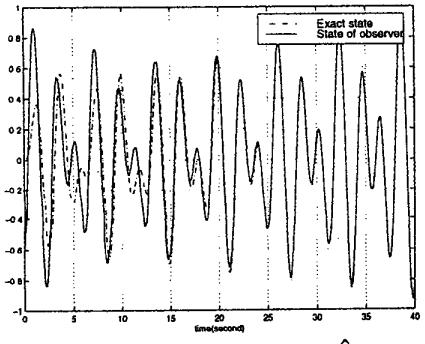

Fig. 4(a) Trajectory of state $x_1, \hat{x}_1$

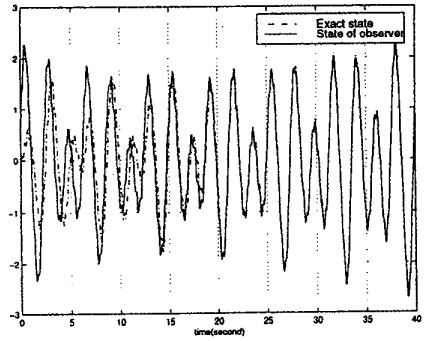

Fig. 4(b) Trajectory of state $x_2, \hat{x}_2$

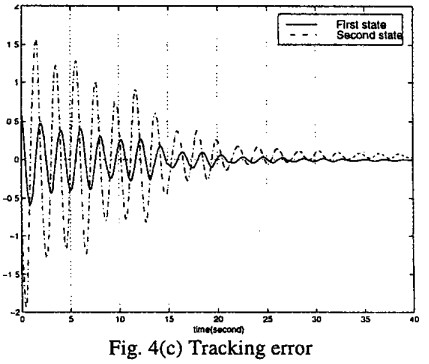

Fig. 4(c) Tracking error

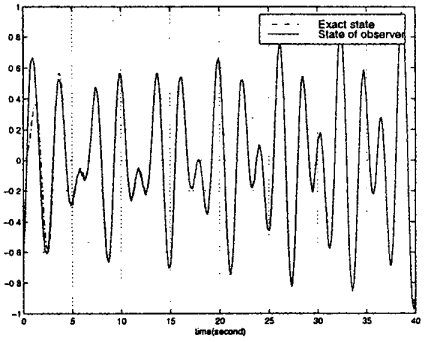

Fig. 5(a) Trajectory of state $x_1, \hat{x}_1$

Now, we can demonstrate the robustness of this observer design method to noise. Let the measurement be $y = x_1 + v(t)$, where v(t) is a noise distributed uniformly in the range of [-0.5, 0.5]. The same observer designed with *R=diag[0.2 0.2]* is used for this noisy measurement example. The state trajectories of the observer and the nominal system are plotted in Fig. 5(a) and Fig. 5(b). The noisy measurement is plotted in Fig. 5(c) and the tracking error in Fig. 5(d). We can see that the state trajectories can track the desired state trajectories very well given noisy measurements.

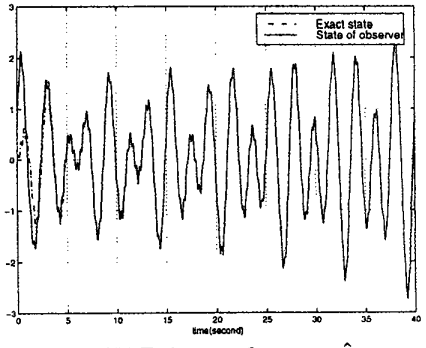

Fig. 5(b) Trajectory of state $x_2, \hat{x}_2$

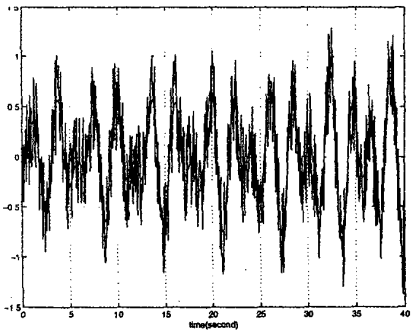

Fig. 5(c) Noisy measurement $y = x_1 + v(t)$

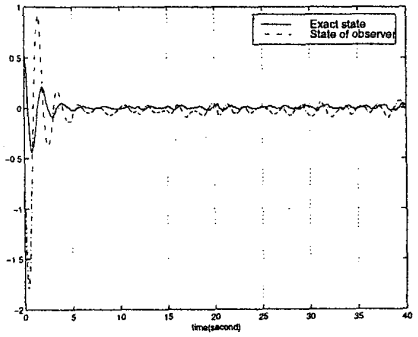

Fig. 5(d) Tracking error

5 Conclusion

The observer design developed in this study is a general methodology that can be used to design observers for a large class of nonlinear systems, with linear or nonlinear measurements. It exhibits good levels of robustness in the test simulations. Therefore, desired tracking results can be obtained even when the measurements are quite noisy. However, the above observers are designed particularly for systems whose functional forms are known before hand. In practice, there are many systems that may not be totally known, i. e. only parts of systems are known before hand. Therefore, it is not enough to design observers for fixed systems only. Now, our research is focused on extending this neuro-observer design for systems which are not totally known or partially known.

