

Article

Adaptive Sampling for WSAN Control Applications Using Artificial Neural Networks

Daniel N. Nkwogu and Alastair R. Allen *

School of Engineering, University of Aberdeen, Aberdeen AB24 3UE, Scotland, UK;

E-Mail: r01dnn8@abdn.ac.uk

* Author to whom correspondence should be addressed; E-Mail: a.allen@abdn.ac.uk;
Tel.: +44-1224-272-501; Fax: +44-1224-272-497.

Received: 28 September 2012; in revised form: 29 October 2012 / Accepted: 11 November 2012 /

Published: 30 November 2012

Abstract: Wireless sensor actuator networks are becoming a solution for control applications. Reliable data transmission and real time constraints are the most significant challenges. Control applications will have some Quality of Service (QoS) requirements from the sensor network, such as minimum delay and guaranteed delivery of packets. We investigate variable sampling method to mitigate the effects of time delays in wireless networked control systems using an observer based control system model. Our focus for variable sampling methodology is to determine the appropriate neural network topology for delay prediction and also investigate the impact of additional inputs to the neural network such as network packet loss rate and throughput. The major contribution of this work is the use of typical obtainable delay series for training the neural network. Most studies have used random generated numbers, which are not a correct representation of delays actually experienced in a wireless network. Our results here shows that adequate prediction of the time delay series using the observer based variable sampling is able to compensate for delays in the communication loop and influences the performance of the control system model.

Keywords: WSAN; Zigbee; ANN; QoS; WPAN

1. Introduction

Recent advances in wireless microelectronic and embedded systems are the backbone of the potential for wireless sensor actuator networks (WSANs) emerging as a viable solution or alternative for closing control loops over wireless networks. The feedback methodology is the central element of a control loop. The WSAN configuration exploits the feedback methodology. As communication plays a central role in distributed systems, it is important that communication protocols or frame works are developed to ensure service is given by a system to an end user in a reliable and efficient manner [1]. Packet based transmission over wireless channels is expected to experience loss and delays. Control applications will have some Quality of Service (QoS) requirements from the sensor network, such as requiring minimum delay and guaranteed delivery of packets [1]. Packet loss or time varying delay can seriously degrade the control performance and cause instability [1]. Closing control loops over wireless channels provides many challenging control problems such as management of data loss and delays, communication scheduling, bandwidth utilization, performance evaluation and stability analysis. During data exchange between wireless devices, the induced delay caused by the wireless network is an important issue. Adaptive sampling can help alleviate the impact of time varying delays on quality of control. Sensors are conventionally designed to have fixed sampling periods. Fixed sampling periods allow for fixed control loop bandwidth requirements given that data packet sizes are fixed. However, in a dynamic environment such as an adaptive sampling rate, this may result in varying network load, which may become overloaded or under-loaded. In this paper, attention is paid to the impact of delay problems on quality of control. This delay experienced usually varies at random. One of the important goals of a wireless control system is to achieve adaptability to the changing network conditions. This makes adaptive learning for wireless networks essential with artificial neural networks (ANN) a suitable candidate to achieve this learning capability. Adaptive learning capability is an essential characteristic of neural networks and they are fast at data processing due to their parallel structure.

Based on the concept of variable period sampling, we investigate the delay problem in WSAN control systems and adaptive sample period mechanism using artificial neural networks. Since delay varies at random in wireless control systems, the system is sampled at each sampling step with a different time interval and this sampling period is calculated based on the predictions made by the neural network algorithm. The predicted delay is used to control the WSAN control system to the next sampling step. We investigate sample period prediction using the backpropagation feedforward, non linear autoregressive, radial basis and time series neural network models. An analysis of the performance of the different neural network types is given. The major contribution of this work is the use of typical obtainable delay series for training the neural network. Most studies have used random generated numbers, which does not reflect typical delays actually experienced in a wireless network.

The rest of this paper is organized as follows. Section 2 provides an overview on WSAN control applications focusing on delay and stability. Section 3 discusses neural networks and related work carried out relating to delay prediction and network stability. In Section 4, we present the ANN adaptive sampling mechanism. In Section 5 experimental studies are given. Finally, Section 6 concludes this paper.

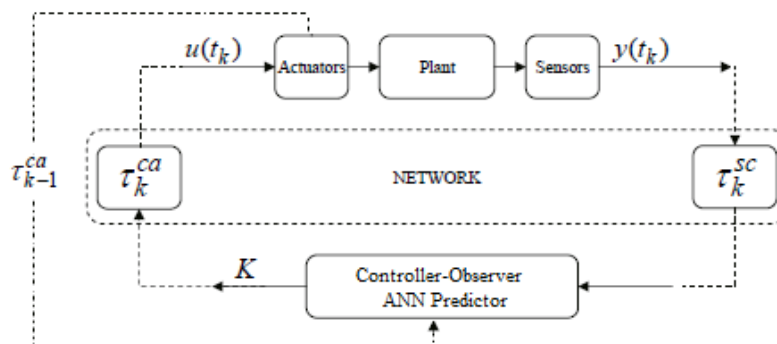
2. WSAN Network Model

A WSAN control system architecture consists of sensor nodes, actuator nodes and a controller. We briefly describe the IEEE 802.15.4 standard for low rate wireless personal area networks (PANs). The ISA100.11a standard developed by the International Society of Automation (ISA) defines Wireless Systems for Industrial Automation Process Control and Related Applications and is built based on the IEEE 802.15.4. The ZigBee protocols are also based on this standard. The IEEE 802.15.4 standard defines the physical and MAC layer specification of WSNs for low rate PANs and aims to provide a low-power, low-cost and reliable protocol for wireless connectivity. The typical operating range of IEEE 802.15.4 is approximately 10 to 20 meters, and the raw data rate is 250 kb/s in the 2.4 GHz band [2].

2.1. Delay and Wireless Control Systems

Factors affecting delay in a wireless control applications network include interference, distance and the underlying protocols used in the network. In WSANs, actuators perform their actions based on received data from the sensor; hence delay bounds are crucial for real time performance [1]. Real time in this context means executing commands or communication within the systems timing requirements. The challenge lies in guaranteeing minimal delay and jitter so that it will not degrade control performance. The knowledge of the different types of delay experienced in a WSAN, including their proportion in the total delay is important. This determines the impact of the delays on the performance of the WSAN control system. The contributions of these various delay components can vary significantly and depend on the communication protocols used, the network traffic patterns and the channel conditions, network devices, throughput and congestion conditions of the network during transmission [3]. The network conditions make WSAN delays non-deterministic and delays can be of different natures, constant, variable or random [4]. For this reason, propagation delay can be regarded as the dominant delay of the network loop [5]. The non-deterministic nature of the network conditions shows the importance of creating a representation of the network delays to be able to create a suitable control for a certain control system.

Figure 1. WSAN control system with induced delays.



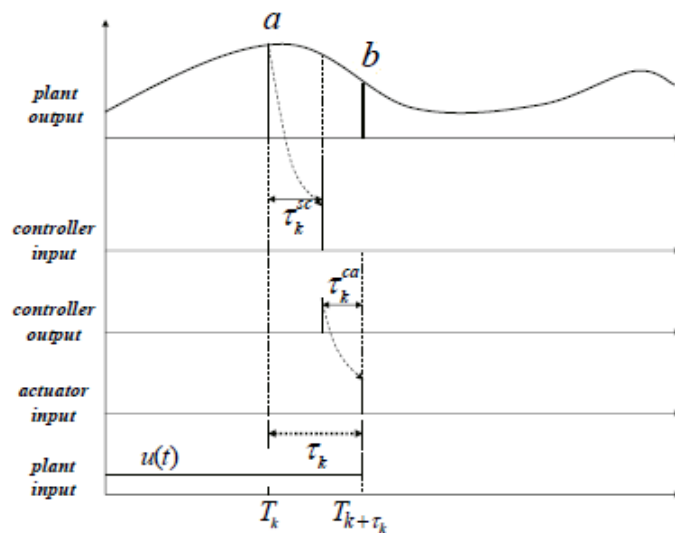
We discuss the model of a networked control system as shown in Figure 1. The data packets from the plant conditions sensed by the sensor travel from sensor to controller, then processed by the controller to compute some control algorithm and then transmitted from controller to actuator. The

actuator then receives the actuator signal from the controller and executes a required action. Due to network conditions, information sent by the sensor-to-controller or controller-to-actuator experiences a delay. Such delays are called network-induced delays. The delays are denoted as sensor-to-controller delays τ_{sc} and controller-to-actuators delay τ_{ca} . The following assumptions are made for this model.

