*Computers and Chemical Engineering*, **in press,**

**DOI: 10.1016/j.compchemeng.2012.01.002.**

# Predictive compensation for variable network delays and packet losses in networked control systems

Guo-Song Tian, Feng Xia [1] and Yu-Chu Tian [2]

*School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane QLD 4001, Australia*

**Abstract**

Networked control systems (NCSs) offer many advantages over conventional control; however, they also demonstrate challenging problems such as network-induced delay and packet losses. This paper proposes an approach of predictive compensation for simultaneous network-induced delays and packet losses. Different from the majority of existing NCS control methods, the proposed approach addresses co-design of both network and controller. It also alleviates the requirements of precise process models and full understanding of NCS network dynamics. For a series of possible sensor-to-actuator delays, the controller computes a series of corresponding redundant control values. Then, it sends out those control values in a single packet to the actuator. Once receiving the control packet, the actuator measures the actual sensor-to-actuator delay and computes the control signals from the control packet. When packet dropout occurs, the actuator utilizes past control packets to generate an appropriate control signal. The effectiveness of the approach is demonstrated through examples.

*Key words:* Process control; networked control systems (NCSs); network-induced delay; packet loss; predictive compensation

[1] Current address: School of Software, Dalian University of Technology, Dalian 116024, China.

[2] Corresponding author: Y.-C. Tian. Phone: +61 7 3138 2177, fax: +61 7 3138 2703. Email: y.tian@qut.edu.au.

## 1 Introduction

Traditional computer control systems use point-to-point architecture to interconnect control system components including sensors, controllers and actuators. With the increasing integration and complexity of modern industrial processes, the point-to-point architecture faces difficulties in many aspects such as poor scalability and costly maintenance. This has motivated significant development of computer network technologies for industrial process control in recent years. As a result, the point-to-point architecture has been gradually replaced by network control systems (NCSs). Research and development of NCSs have received considerable attention particularly in industrial process systems (El-Farra and Mhaskar, 2008; Sun and El-Farra, 2008) because of many benefits they can offer, e.g., good scalability and flexibility, easy maintenance and installation, and low cost. Recent advances in NCS can be found in Antsakls and Baillieul (2007); Baillieul and Antsaklis (2007); El-Farra and Mhaskar (2008); Gupta and Chow (2010); Marti et al. (2005); Nilsson (1998) and references therein.

Fig. 1 shows a typical NCS, which is a distributed feedback control system. In an NCS, control loops are closed via a communication network (Marti et al., 2005). All data transmissions between sensors, controllers, actuators, and other system components are implemented over the NCS network. In a control period of an NCS, the sensors sample the plant and then send the sensed data to the controller via network communications. When the sampled data are received by the controller, the controller computes the control signals and send out the control signals to the actuators via the NCS network. Once receiving the control signals, the actuators output the signals to the plant.

Introducing wired and/or wireless networks into control systems cause challenging problems. Two of those challenging problems are time-varying network-induced delays and packet losses (Tian and Levy, 2008a,b). The network delays result from the sharing of the NCS network resources by multiple nodes in which only one node is allowed to transmit its packet per transmission; while the packet losses are mainly due to unreliable network communications (Baillieul and Antsaklis, 2007; Nilsson, 1998; Tipsuwan and Chow, 2003).

There are many industrial processes whose dynamics are slow enough and thus network-induced delays are negligible. In this case, compensation for the network-induced delays is not always needed. However, network-induced delays becomes significant in plenty of other industrial processes with fast dynamics, justifying the need for delay compensation. A typical example is thermoplastic injection molding processes (Peng, 2007; Tian and Gao, 1999). Other typical examples include fluid flow processes and motor control, which are widely deployed in process industries.