Reference

1. Bader Aloliwi, Hassan K.Khalil and Elias G.Strangas, *Robust Speed Control of Induction Motors Using Adaptive Observer*, Proceedings of the American Control Conference, San Diego, California, June 1999, pp. 931-935
2. Kal Brammer and Gerhard siffling, *Kalman-Bucy Filters*, Artech House, INC, 1989
3. D.G.Luenberger, *Observing the State of Linear System*, IEEE Trans. Military Electron, Vol. 8, pp. 74-90, 1964
4. Georges Bastin and Michel R. Gevers, *Stable Adaptive Observers for Nonlinear Time-Varying Systems*, IEEE Trans. Auto. Ctrl., Vol. 33, No. 7, pp. 650-657, 1988
5. Yaolong Tan, Kanellakopoulos I. and Zhong-Ping Jiang, *Nonlinear Observer / Controller Design for a Class of Nonlinear Systems*, Proceedings of the 37th IEEE Conference on Decision and Control, Piscataway, NJ, Vol. 3, pp.2503-2508. 1998
6. Krishna K. Busawon and Mehrdad Saif, *A Constant Gain Observer for Nonlinear Systems*, Proceedings of the 1993 American Control Conference, Evanston, IL, Vol. 4, pp. 2334-2338, 1998
7. Jun Wang and Guang Wu, *Real-Time Synthesis of Linear State Observers Using a Multi-layer Recurrent Neural Networks*, Proceedings of The IEEE International Conference on Industrial Technology, New York, NY, pp. 278-282, 1994
8. Thomas Frenz and Dierk SchrÖder, *Learning Unknown Nonlinearities Using a Discrete Observer in Combination With Neural Networks*, Conference Record of the 1995 IEEE Industry Applications Conference, New York, NY, Vol. 2, pp. 1800-1806, 1995
9. D. Strobl, U. Lenz and D. SchrÖder, *Systematic Design of Stable Neural Observers for a Class of Nonlinear Systems*, Proceedings of the 1997 IEEE International Conference on Control Applications, Hartford, pp. 377-82, 1997
10. C.L.Hwang and F.Y.Sung, *Neuro-Observer Controller Design for Nonlinear Dynamical Systems*, Proceedings of the 35th IEEE Conference on Decision and control, Kobe, Japan, Vol.3, pp. 3310-3315, 1996
11. H.Y.Zhang, C.W.Chan, K.C.Cheung and H.Jin, *Nonlinear Observer Design with Unknown Nonlinearity via B-Spline Network Approach*, Proceedings of the American Control Conference, Philadelphia, Pennsylvania, June 1998, pp. 2339-2343
12. Y.H.Kim, F.L.Lewis and C.T.Abdallah, *Nonlinear Observer Design Using Dynamic Recurrent Neural Networks*, Proceedings of the 35th Conference on Decision and Control, Kobe, Japan, December 1996, pp. 949-954
13. Ruijun Zhu, Tianyou Chai and Cheng Shao, *Robust Nonlinear Adaptive Observer Design Using Dynamic Recurrent Neural Networks*, American Control Conference, Albuquerque, New Mexico, pp. 1096-1100, June 1997
14. E.B.Kosmatopoulos, M.M.Polycarpou, *High-Order Neural Network Structures for Identification of Dynamical Systems*, IEEE Trans. on Neural Networks, Vol. 6, No. 2, pp. 422-431, March 1995
15. R.Marino and P. Tomei, *Adaptive Observer with Arbitrary Exponential Rate of Convergence for Nonlinear Systems*, IEEE Trans. Auto. Ctrl., Vol. 40, No. 7, pp. 1300-1304, 1995
16. R. Boudarel, J.Delmas, P.Guichet, *Dynamic Programming and Its Application to Optimal Control*, New York, NY, Academic Press, 1971
17. D.A. White and D.A.Sofge, Eds., *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, New York, NY, Van Nostrand Reinhold, 1992
18. J. E. Slotine and W. Li, Applied Nonlinear Control, Englewood, NJ: Prentice Hall, 1991
19. J. R. Sloutier and D. T. Stansbery, Control of Continuously Stirred Tank Reactor Using an Asymmetric Solution of the State-Dependent Riccati Equation, Proceedings of the 199 IEEE International Conference on Control Applications Piscataway, NJ, Vol. 2, pp. 893-898, 1999
20. X. Liu and S. N. Balakrishnan, Convergence Analysis c Adaptive Critic Based Optimal Control, Proceedings of th American Control Conference, Chicago, Illinois, June 200 pp 1229-1933.
21. S. N. Balakrishnan, V. Biega, "Adaptive critic based neu networks for aircraft optimal control," Journal of Guidan Control, and Dynamics, Vol. 19, No. 4, July-August 19 pp. 893-898