1. Computational, processing and transmission delays are absorbed by τ_{sc} or τ_{ca} .
2. Sensors data is time-stamped prior to transmission to the controller.
3. The controller has information on last time controller to actuator delay (using the 802.15.4 data acknowledgement mechanism). This information is used by the neural network to compute the delay τ_{ca} that is going to happen.

The control feedback loop network delay propagation can be described using the timing diagram in Figure 2.

Figure 2. Network delay propagation timing diagram.



From the timing diagram the packet containing the sensed plant state is at point (a) at sampling instant T_k . The sensor transmits its sensed information to the controller. During transmission and propagation, the packet experiences a time lag (delay) τ_{sc} prior to arrival. The received information is then used by the controller to compute a control signal. This control signal is then transmitted to the actuator and further experiences a time delay τ_{ca} , which is applied to the plant. When the control signal is applied at point (b), the control signal computed at τ_{sc} may not reflect the plant conditions at point (b). Hence, the control signal is not updated for the actual plant conditions. The total delay τ_k can be used to compute a more suitable control signal to reflect the plant conditions at point (b). Then the idea is to predict (at sampling step T_k) the total delay $\tau_k = \tau_{sc} + \tau_{ca}$ that is going to happen. This delay prediction will be achieved using artificial neural network algorithms. This approach requires that the controller and actuator are event driven and the system now becomes a discrete linear time-invariant variable sampling system.

Our system model [6] is based on the classical approach to sampling dynamic systems with delays and is described below. Consider the continuous-time representation of a dynamic system subject to input delays given as

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t - \tau), x(t_0) \\ y(t) &= Cx(t)\end{aligned}$$

with feedback controller

$$u(t) = -Kx(t)$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$ represent the system states, the controlled input and output respectively. A , B and C are constant matrices of appropriate dimensions. K is the $m \times n$ feedback gain matrix. Here the system has variable sampling intervals, which implies that the system determines when the control signal is changed at sampling instant Tk . The total delay τk is assumed to be same as the sampling period Tk as it is expected that the ANN will precisely predict the future total delay τk [7]. The predicted delay $\tau k = Tk$, now becomes a parameter of the system matrix Φ and the input distribution matrix Γ . The discretized representation of the linear time-variant system with time-delay affecting the feedback is

$$x(t_k + 1) = \Phi(Tk)x(t_k) + \Gamma(Tk)u(t_k)$$

where $\Phi(T_k) = e^{AT_k}$, $\Gamma(T_k) = \int_0^{T_k} e^{A(t_k - \tau)} B d\tau$.

From the equations above, the plant's current output $y(t_k)$ and its future state are determined solely by its current state $x(t_k)$ and the current inputs $u(t_k)$. When all the states $x(t_k)$ cannot be measured, an observer is used to estimate them. The observer is basically a copy of the plant that estimates the future plant states with an extra term that compares the actual measured output $y(t_k)$ to the estimated output $\hat{y}(t_k)$, which will cause the estimated states $\hat{x}(t_k)$ to approach the values of the actual states $x(t_k)$. The prediction observers estimate the future plant states with differing sample times as predicted by the neural network. Control actions are calculated using the predicted states. Assuming $u(tk)$ remains the same for $tk \leq t \leq tk+1$, the state x can be approximated by

$$\hat{x}(t_k + 1) = \Phi(T_k)\hat{x}(t_k) + \Gamma(T_k)u(t_k)$$

Introducing the difference between the measured and estimated outputs $y(t_k) - \hat{y}(t_k)$, where $\hat{y}(t_k) = C\hat{x}(t_k)$, the predicted state with observer becomes

$$\begin{aligned}\hat{x}(t_k + 1) &= \Phi(T_k)\hat{x}(t_k) + \Gamma(T_k)u(t_k) + L(T_k)(y(t_k) - \hat{y}(t_k)) \\ \hat{x}(t_k + 1) &= \Phi(T_k)\hat{x}(t_k) + \Gamma(T_k)u(t_k) + L(T_k)(Cx(t_k) - C\hat{x}(t_k))\end{aligned}$$

Here $L(Tk)$ is the observer gain matrix based on the predicted values Tk . The control feedback $u(tk)$ becomes an estimated value based on the observer as denoted by

$$\hat{u}(t_k) = -K(T_k)\hat{x}(t_k)$$

The closed loop system is then expressed as

$$\begin{bmatrix} x(t_k + 1) \\ \hat{x}(t_k + 1) \end{bmatrix} = \begin{bmatrix} \Phi(T_k) & -\Gamma(T_k)u(t_k) \\ L(T_k)C & \Phi(T_k)\hat{x}(t_k) - \Gamma(T_k)u(t_k) - L(T_k)C \end{bmatrix} \begin{bmatrix} x(t_k) \\ \hat{x}(t_k) \end{bmatrix}$$

Selection of the feedback gain K and the observer gain L are based on optimal control given by the linear quadratic regulator (LQR) technique for discrete systems and the Ackerman method for pole

placement respectively. These system models are the basis for the control simulations carried out in section V.

2.2. Stability Analysis

WSANs are inherently prone to transmission delays. The variation in delay can degrade control performance or even destabilize the system. Hence, it is important that the sampling rate of the system is adjusted within the limits or optimal performance criteria to achieve desired system performance and stability [8,9]. However, the sampling rate must not be adjusted beyond limits that would cause system instability. In a wireless control network, a fast sampling rate can increase the network load thereby increasing transmission delays within the shared medium [10]. We look at two issues in wireless networked systems: system stability and control performance.

System stability: In this paper, we define system network stability as a network whose average rate at which data is generated (network load) is smaller than the network capacity (maximum throughput, bitrate). For instance, consider an IEEE802.15.4 based control loop with a maximum bite rate of 250 k/bps as defined in the standard. Studies in [11,12] have shown that the maximum theoretical throughput achievable in an IEEE802.15.4 based network is 225 k/bps. The IEEE802.15.4 physical frame format consists of preamble (4 bytes), SFD (1 byte), frame length field (1 byte), data payload (127 bytes) and 6 bytes representing transmission overhead. Therefore, assuming a maximum theoretical throughput of 225 k/bps and perfect queuing, the smallest allowable sampling period without overloading the network is calculated as network throughput/data size in bits $225000/1128 = 0.005 \text{ s} = 5 \text{ ms}$

Control Performance: We have assumed that the queuing and processing delays experienced are negligible and can form part of the sensor to controller and controller to actuator delays. The following are the considerations taken regarding the duration of delays in the system loop. As stated above, it is important that the correct sensor data is received in time in a control loop subject to delays within its feedback communication to guarantee that the correct information is used to control the loop. Consider a controlled system, if at the current sampling step, the total delay τk (sensor to controller and controller to actuator delay) is smaller than the maximum allowed delay T_{max} , then the system is stable. This implies that the controlled process may become unstable or not controlled correctly when the combination of the sensor to controller and controller to actuator delay is greater than the sample time because the current sensed data is not used to control the system. A delay where $\tau k > T_{max}$, can be classed a packet loss.

3. Artificial Neural Network Models and Related Work

3.1. Neural Network Models

An artificial neural network is a mathematical computational model. It generally consists of an interconnected group of neurons and can be used to model complex relationships between inputs and outputs. Artificial neural networks in most cases are adaptive systems that change their structure based on external or internal information flowing through them and use a connectionist approach to process information [13].

Back propagation feedforward networks are standard neural networks for any supervised learning pattern recognition. The error between the network output and target output is backpropagated through the network and used to update the weights.