The performance of an NCS may deteriorate significantly due to time-varying

network-induced delays and packet losses for those processes with fast dynamics, particularly when the traffic load of the NCS network is heavy. As shown in Fig. 1, due to the sensor-to-controller delay, the controller input values do not represent the instantaneous and actual plant status at the exact sampling time instance. Consequently, the controller uses outdated information in its algorithm execution, hence generating outdated control signals. This becomes even severer because of the controller-to-actuator delay. Moreover, the control signals may not be fully implemented because of packet losses along the way from the sensor to the controller and then to the actuator. Therefore, these challenging problems of network-induced delay and packet losses may lead to significant deterioration of the overall NCS performance and even cause system instability, particularly in time-sensitive control applications (Cervin et al., 2003; Decotignie, 2005; Nilsson, 1998). In addition, they also result in difficulties in NCS analysis and design (Tian and Levy, 2008b).

So far, the majority of NCS research have focused on controller design to provide sufficient stability margin in the presence of time-varying network-induced delays and packet losses (Gupta and Chow, 2010; Heemels et al., 2010). Significant work has also been reported on modelling network-induced delays and packet losses in a unified NCS model (Gupta and Chow, 2010; Heemels et al., 2010; Peng and Tian, 2009; Tipsuwan and Chow, 2003; Zou et al., 2010). In the sense of providing sufficient system stability guarantee, this type of methodology is effective in compensation for time delays and packet losses and in analysis of NCS stability and transient responses. However, the control actions derived from this type of methodology are generally conservative; and precise plant modelling, which may not be available for many industrial processes (Tian et al., 2003), is also an essential requirement (Heemels et al., 2010; Martins and Jota, 2010). As an example, the optimal stochastic control methods developed for an NCS (Koivo and Reijonen, 2004) are based on the assumption that the network-induced delay is constant or state-dependent. But it is not clear how to fit time-varying delays of an NCS, e.g., delays from random access networks (Gupta and Chow, 2010; Martins and Jota, 2010), into such methods requiring constant delays. Therefore, implementation of these controller-based solutions becomes difficult, if not impossible, with the increase in the complexity of the controller design and/or communication network modelling.

Recently, effort is being made to develop appropriate co-design NCS methodologies that consider both network Quality-of-Service (QoS) and Quality-of-Control (QoC) together (Branicky et al., 2003; Gabel and Litz, 2004; Tian and Levy, 2008b). The sampling period scheduling is a typical co-design methodology that shapes the traffic load of an NCS by regulating sampling periods of the control loops in respect of measured QoS parameters (Colandairaj et al., 2007; Gabel and Litz, 2004; Martins and Jota, 2010; Sala et al., 2009; Tipsuwan and Chow, 2003). Implementation of this method is based on the precise estimate of the NCS network condition; so difficulties arise in an NCS with unknown traffic pattern or variable networks architecture.

Another typical NCS co-design methodology is packet-based control (Chaillet and Bicchi, 2008; Zhao et al., 2009a). It is believed that in most communication networks, sending a single bit or several hundred bits in a packet consumes almost the same amount of network bandwidth (Hespanha et al., 2007; Zhao et al., 2009a). This makes it technically viable for an NCS controller to actively send a sequence of control predictions in one packet. The actuator selects an appropriate control value from multiple ones according to current network condition. In this way, the network-induced delay can be compensated. Due to the active delay compensation in the packet-based control approach, better control performance can be expected than that from the control methods that compute only one control value.

Along this direction, Zhao et al. (2008) proposed a two-step predictive control scheme for NCSs with random delays and input nonlinearity. A theoretical stability result is also given for the NCSs. However, the theoretical result is for a constant delay only, different from the general assumption of random delays in the proposed scheme. The same group of authors (Zhao et al., 2009b) further extended the packet-based control approach to continuous-time case by using a discretization technique for continuous network-induced delay. But theoretical evidence was not provided to show that the dynamics of the discretized system are the same as those of the original continuous-time system. Recently, Xia et al. (2009) employed the idea of packet-based predictive control in NCSs with random delay and packet dropout. Zou et al. (2010) investigated network-based predictive control of multirate systems.

Despite recent progress in packet-based control of NCSs, existing packet-based control methods have some common problems that impede their implementation in practical applications. The first problem is that they all rely on precise process models. If a good process model is not available (Tian et al., 2003) or the process model is too complicated (Zhang et al., 2008), which is the usual case in modern industrial processes, the implementation of such methods developed for simple process models becomes questionable. The second problem is that some predictive model controllers and adaptive controllers have been proposed based on the assumption that the network characteristics are well understood in advance (Gabel and Litz, 2004; Martins and Jota, 2010; Tavassoli et al., 2009; Tipsuwan and Chow, 2003). If such an assumption does not hold, it is still difficult to design simple yet effective predictive control algorithms in an NCS with random network-induced delays. This motivates the research of this work for predictive control design that alleviates the requirements of precise process models and full understanding of the NCS network dynamics.

In the same category of packet-based control methods (Chaillet and Bicchi, 2008; Zhao et al., 2009a,b), the work presented in this paper proposes a new packet-based co-design approach to compensate for simultaneous network-induced delays and packet losses in an NCS. The network-induced delay considered in this work is the end-to-end network transmission delay along the way from the sensor through

controller to the actuator excluding the computational delay in the controller. The control computation delay as well as any delays from the actuator to the plant are lumped into the plant delay, and thus are addressed indirectly in the predictive control design. In the proposed approach, the NCS controller computes a series of predictive and redundant control values corresponding to possible end-to-end network-induced delays on a given time horizon. Then, it sends those control values in a single packet to the actuator. Once receiving the control packet, the actuator measures the actual sensor-to-actuator delay by using the time-stamping technique, and then computes the control signals from the series of redundant control values. In the presence of packet dropout, the queuing methodology developed by Tian and Levy (2008b) is incorporated into the proposed co-design approach to predict lost control signals from past control packets. The proposed packet-based approach has three useful features: co-design of network and control, compensation for simultaneous network-induced delay and packet losses, and loose dependence on precise process models and good understanding of NCS network dynamics. These features differentiate the approach from other packet-based methods.

The paper is organized as follows. Following this introductory section, Section 2 presents the NCS architecture considered in this paper. Section 3 describes our new co-design approach for predictive compensation for network-induced delays and packet losses. Algorithms for implementation of the proposed co-design approach are designed in Section 4. In Sections 5, 6 and 7, case studies are carried out to demonstrate the effectiveness of the proposed co-design approach. Finally, Section 8 concludes the paper.

## 2 System Architecture and Basic Assumptions

Various network technologies are available for NCS design, e.g., Fieldbus, Ethernet (or IEEE 802.3), and DeviceNet. Recently, contention-based Ethernet has been promoted for industrial process control applications (Decotignie, 2005; Marti et al., 2005; Tian and Tian, 2011). NCSs over contention-based networks will be considered throughout this paper.

In large-scale industrial applications, factory automation is usually implemented in multiple levels. For example, a typical two-level implementation of industrial control systems was discussed in Tian et al. (2006) with plant level and control and management level. The plant level is composed of sensors, actuators, and the plant to be controlled; and the control and management level consists of controllers, management computers, and other related components. This two-level system architecture is also adopted in this paper for development of predictive compensation for network-induced delays and packet losses over contention-based networks of an NCS. As shown in Fig. 2, sensors and actuators share a local area network (LAN) while controllers and management computers share another LAN in the NCS. Two

LANs are interconnected via a router and thus can exchange data through the network interconnection.

The following assumptions or settings are made for implementation of our proposed compensation method throughout this paper:

A1. The sensors in a control loop send measurement packets to the controller with a constant period, which is called control (or sampling) period. A constant sampling period can be guaranteed by clock-driven sampling and interrupt programming, and is a common practice in real industrial design and implementation.
A2. The sensors and actuators in a control loop are time-synchronized by using the time-stamping technique in order to deal with possible out-of-order packets and to measure end-to-end network-induced delays. Time synchronization protocols have been developed, e.g., Tian et al. (2008), to guarantee that all devices and components in an NCS are well synchronized.
A3. In each control period and for each control loop, the controller organizes predictive control values in a single control packet and sends this packet to the actuators. This assumption is common in existing packet-based methods, e.g., Chaillet and Bicchi (2008); Zhao et al. (2009a,b).
A4. The actuators in a control loop have some computational capacities to carry out simple computing; in addition, they also have a buffer to store predictive control values received from the controller. This is an inherent assumption in existing packet-based NCS control methods, e.g., Chaillet and Bicchi (2008); Zhao et al. (2009a,b).
A5. The control computation delay in the controller as well as any delays from the actuator to the plant are lumped into the plant delay. Existing packet-based methods, e.g., Chaillet and Bicchi (2008); Zhao et al. (2009a,b), have ignored those delays. However, those delays may be significant in some systems and thus are considered explicitly in our approach.