The time delay neural network (TDNN) is a multilayer feedforward network whose past values of the time series are used to predict future values using tapped delay lines. The term “tapped-delay-line” means that the neural network needs to have certain memory of previous values of the input and/or output events. TDNNs are similar to back propagation feedforward neural networks and are designed to deal with time varying signals.

The non-linear auto regressive neural network with exogenous input NARXNN is a recurrent dynamic multilayer network. The network architecture consists of feedback connections enclosing several layers of the network. The NARXNN is similar to the TDNN. The essential difference between the TDNN and NARXNN is that the output of the NARXNN is fed back as an input to the network [14].

A radial basis function network (RBFNN) is a multilayer feedforward neural network. The RBFNN uses a radial basis activation function. RBFNN's have been used in function approximation, time-series prediction and control applications. In multilayer networks, the radial basis functions are commonly used to replace the sigmoidal hidden layer transfer functions [13].

3.2. Related Work

Various methods have proposed to adapt the sampling periods of controls loops. In [15] the sampling period of the control loops is adjusted at the application layer based on information about deadline miss ratio and transmission rate from the physical layer. Control performance is then maximized through controlling the deadline miss ratio and it is suggested that enlarging the sampling period can reduce the deadline miss ratio. In [7], a backpropagation (BP) feedforward neural network is used to predict delay. The predicted delay is then used as the sampling period. By comparing control simulations using variable and fixed sampling periods they show that variable period sampling can alleviate the influence of delay to a great extent and also improve the performance of the networked control systems. In [16], a scheme for predicting mean per-packet one-hop delays using Multi-Layer Perceptron (MLP) network or tapped-delay-line Radial Basis Function (RBF) neural network approaches is proposed. Based on nodes' past experiences of traffic conditions they make future predictions of the network traffic, and the predicted one-hop delays are then used by the nodes to participate in routing decisions. In [17], a feedforward neural network is used to predict delays. Their simulation results show that their approach reduces the influence of closed loop delay and they ensure system stability by closed loop pole placement. Stability based sample period adaptation method is proposed in [8] and focuses on the analysis of stability and performance of a basic wireless communication strategy in the areas of switched systems, stochastic stability and H_∞ disturbance-to-output gain. Their proposed strategy adapts the sample period by shifting to a more favorable network operating point based on network disturbances and changes in network loading by estimating the network state in terms of a 2-state Markov model. A multiple-observer approach to stability Wireless network Control Systems is described in [9] and addresses the problems of both time varying delays and packet loss in wireless control networks. To compensate for time varying delay, a State Prediction Observer (SPO) is designed using Linear Matrix Inequalities (LMI) to ensure closed-loop stability.

They show that this approach maintains stability in the presence of severe time-varying delays. The analysis of delay and stability of wireless control system is discussed in [10] and the influence of the sampling rate and network delay on system stability. Further studied is the stability of wireless control system using a hybrid system stability analysis technique. The random delay discrete time domain stabilization problem in wireless control system is also investigated in [18]. Here the sensor-to-controller delay and controller-to-sensor delay is modeled as two Markov chains and the necessary and sufficient conditions on the existence of the stabilizing controllers established. To calculate the state feedback gains an iterative linear matrix inequality approach is used. Nilsson [19] analyzed networked control system in discrete-time domain, and further modeled the network delay as constant, independently random, and random but governed by an underlying Markov chain. By solving a LQG optimal control problem, he generated a controller that guaranteed the system stability. However, the design of the controller used the knowledge of known distribution or the state of Markov chain. [19] discusses the modeling and analysis of real time systems subject to varying delay in a network. They present a new control scheme analysis based on stochastic control theory. They assume the probability distribution of delays is known. Their numerical results support theoretical claims on the influence of random delays.

As discussed above, random delay in wireless networked control systems is inevitable. In [7,16,17], their experiments have not used actual delays from practical or typical scenarios. They have used randomly generated delay values, which do not represent the delays experienced in a WSN control system. Works carried out on adaptive sampling techniques have not investigated the importance of accurate prediction of network delays.

4. Neural Network Based Adaptive Sampling Mechanism

In our proposed sampling period adaptation prediction method, the sample period is predicted using known historical delay values, network load and throughput as the input layer vectors. The purpose of investigating different neural network models is to verify the appropriate neural network model for delay prediction.

4.1. Prediction Target

At the current sampling step, the neural network is required to predict the network delay. This requires that the neural network is properly trained to reliably predict the real delay values to occur. To train the network, the desired/target output needs to be known. We use simulated delay results from the OPNET [20] modeler wireless network simulation results based on the zigbee protocol as the desired output. After the offline training, the neural network is switched online. Since it is not possible to know the real delay value in advance at the current sampling step, the real delay value at current sampling step becomes available at next sampling step. Using the known historical delay data, the network predicts the delay at the current sampling step. The predicted delay is then used to control the wireless networked control system to the next sampling step. At this time, the real network delay from the previous sampling step is now obtained. The real delay value is compared with the predicted time delay and the error difference calculated. If the error is below a predefined threshold, then the prediction is still precise and the weight and values of the network do not require updating.

4.2. Simulation Data

Data collection is an important part of the neural network modeling process. The performance of neural network models is impacted by the quality and quantity of data used. Our data used to validate the proposed neural network models are real data of delays occurring within a wireless sensor actuator network. We use delay results from the OPNET modeler wireless network simulation based on the 802.15.4/Zigbee protocol. The network consists of 200 sensor/actuator nodes communicating via wireless connections. Simulations were carried out for 30 minutes and the sequence of data obtained includes network end-to-end delay, packet loss and network throughput. Each value of the sequence was taken every 180 ms giving a data set of 1,000 points. The data set was divided into a training set of 70%, validation set of 15% and testing set of 15%. The experiment aims to predict delay series in wireless sensor actuator networks, but we also use the network throughput and packet loss as additional inputs to the network. The aim is to determine the correlation and effect of the additional inputs in precise prediction of the delay.

For efficient neural network training, data pre-processing is normally performed on the network inputs and targets. In neural networks, there are different activation function types, and the various activation functions are centered on certain values in their output space. Pre-processing is done so that the data falls within their range. In multilayer feedforward networks, sigmoid transfer functions are normally used in the hidden layers. Sigmoid functions will be saturated if the net input becomes greater than three exponents. For instance, the *log* sigmoid function is centered around 0.5 with a range of (0,1) and the *tanh* sigmoid function has its center around zero with a range of (-1,1) [21]. If saturation occurs at the start of the training, the error gradients will become small, leading to a slow network training and longer convergence. To perform the correct normalization the configuration settings will include the minimum and maximum values of the sample data and calculated as follows:

$$d_{\text{norm}} = \frac{d_i - \text{midrange}}{\text{range} / 2}$$

where d_i is the i th input data value. The midrange and range are calculated as

$$\begin{aligned} \text{midrange} &= \frac{1}{2} (\max_{di}(d_i) + \min_{di}(d_i)) \\ \text{range} &= \max_{di}(d_i) - \min_{di}(d_i) \end{aligned}$$

where $\max_{di}$ and $\min_{di}$ are the maximum and minimum values of the transfer function range. For the *log* and *tanh* sigmoid functions, the $\max_{di}$ and $\min_{di}$ values are [0, 1] and [-1, 1] respectively. The normalized d_{norm} data values will then match the range of the activation function.

4.3. Neural Network Model

To verify the most suitable neural network for delay series prediction, we implemented predictive BPNNs, TDNN's, NARXNN's and RBFNN's. As mentioned before, we also used permutations of delay with packet loss and network throughput as additional inputs to the network to determine their correlation and impact on precise delay prediction. We denote the BPNNs, TDNN's, NARXNN's network topologies by $NET[d,t,p]i,h$, where NET is the neural network type as represented by BPNN, TDNN etc., d is the delay input to the network, t is the throughput input to the network, p is the packet

loss input to the network, i is the number of nodes at the input layer or tapped delay line inputs for the delay and NARX neural network models, h is the number of nodes in the hidden layer. The RBFNN networks are denoted by $NET[d,t,p]i,s$, where s is the spread factor. We have specifically not included the output layer node, all the neural network models used for this research have just one output node. As it is known in the literature that one hidden layer gives good results on prediction [7] and added hidden layers introduce increased complexity, control or real time predictions require optimal performance with reduced complexity hence we limit our hidden layer to one. We implemented several different topology configurations for the neural network types with just one hidden layer. The several topologies were determined by increasing the number of input and hidden nodes in steps of one. The maximum number of input layer and hidden layer nodes is 10. We used the early stopping method during network training to determine the best validation performance for each test sequence and its generalization capacity. Our data sets consisting of 1000 data samples have been partitioned into sets for training, validation and test with a ration of 70%, 15% and 15% respectively. There is no specific rule for the splitting of the data set. However, it is generally agreed that larger proportions of the data set should be used for training. A total of 100 neural network combinations was trained, validated and tested four times. The average of the root mean square error (RMSE) for the validation phase and regression results of the validation and testing phases recorded. For the RBF network, up to 10 inputs were tried with different spread factors ranging from 0.001 to 3. The number of hidden nodes required to meet the error target function was determined during the training process.

5. Experimental Analysis

The different neural network algorithms namely the BPNN, TDNN, NARXNN and RBFNN provided by Matlab Neural Networks package [14] were tried. The prediction results and the training performances of all the neural network models detailed below.

5.1. ANN Results

The Tables 1 to 4, shows the regression results of the testing phase *i.e.*, the correlation of the neural network output to the actual output during the testing phase, recall that the testing phase is when a data set is used to test the neural network performance or prediction capability.

Table 1. BPNN topology regression performance (i = nodes in the input layer, h = nodes in the hidden layer).

i/h	1	2	3	4	5	6	7	8	9	10
1	0.99771	1	0.99394	0.99972	0.99749	0.99049	0.99763	0.99025	0.94882	0.97005
2	0.99782	0.99995	0.99831	0.99999	0.99853	0.99992	0.99982	0.99931	0.99968	0.99995
3	0.99662	0.99202	0.99778	0.99979	0.99942	0.97385	0.99937	0.99979	0.99946	0.99974
4	0.99557	0.99871	0.99932	0.9991	0.99824	0.99951	0.98813	0.85575	0.99998	0.99601
5	0.99508	0.99704	0.99762	1	0.96259	1	0.99953	1	0.99997	0.99992
6	0.99994	0.93792	0.99997	0.99947	0.99947	1	0.99986	1	0.99795	0.99265
7	0.9844	0.99933	0.99963	0.96674	0.99967	0.9999	1	0.9992	1	0.97413
8	0.99211	0.99725	0.99997	0.9994	0.99521	0.99939	1	0.99997	1	1
9	0.99765	0.99261	0.9998	0.99879	0.99999	0.99996	1	0.99664	1	1
10	0.99704	0.99232	1	0.96654	1	0.99964	1	0.99975	0.99985	0.99999

Table 2. TDNN topology regression performance (i = nodes in the tapped delay line, h = nodes in the hidden layer).

i/h	1	2	3	4	5	6	7	8	9	10
1	0.99713	0.99749	0.99333	0.99642	0.99546	0.99684	0.98664	0.9949	0.98915	0.99394
2	0.99842	0.99254	0.99948	0.99999	0.98435	0.98885	0.98885	0.99356	0.99418	0.99646
3	0.99877	0.98948	0.99619	0.99864	0.9991	0.99993	0.99658	0.99352	0.92443	0.99786
4	0.99929	0.95056	0.99799	0.99977	0.9938	0.99777	0.98805	0.98998	0.99	0.99886
5	0.99657	0.99961	0.99999	1	0.99651	0.99974	0.99978	0.9945	0.98714	0.97938
6	0.99995	0.99429	0.99905	0.99656	0.99473	0.97471	0.99848	0.99858	0.99783	0.97157
7	0.99928	0.99209	0.99997	0.99989	0.99886	0.99989	0.99982	0.9998	0.99981	0.99792
8	0.99665	0.99401	0.99963	0.92658	1	0.9993	0.98412	0.99956	1	0.99444
9	0.98997	0.99735	1	1	0.98889	0.99964	0.99988	0.99742	0.99547	0.82498
10	0.99754	0.99985	0.9856	0.99975	0.99806	0.99998	0.99994	0.99602	0.99823	0.84556

Table 3. NARXNN topology regression performance (i = nodes in the tapped delay line, h = nodes in the hidden layer).

i/h	1	2	3	4	5	6	7	8	9	10
1	0.67563	0.70139	0.16444	0.62261	0.09675	0.37677	0.78767	0.34238	0.28678	0.222
2	0.71451	0.83411	0.73071	0.82169	0.63463	0.19166	0.6893	0.38353	0.14549	0.81205
3	0.69679	0.21447	0.78564	0.14763	0.59608	0.15593	0.4955	0.28168	0.04855	0.86435
4	0.47557	0.35143	0.60278	0.18833	0.24992	0.34382	0.60802	0.46657	0.26551	0.59912
5	0.7848	0.76433	0.63474	0.67272	0.44231	0.09158	0.62777	0.54616	0.65428	0.61139
6	0.62871	0.55258	0.56243	0.75613	0.47594	0.15447	0.64026	0.78239	0.24806	0.7588
7	0.78214	0.6706	0.48063	0.64128	0.04806	0.44459	0.78072	0.86537	0.49045	0.36757
8	0.71624	0.78585	0.09391	0.35222	0.65947	0.74706	0.81969	0.64936	0.77512	0.06884
9	0.85177	0.63294	0.78136	0.65603	0.68281	0.62178	0.29884	0.11818	0.04215	0.06341
10	0.76553	0.37205	0.51179	0.37349	0.73053	0.53513	0.50677	0.90871	0.14541	0.71483

Table 4. RBFNN topology regression performance (i = nodes in the input layer, s = spread factor).

i/s	0.01	0.05	0.1	0.25	0.50	0.70	0.90	1	2	3
1	0.96164	0.96164	0.99832	0.99969	0.99992	0.99996	0.99998	0.99998	1	1
2	0.96129	0.9563	0.9563	0.9994	0.99984	0.99992	0.99995	0.99996	0.99999	1
3	0.96059	0.99219	0.99634	0.99913	0.99977	0.99988	0.99993	0.99994	0.99999	0.99999
4	0.96233	0.98208	0.99677	0.99887	0.99969	0.99984	0.9999	0.99992	0.99998	0.99999
5	0.96303	0.97637	0.99814	0.99862	0.99962	0.9998	0.99988	0.9999	0.99998	0.99999
6	0.96166	0.97367	0.99861	0.99838	0.99954	0.99976	0.99986	0.99988	0.99997	0.99999
7	0.96171	0.97323	0.99661	0.99816	0.99947	0.99972	0.99983	0.99986	0.99997	0.99998
8	0.96147	0.96763	0.99393	0.99794	0.9994	0.99969	0.99981	0.99984	0.99996	0.99998
9	0.9614	0.96758	0.98769	0.99774	0.99933	0.99965	0.99978	0.99982	0.99996	0.99998
10	0.96432	0.96758	0.98152	0.99755	0.99926	0.99961	0.99976	0.99981	0.99995	0.99998

For the four network types studied above, we chose the topology with best performance. The selected topologies are listed in Table 5 and are used for final analysis and the experiments with additional inputs namely packet loss and network throughput. The term ‘epochs’ used in Table 5 represents the number of network input sets (training sets) that was presented during the network training process prior to achieving a suitable error target.

Table 5. Best topologies performance summary.

Topology	Validation RMSE	Epochs	Validation Regression	Testing Regression
BPNN(d)6,8	4.49E-06	5	0.999999999	0.999999999
TDNN(d)8,5	2.25E-05	9	0.999999702	0.999999980
NARXNN(d)10,8	8.55E-05	7	0.858126729	0.908709757
RBFNN(d)1,3	1.00E-06	3	0.999998120	0.999997805

In Table 6, the summary of results with the best prediction using the additional inputs is listed. From Table 6, it shows that the combination of delay and network throughput sequences yields better results to a delay and packet loss combination for the selected best experimented neural network models.