## 3 Structure of the Predictive Compensator

This section develops a general predictive compensator for simultaneous sensor-to-actuator delays and packet losses in an NCS. As mentioned in Section 1, it is understood that sending a single bit or several hundred bits in a packet consumes almost the same amount of network bandwidth in most data networks (Hespanha et al., 2007; Zhao et al., 2009a). This has inspired the development of packet-based control methods. The approach to be presented in this paper is also a packet-based control methodology.

In this approach, the sensors in a control loop send measurement packets with a constant transmission interval to the controller. From the measurement data, the

controller computes multiple predictive control values corresponding to a series of possible sensor-to-actuator delays in a fixed time horizon. The proposed approach can be integrated into any types of controllers, while a proportional-integral (PI) controller is adopted in this paper for easy demonstration of the approach and the PI controller settings are tuned corresponding to the possible delays.

The multiple predictive control values computed for the series of possible delays are sent to the actuator in a single packet at the next network access. Once received by the actuator, they are stored in the actuator before the next control packet is received, and are used to computer a single control value based on the actual sensor-to-actuator delay.

The actuator also maintains a list of past control values which have been actually output to the plant. In the presence of packet losses, these past control values will be re-used for estimating a control signal for packet loss compensation.

### *3.1 Workflow of the Predictive Compensator*

Fig. 3 shows the workflow of the proposed predictive compensation approach for network-induced delays and packet losses. In this figure, $T_k$ denotes the $k$th control period. As shown in Fig. 3, similar to the queuing methodology implemented in Tian and Levy (2008a,b), the proposed predictive compensation approach also uses two queues $Q_1$ and $Q_2$ for designing and implementing compensation for network-induced delays and packet losses, respectively. However, the structure of the two queues, the control values stored in the queues, and the compensation policies to compute the multiple control values to be stored in the queues are fundamentally different from those in Tian and Levy (2008a,b). This will become clear when the detailed design and implementation of the proposed approach are presented in the following.

In our approach, corresponding to a series of $n$ possible sensor-to-actuator delays, $\{u(k)_{d(1)}, u(k)_{d(2)}, \cdots, u(k)_{d(n)}\}$ in $Q_1$ are a sequence of $n$ predictive redundant control values computed by the controller and received by the actuator in a single control packet. The actuator also maintains a sequence of $m$ predictive control values $\{u(k)_{l(1)}, u(k)_{l(2)}, \cdots, u(k)_{l(m)}\}$ in $Q_2$. These $m$ predictive control values are computed by using past actual control signals output $\{u(k-1), \ldots, u(k-\phi\}$ to the plant by the actuator. Computed by the actuator either from $Q_1$ without packet dropout or from $Q_2$ in the presence of packet losses, $u(k)$ is the control signal actually output from the actuator to the plant in the $k$th control period.

As shown in Fig. 3, if a control packet arrives at the actuator by the deadline, the queue $Q_1$ is refreshed. Upon receiving the control packet, the actuator measures the actual sensor-to-actuator delay and computes the appropriate control signal by using the sequence of the predictive control values in $Q_1$. Once the control signal

is computed and output to the plant, it is pushed into the first-in-first-out (FIFO) queue $Q_2$, and then $Q_1$ is emptied. Therefore, $Q_2$ maintains a series of $m$ predictive control values. When a control packet is lost, the queue $Q_1$ is still empty, then the predictive control values in the queue $Q_2$ are utilized to compute an appropriate control signal to control the plant.