Table 6. Performance of best topologies with added inputs.

Topology	Validation RMSE	Epochs	Validation Regression	Testing Regression
BPNN(dp)6,8	8.45E-04	6	0.999685553	0.999285754
BPNN(dt)6,8	2.34E-05	6	0.999994186	0.999996961
TDNN(dp)8,5	8.26E-04	9	0.9993673310	0.991370462
TDNN(dt)8,5	5.15E-04	9	0.9999325536	0.999974604
NARXNN(dp)10,8	3.97E-03	7	0.22702980098	0.81293043617
NARXNN(dt)10,8	1.09E-02	7	0.30071510804	0.93552375907
RBFNN(dp)1,3	5.03E-05	3	0.999999996	0.999999958
RBFNN(dt)1,3	1.98E-05	3	0.999999999	0.999999987

5.2. Prediction Results Analysis

The plots below show the prediction correlation results by the best neural network models selected. Each neural network type has two models with the same topology, which differ in the way they work, namely, one uses only the delay as its inputs to the network and the other uses delay and network throughput information as inputs. The best prediction performances for the BPNNs, TDNN's, RBFNN's networks show that any of the three models has acceptable prediction accuracy.

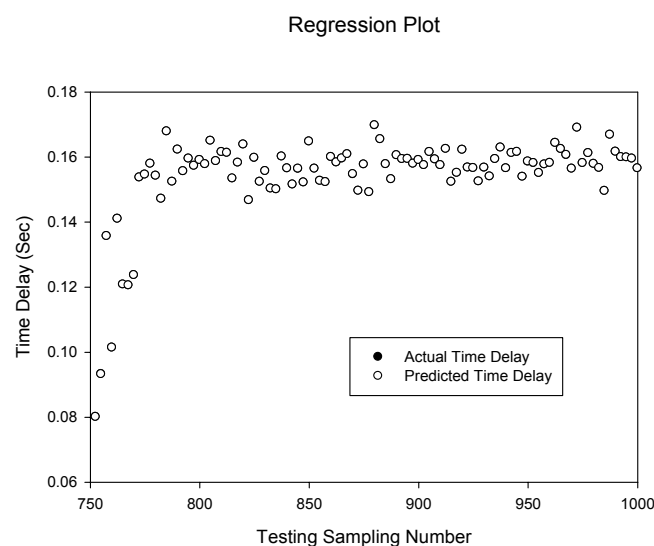
Figure 3. Prediction of delay sequence using $BPNN(d)_{6,8}$.

Figure 3 shows the prediction of the time delay sequences and gives a correlation of 0.999999999 by the $BPNN(d)_{6,8}$. It can be seen that prediction has a very high accuracy. The predicted values match the actual delay values. This high accuracy is also observed with the delay and network throughput combination as shown in Figure 4 for the $BPNN(dt)_{6,8}$ network which has a correlation of 0.999996961. However, this is slightly lower than the $BPNN(d)_{6,8}$ network without the additional network throughput input.

Figure 4. Prediction of delay sequence using $BPNN(dt)_{6,8}$.

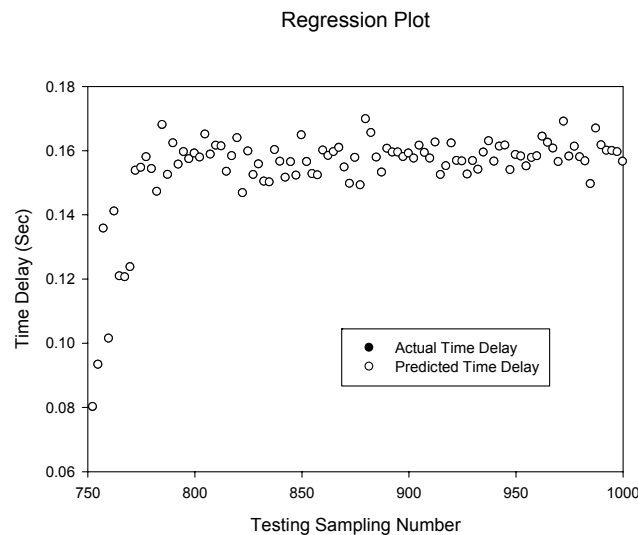


Figure 5 shows the prediction of the time delay sequence with correlation of 0.999999980 given by the $TDNN(d)_{8,5}$. It can be seen that the prediction accuracy is identical to the $BPNN(d)_{6,8}$ topology predictions. However, the TDNN model requires a larger number of hidden neurons and more training iterations to achieve the same prediction accuracy. From Figure 6, the $TDNN(dt)_{8,5}$ prediction accuracy is noticeably lower than the $TDNN(d)_{8,5}$. Compared to the previous two cases of the BPNN networks, the addition of the network throughput as an extra input is not significantly noticeable.

Figure 5. Prediction of delay sequence using $TDNN(d)_{8,5}$.

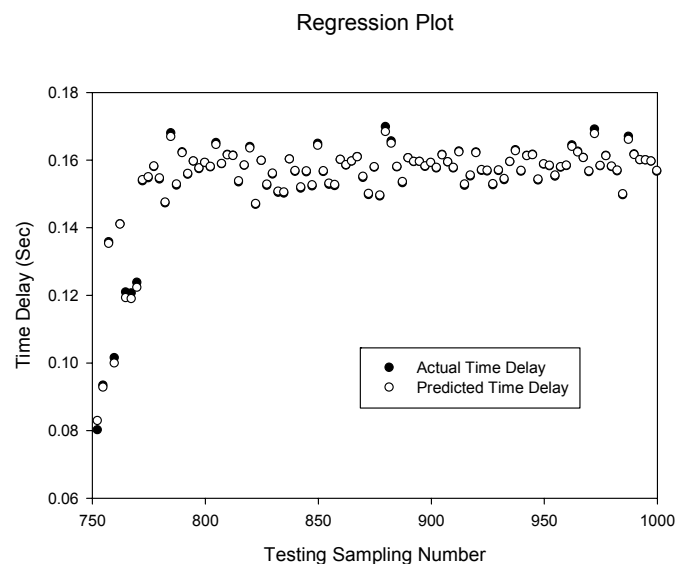
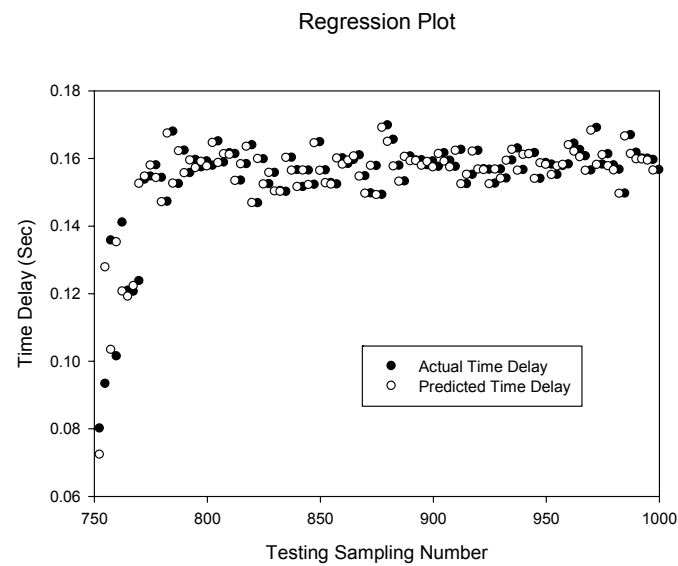


Figure 6. Prediction of delay sequence using $TDNN(dt)_{8,5}$.

From Figures 7 and 8 it can be seen that delay prediction using the NARXNN models lack adequate accuracy. The predicted values are around 10% deviation of the actual delay series. The prediction correlation for the delay $NARXNN(d)_{10,8}$ and $NARXNN(dt)_{10,8}$ delay plus network throughput combinations are 0.908709757 and 0.935523759 respectively. Interestingly, the delay plus throughput combination produces improved results in the NARXNN compared to the BPNN and TDNN models.

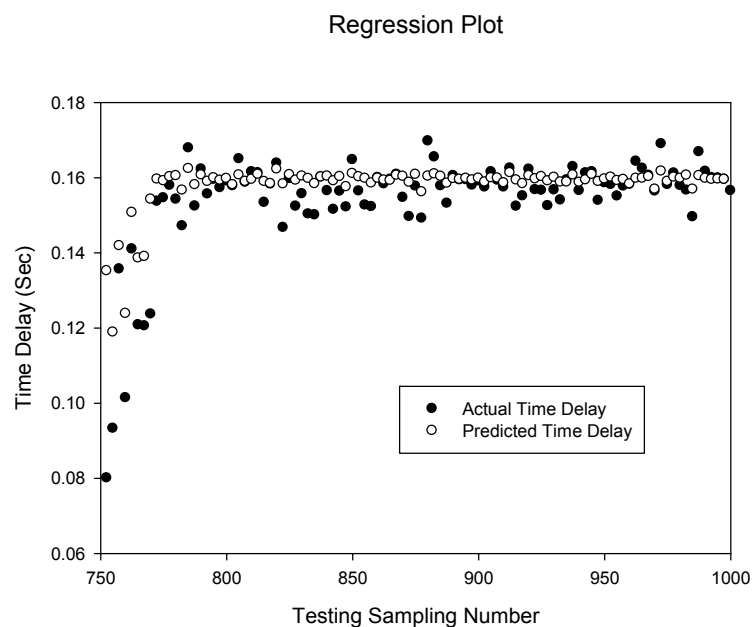
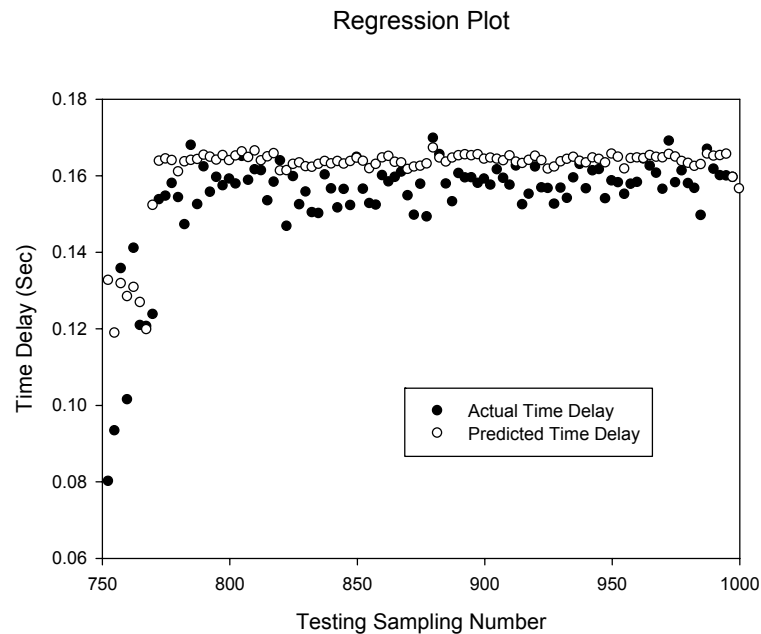
Figure 7. Prediction of delay sequence using $NARXNN(d)_{10,8}$.

Figure 8. Prediction of delay sequence using $NARXNN(dt)_{10,8}$.

Figures 9 and 10 show the prediction of the delay sequence with correlation of 0.999997805 and 0.999999987 given by the $RBNN(d)_{1,3}$ and $RBNN(dt)_{1,3}$ respectively. It can be seen that the $RBNN(dt)_{1,3}$ prediction accuracy is higher than the delay only input model $RBNN(d)_{1,3}$ compared to the previous case of the NARXNN neural network model. Generally the RBF networks have a high correlation and simulations suggest that increased spread factor yields better results.

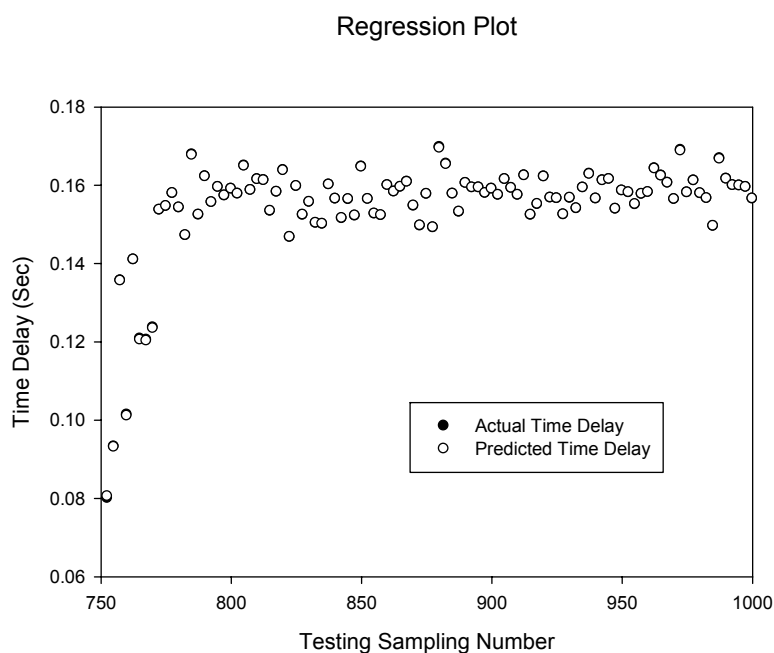
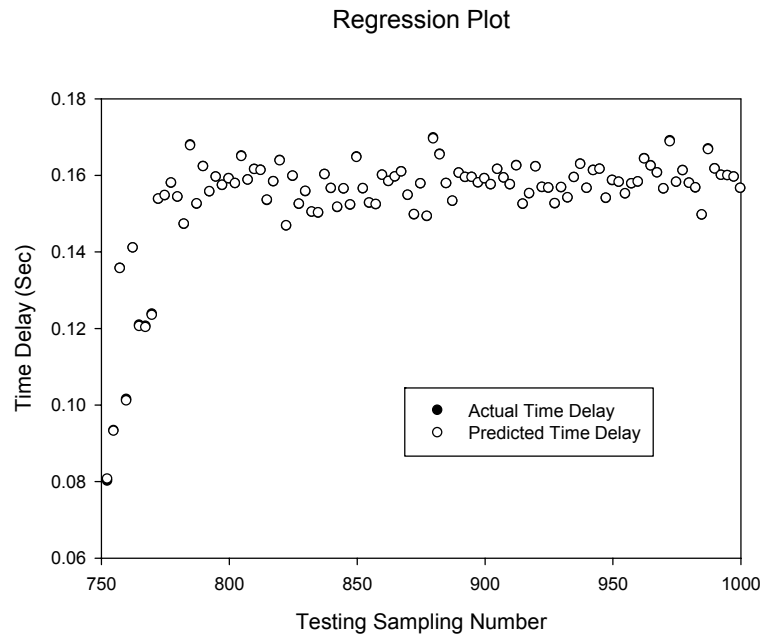
Figure 9. Prediction of delay sequence using $RBNN(d)_{1,3}$.

Figure 10. Prediction of delay sequence using $RBFNN(dt)_{1,3}$.

In summary, either the BPNNs, TDNN's, RBFNN's can achieve desired prediction accuracy for delay prediction in WSA's. The BPNN and TDNN networks needs more iterations (epochs) of training compared to an RBF network, while an RBF network needs more hidden neurons, *i.e.*, the RBF networks required 17 hidden nodes which is double the size required compared to the BPNN and TDNN networks. This means that an RBF network is capable of faster learning compared to the BPNN and TDNN networks, but an RBF network needs more memory space for the network. The NARXNN network simulations did not yield acceptable results hence not used in our results comparison. As expected, the combination of the delay inputs with the network throughput inputs produced better results than the combination with the network packet loss rate. This shows that the network throughput is correlated with the delay experienced in wireless networks.

Comparing the results of the eight best sequences of the test results, we want to identify the best configuration capable of predicting delay sequences in wireless personal area networks for control applications. Based on these results, the appropriate topology for this is the topology with the best correlation and least network computing requirements which is the $BPNN(d)_{6,8}$.