Recent studies have revealed that network-induced delays are distributed in a limited range in an NCS network (Tian and Levy, 2008a; Tian et al., 2007). In a well-designed NCS, the majority of the network-induced delays should be less than a predefined deadline $\sigma$. For NCS design, if a control packet is not received by this deadline, it is treated as a packet loss even if the packet arrives later.

The detailed design and implementation of the predictive compensator are developed below.

### *3.2 Compensation for Network-Induced Delay - Policy 1*

Network-induced delays are regarded as a factor to be added in different types of process models in recent NCS research (Gabel and Litz, 2004). Therefore, optimal controller parameters can be obtained corresponding to the possible network-induced delays. For example, for a first-order plus time delay process in an NCS, the controller can be tuned with consideration of the time delays by using appropriate tuning rules for improving the control performance (Aidan, 2006; Mikael et al., 2006; Tavakoli and Tavakoli, 2003). In this paper, a digital controller, which is discretized from a continuous-time one, is designed in control loops for practical controller implementation.

For a series of possible network-induced delays in the range of $[\tau_{min}, \tau_{max}]$, i.e.,

$$\tau_j \in [\tau_{min}, \tau_{max}] \ \ \forall j \in \{0, 1, \cdots, n-1\}, \tag{1}$$

where $\tau_j$ is the $j$th possible network-induced delay, the controller settings are adjusted and corresponding predictive control values are computed. The timelines of the policy for delay compensation are depicted in Fig. 4.

Fig. 4 shows the distribution of possible network-induced delays within the predefined deadline $\sigma$, which is set to be $2T_s$ in this paper. In this figure, $\tau_j$ represents the $j$th possible delay as explained above, and $\tau$ is the actual delay measured by the actuator through time-stamping. Corresponding to $\tau_j$, the controller parameter setting is denoted by $P_j$, which in general is a function of $\tau_j$. For $n$ possible network delays $\tau_0, \tau_1, \cdots, \tau_{n-1}$, there are $n$ sets of controller parameter settings $P_0, P_1, \cdots, P_{n-1}$. For example, for a PI controller, the controller setting $P_j$ consists of two parameters: the proportional gain $K_c$ and integral time $T_i$.

Using these control parameter settings, the controller computes $n$ predictive control values corresponding to the $n$ possible network-induced delays within the predefined deadline $\sigma$, and then sends this sequence of predictive control values to the actuator in a single control packet.

Once the actuator receives the control packet with a sequence of predictive control values, it refreshes queues $Q_1$ (Fig. 3) and measures the actual sensor-to-actuator network-induced delay. Then, from the $n$ predictive control values in the control packet stored in queue $Q_1$, an appropriate control signal $u$ is computed by the actuator based on the measured network-induced delay. The algorithm for the control signal computation in the actuator will be proposed later in Section 4.

### *3.3 Compensation for Packet Loss - Policy 2*

For packet losses in an NCS, the basic idea of the queuing methodology is adopted for development of a compensation scheme. The queuing methodology was originally proposed in Luck and Ray (1994) as a deterministic and predictor-based compensation technique. It has been re-developed recently in Tian and Levy (2008a,b) with a co-design feature and much simplified implementation.

The queuing mechanism utilizes some deterministic or probabilistic information of an NCS for control algorithm design. In the presence of a packet loss, the predicted control values in the $k$th control period can be computed by using a few past control values and their timestamps through the following general formulation

$$u(k)_{l(i)} = f(u(k-1), u(k-2), \ldots, u(k-\phi), t_{k-1}, t_{k-2}, \ldots, t_{k-\phi}),\ i = 1, \ldots, m, \quad (2)$$

where $u(k-1)$, $u(k-2)$, $\cdots$, $u(k-\phi)$ are past control values output from the actuator to the plant; $t_{k-1}$, $t_{k-2}$, $\cdots$, $t_{k-\phi}$ are their timestamps that the past control values are outputted to the plant; and $\phi$ is the number of past control values used in the prediction. As shown in Fig. 3, $m$ predictive control values are stored in $Q_2$.

It is noted that if a control packet is not received by the predefined deadline $\sigma$, it is treated as a packet loss even if the packet arrives later. Therefore, the delay compensation is event-triggered while the packet loss compensation is time-triggered. For example, if the actuator does not receive the control packet by the deadline $\sigma$, which is set to be $2T_s$ in this paper, the packet loss compensator is triggered and the actuator computes an appropriate control signal $u$ from the predictive control values ($u(k)_{l(0)}$, $u(k)_{l(1)}$ ,..., $u(k)_{l(m)}$) in $Q_2$. The detailed algorithm for packet loss compensation will be developed later in Section 4.

### *3.4 Other Settings of the Predictive Compensator*

Throughout this paper, the deadline $\sigma$ is set to be two sampling periods $2T_s$. Such a setting allows certain flexibility in controller design while still maintaining certain level of soft timeliness. Depending on the actual requirements in a control system, other values may be designed, e.g., $T_s$.

In the considered delay range $[0, \sigma]$, the number of predictive control points, $n$, also needs to be chosen. The bigger the value of $n$, the better predictive results would be derived while the more computing power would be demanded as well for both the controller and actuator. Heuristically, two time instants at the lower and upper bounds of the delay range should be chosen, and at least an additional time instant between the two bounds should be selected. We consider choosing three predictive points in each control period; and for the setting $\sigma = 2T_s$ in this paper, we have chosen $n = 5$.

As for the value of $m$, the number of predictive control values in the FIFO queue $Q_2$, Tian and Levy (2008a) have shown that three past control values and their performance would be enough to design a proportional-derivative (PD) packet lost compensator with good performance. Therefore, the value of $m$ could be chosen to be 3 to 5 in general.

## 4 Algorithm Design for the Predictive Compensator

### *4.1 Predictive Redundant Control Values*

Consider network-induced delays in a typical NCS scenario (Tian and Levy, 2008a; Tian et al., 2006, 2007). As mentioned earlier, the deadline $\sigma$ in Fig. 4 is set to be two control periods in this paper, i.e.,

$$\sigma = 2T_s \tag{3}$$

Now, consider $n$ possible network-induced delays within this predefined deadline, i.e., $\tau_0, \tau_1, \tau_2, \cdots, \tau_{n-1}$. Set

$$\begin{aligned} &\tau_0 = 0; \tau_{n-1} = \sigma; \\ &\tau_0 < \tau_j < \tau_{n-1} \ \forall \ j \in \{1, 2, \cdots, n-2\}. \end{aligned} \tag{4}$$

Next, corresponding to these $n$ possible network-induced delays, $n$ predictive control values $u(k)_{d(1)}$, $u(k)_{d(2)}$, $\cdots$, $u(k)_{d(n)}$ are computed using the $n$ respective sets

of controller parameters $\{P_j\}$, $j \in \{1, 2, \cdots, n\}$. While various methods have been developed to compute the controller parameter settings under certain performance index functions, simple yet effective controller tuning methods will be used in this paper as will be shown later in Sections 5, 6 and 7.

In is worth mentioning that the applicability of the proposed delay compensation approach does not reply on specific plant dynamics and controller design. However, for easy demonstration of the proposed delay compensation approach, PI and PID controllers are adopted in our case studies, which will be presented later in Sections 5, 6 and 7. Thus, well-developed PI and PID controller tuning rules, i.e., those in (Aidan, 2006; Åström and Hägglund, 1995; Skogestad, 2003), can be used to tune the controller parameters.

### *4.2 Control Signal for Delay Compensation*

Corresponding to $n$ possible network-induced delays within the predefined deadline, $n$ predictive control values are computed by the controller. Then, they are transmitted to the actuator in a single packet, and are stored in the queue $Q_1$ once received by the actuator.