5.3. Inverted Pendulum Simulations

We integrate the proposed neural network to the control models and use the inverted pendulum on a cart system to test them to determine their effects and system response. The inverted pendulum is one of the most common systems used to test control system algorithms. Our simulations are done in the Matlab programming environment using the Simulink inverted pendulum simulation tool [14]. We also assume that there is no packet loss in our experiments.

Without control, the inverted pendulum is unstable. To keep the inverted pendulum stable, a force F is applied to the cart to keep the pendulum in an upright position. The dynamics of the system are nonlinear. The coordinate of the cart's position is denoted by x and the angular position of the

pendulum denoted by θ . For this pendulum system, the control input is the force F that controls the cart in the horizontal plane. The system outputs are the angular position of the pendulum θ and the horizontal position of the cart x . The equation of motion used for the pendulum is determined by the constraints placed on its motion. The equations of motion for the pendulum can be thus described.

With the following notations: x = cart position, θ = pendulum angle, F = applied force, M = mass of cart, m = mass of pendulum bob, l = length of pendulum, g = gravitational constant. The dynamics of the system can be derived from the Lagrange equations [21].

$$\frac{d}{dt} \left[\frac{\partial}{\partial \dot{x}} L \right] - \frac{\partial}{\partial x} L = F$$

$$\frac{d}{dt} \left[\frac{\partial}{\partial \dot{\theta}} L \right] - \frac{\partial}{\partial \theta} L = 0$$

where L represents the Lagrangian function. $L = K - V$ where K and V are the kinetic and potential energies of the system respectively. The kinetic energy of the cart is defined as

$$K_1 = \frac{1}{2} M \dot{x}^2$$

The kinetic energy of the pendulum bob is

$$K_2 = \frac{1}{2} m (\dot{y}^2 + \dot{p}^2)$$

where p is the distance between the bob and the base of the cart

$$y = x + l \sin \theta, p = l \cos \theta$$

$$\dot{y} = \dot{x} + l \dot{\theta} \sin \theta, \dot{p} = -l \dot{\theta} \cos \theta$$

The total kinetic energy is defined as

$$K = K_1 + K_2 = \frac{1}{2} M \dot{x}^2 + \frac{1}{2} m (\dot{x}^2 + 2 \dot{x} \dot{\theta} l \cos \theta + l^2 \dot{\theta}^2)$$

Substituting for L in the lagrange equations and performing partial differentiation we have

$$F = (M + m) \ddot{x} + ml \ddot{\theta} \cos \theta - ml \dot{\theta}^2 \sin \theta$$

$$0 = ml \ddot{x} \cos \theta + ml^2 \ddot{\theta} - mgl \sin \theta$$

Solving for $\ddot{x}$ and $\ddot{\theta}$ and performing linearization to obtain the linear state equations (the reader is referred to [21]) we have

$$\ddot{x} = -\frac{mg}{M} \theta + \frac{F}{M}$$

$$\ddot{\theta} = -\frac{(M + m)g}{Ml} \theta - \frac{F}{Ml}$$

These equations yield the linear state space form

$$\dot{x} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{(M+m)g}{Ml} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ \frac{1}{Ml} \end{bmatrix} u = Ax + Bu$$

The output equation depends on the measurements to be taken. Our outputs are the cart position x and angular position of the pendulum θ . The output equation is then

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

Our inverted pendulum system parameters are listed below in Table 7. We assume that the pendulum is linearized around the vertical position *i.e.*, it does not move more than few degrees.

Table 7. Inverted pendulum parameters.

Mass of cart, M	3 kg
Mass of pendulum bob, m	1.5 kg
Length of pendulum, l	0.7 metres
Gravitational constant, g	9.81

Substituting the inverted pendulum system parameters defined in Table 7 into the state space equation we have

$$\dot{x} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -4.9050 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 21.0214 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.3333 \\ 0 \\ 0.4762 \end{bmatrix} u = Ax + Bu$$

Looking at the eigen values of the system matrix A, we can verify the stability of the dynamic system. The calculated eigen values of A are [0, 0, 4.5849 −4.5849] and from this we can see that the system is unstable. We apply the linear quadratic control (LQR) methodology. The Q weights represent the different parts of the state. To weight the cart's position and the pendulums angle, the elements in the 1,1 position and 3,3 position is used respectively. The R matrix represents the input weighting. Since we are interested in maintaining the pendulum in an upright position, we drive the pendulum position state towards zero by inputting a larger entry in the corresponding part of the system state. After trial and error, we selected the weight matrices Q and R as:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 15 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } R = [1]$$

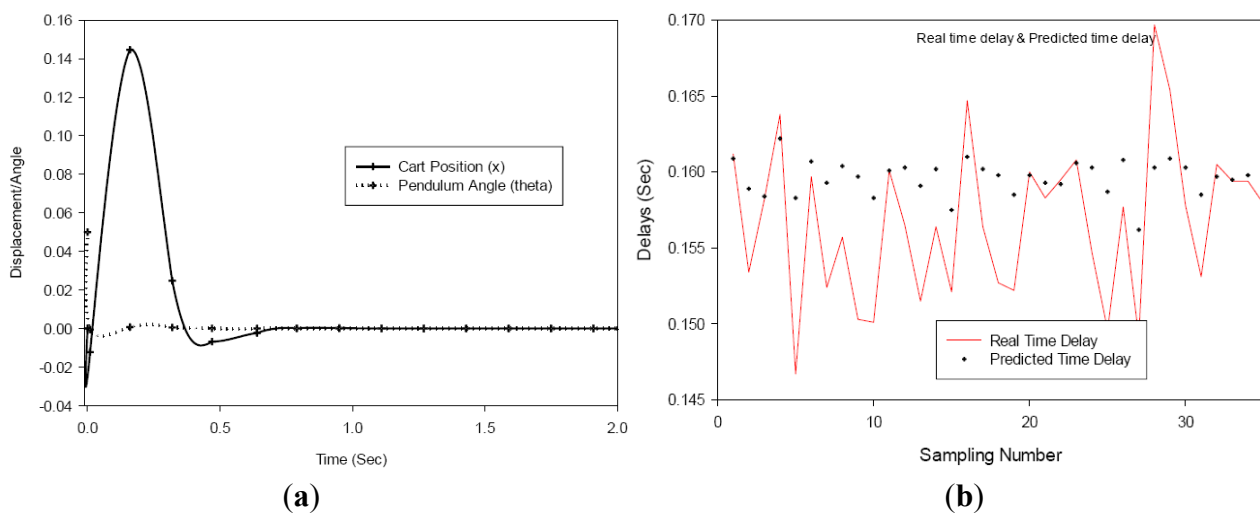
The resultant desired state feedback gain K matrix used is [0.4023 −1.0747 72.3102 15.5374].

Variable Sampling Period

Consider the control system with delays using the same feedback gain and the observer based model. To simulate the networked circumstance, the neural network $BPNN(d)_{6,8}$ predicted delay sequence is used which is the predicted value at sampling step k of the network induced delay one-step ahead. For the observer model, the system observer is sampled at every predicted total delay. First we investigate the case of variable sampling with the predicted delays by the $NARXNN(d)_{10,8}$ as the sampling periods.

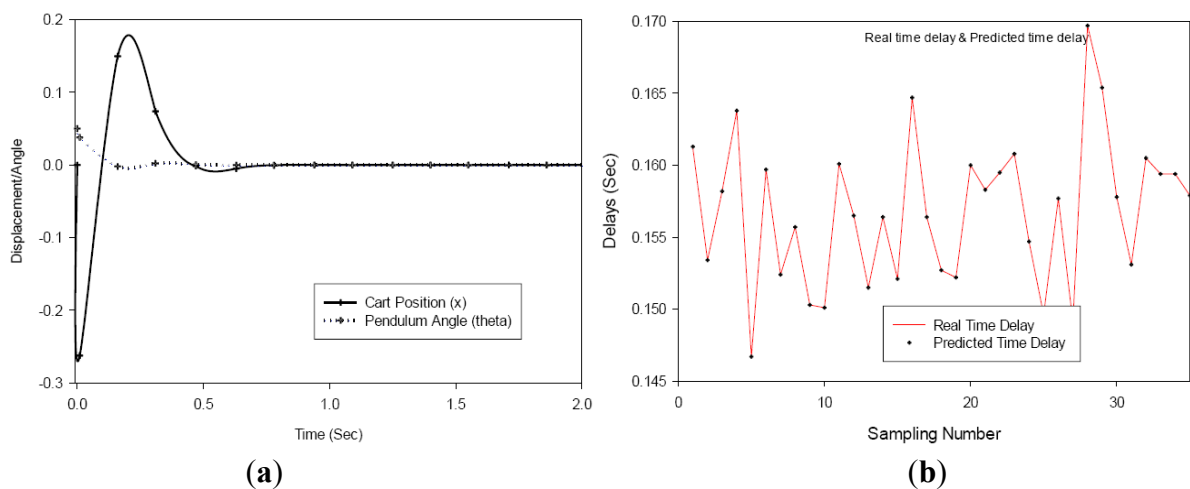
In Figure 11 it shows that the pendulum converges to the goal about 0.804 s and the cart returns to its original position is 0.95 s with a maximum distance excursion of 0.16 to -0.01 m.

Figure 11. (a) Variable sampling with inadequate delay prediction (b) Real vs. predicted time delay.



We also apply the $BPNN(d)_{6,8}$ predicted delay values to the variable sampling observer. The initial position of the pendulum in the observer was set to $\theta = 0.05\text{rad}$, same as the actual plant condition. After trial and error we set the observer gain to $[-0.3 \ -0.31 \ -0.32 \ -0.33]$.

Figure 12. (a) Variable sampling with adequate prediction (b) Real vs. predicted time delay.



From the observer based variable sampling results in Figure 12, the position of the pendulum is controlled adequately and the pendulum converges to its goal in about 0.64 s, with the maximum excursion of the cart 0.154 m and reaching steady state in 0.84 s. The results with the better predicted values by the $BPNN(d)_{6,8}$ produced better results than the $NARXNN(d)_{10,8}$ predicted delay values and points to the need for accurate prediction of the delays by the neural network.

6. Conclusion

Self organizing and adaptive networks is one of the desirable features of wireless networks for control applications to eliminate human intervention and guarantee desired QoS in undesirable network conditions. In this paper, we have highlighted the importance and challenges of reliable data transmission and real time communications in WSNs for control applications. From the ANN simulation results, it can be seen that, determining the appropriate topology for WSN delay series prediction requires several iterations. From our test results, we can conclude that the backpropagation neural network with six inputs and eight hidden layers produced the best results during training, validation and testing. From the simulations, we can also conclude that the network throughput input is more correlated to the delay experienced in wireless networks than the network packet loss rate hence provides better results when used as an extra input in training the network. However, the results did not prove better than the networks without the added input. This may require that several topologies be tested to determine the most suitable topology with added inputs.

The adequate and inadequate neural network predicted delay series are used to simulate the effect of delays. The variable sampling tests using the cases of predicted values highlights the importance of accurate prediction as an important factor in variable sampling. Networked control simulations indicated that adequate prediction of the delay series influences the performance of the variable sampling control system. Future work includes assessing and comparing how the results of the various neural network models will generalize to an independent time delay series data set. This will involve the use of the Dietterich's 5×2-Fold cross-validation paired T-test or Alpaydin's 5×2-fold cross-validation paired F -test [22].

References

1. Nkwogu, D.N.; Allen, A.R. Adaptive Learning and Reconfiguration in Wireless Sensor networks for Control Applications. In *Proceedings of the 12th Annual Postgraduate Symposium on the Convergence of Telecommunication, Networking and Broadcasting*, Liverpool, UK, 27–29 June 2011.
2. IEEE Standard for Information Technology—Telecommunications and Information Exchange between Systems—Local and Metropolitan Area Networks—Specific Requirement Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs), IEEE Std 802.15.4a-2007.
3. Gmez, R.C.L.; Velasco, P.M.; Fuertes, J.M. *Wireless Network Delay Estimation for Time Sensitive Applications Research Report*; ESAII RR-06-12; Technical University of Catalonia, Barcelona, Spain, 2006.

4. Nilsson, J. Real-Time Control Systems with Delays. Ph.D Thesis, Lund Institute of Technology, Lund, Sweden, 1998.
5. Kurose, J.F.; Ross, K.W. *Computer Networking: A Top-Down Approach*, 4th ed.; Addison Wesley, 2007.
6. Nise, N.S. *Control Systems Engineering*; Benjamin/Cummings: San Fransisco, CA, USA, 1992.
7. Yi, J.; Wang, Q.; Zhao, D.; Wen, J. BP neural network prediction based variable sampling approach for networked control systems. *Appl. Math. Comput.* **2007**, *185*, 976–988.
8. Kawka, P.; Alleyne, A. Stability and Feedback Control of Wireless Networked Systems. In *Proceedings of the 2005 American Control Conference*, Portland, OR, USA, 8–10 June 2005, Volume 4, pp. 2953–2959.
9. McKernan, A.; Arino, C.; Irwin, G.; Scanlon, W. A Multiple Observer Approach to Stability in Wireless Network Control Systems. In *Proceeding of the UKACC International Control Conference*, Manchester, UK, 2008.
10. Branicky, M.; Phillips, S.; Wei, Z. Stability of Networked Control Systems: Explicit Analysis of Delay. In *Proceedings of the 2000 American Control Conference*, Illinois, IL, USA, June 2000; Volume 4; pp. 2352–2357.
11. Osterlind, N.; Dunkels, A. Approaching the Maximum 802.15.4 Multi-Hop Throughput. In *Proceedings of ACM HotEmNets '08*, Charlottesville, Virginia, USA, 2–3 June 2008.
12. Latré, N.; De Mil, P.; Moerman, I.; Van Dierdonck, N.; Dhoedt, B.; Demeester, P. Maximum throughput and minimum delay in IEEE 802.15.4. *Lect. Note. Comp. Sci.* **2005**, *3794*, 866–876.
13. Haykin, S. *Neural Networks: A Comprehensive Foundation*; Macmillian College Publishing: New York, NY, USA, 1994.
14. MATLAB® & Simulink® Student Version R2011 by The MathWorks, Inc. Matlab reference © Copyright 1984–2011
15. Xia, F.; Ma, L.; Peng, C.; Sun, Y.; Dong, J. Cross layer adaptive feedback scheduling of wireless controls systems. *Sensors* **2008**, *8*, 4265–4281.
16. Guo, Z.; Malakooti, B. Delay Prediction for Intelligent Routing in Wireless Networks Using Neural Networks. In *Proceedings of IEEE International Conference on Networking, Sensing and Control (ICNSC06)*, Ft. Lauderdale, FL, USA, 23–25 April 2006.
17. Sadeghzadeh, N.; Afshar, A.; Menhaj, M. An MLP Neural Network for Time Delay Prediction in Networked Control Systems. In *Proceedings of Chinese Control and Decision Conference*, Yantai, China, 2–4 July 2008.
18. Zhang, L.; Shi, Y.; Chen, T.; Huang, B. A new method for stabilization of networked control systems with random delays. *IEEE Trans. Automatic Control* **2009**, *50*, 1177–1181.
19. Nilsson, J.; Bernhardson, B.; Wittenmark, B. Stochastic analysis and control of real-time systems with random time delays. *Automatica* **1998**, *34*, 57–64.
20. OpnetModeler. Available online: https://enterprise1.opnet.com/opnet_directory/login/validate_static_pages?REDIRECT_URL=https://www.opnet.com/support/110-opnetreference (accessed on 28 March 2012).
21. Friedland, B.; McKendrick, J.A. *Control System Design an Introduction to State-Space Methods*; McGraw-Hill: New York, NY, USA, 1987.

22. Alpaydin, E. *Introduction to Machine Learning*, 2nd ed.; The MIT Press Cambridge: Cambridge MA, USA, 2010.

© 2012 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/3.0/>).