Using the time-stamping information in the control packet, the actuator measures the actual sensor-to-actuator delay $\tau$. After that, a simple step algorithm can be implemented in the actuator to compute a single control signal from the $n$ predictive control values in the queue $Q_1$:

$$u(k) = \begin{cases} u(k)_{d(2)}, \ \tau_0 < \tau \leq \tau_1; \\ u(k)_{d(3)}, \ \tau_1 < \tau \leq \tau_2; \\ \ \ \vdots \\ u(k)_{d(n)}, \ \tau_{n-2} < \tau \leq \tau_{n-1}. \end{cases} \tag{5}$$

As an alternative to the above step algorithm shown in Eq. (5), another simple yet more accurate algorithm, a linear interpolation algorithm, can be adopted in the actuator to compute the control signal. If the actual sensor-to-actuator delay $\tau$ falls within $[\tau_{j-1}, \tau_j]$, the control signal $u$ is computed using a linear interpolation as:

$$u(k) = u(k)_{d(j)} + \frac{\tau - \tau_{j-1}}{\tau_j - \tau_{j-1}} \left[ u(k)_{d(j+1)} - u(k)_{d(j)} \right], \ j = 1, 2, \cdots, n-1. \tag{6}$$

It is worth mentioning that the linear interpolation in Eqn. (6) is carried out from the predicted control values stored in the queue $Q_1$. As it is not directly computed from the process models, it is applicable to control systems with either linear or nonlinear dynamics. The $n$ predictive control values can be estimated from process

models and the controller law if accurate process models can be established. They can also be derived using simple control strategies such as PI control when accurate process models are not available.

### *4.3 Prediction of Lost Packets*

In the presence of packet losses, a packet loss compensation scheme resulting from Eqn. (2) in employed. Several such functions have already been developed whose effectiveness has been demonstrated (Tian and Levy, 2008a,b). This paper incorporates the proportional-derivative (PD) method (Tian and Levy, 2008b) into our predictive compensation approach. To apply the PD method, it is necessary to achieve equal control period. By using a linear interpolation, we compute the first predictive control value as follows:

$$u(k)_{l(1)} = u(k) + \frac{(k+2)T_s - t(k)}{t(k) - t(k-1)}[u(k) - u(k-1)] \tag{7}$$

The other predictive control values for packet loss compensation can be computed by (Tian and Levy, 2008a,b):

$$u(k)_{l(i)} = u(k)_{l(i-1)} + KT_s u(k)^{(1)}_{l(i-1)},\ K \in [0,1] \text{ and } i = 2, \cdots, m \tag{8}$$

where the derivative gain ($K$) is set to 0.5 throughout this paper. $u(k)^{(1)}_{l(i-1)}$ is the first-order derivative of $u$ in the $(k-2)$th control period, and is approximated by:

$$\begin{aligned} u(k)^{(1)}_{l(1)} &= (u(k)_{l(1)} - u_{(k+1)T_s})/T_s, \\ u(k)^{(1)}_{l(i-1)} &= (u(k)_{l(i-1)} - u(k)_{l(i-2)})/T_s \ i = 3, \cdots, m \end{aligned} \tag{9}$$

where $u_{(k+1)T_s}$ represents the actual actuation signal at time $(k+1)T_s$. It is assumed that the actuator implements zero-order hold (ZOH) holding the last actuation signal until the next one arrives, so $u_{(k+1)T_s} = u(k)$.

In this PD scheme, a predictive control value is computed by using one-step ahead prediction through a proportional term plus a derivative term from past control signals. In this paper, $m$ predictive control values are stored in the queue $Q_2$ to enable the implementation of the packet loss compensation.

As described in Section 3.3, the packet loss compensation is time-triggered. In the presence of a packet loss from $k$th sampling instance, the control signal $u(k+2)$ that is outputted to the plant at $(k+2)$th sampling instance is set to $u(k)_{l(1)}$ and . If consecutive packet losses occur, $u(k+i) = u(k)_{l(i-1)}$ where $i = 3, 4, \ldots, m$.

## 5 Case Study Configurations

### *5.1 Simulation Environment*

For NCS simulations, both network dynamics and plant dynamics need to be modelled and simulated together (Branicky et al., 2003). Currently, there is a lack of such a simulation tool for comprehensive NCS modelling and simulations. While Matlab/Simulink and related toolboxes and packages, e.g., TrueTime, are good tools for simulation of traditional real-time control systems, they do not provide good support for simulation of various network protocols and complex network behaviours. For example, it is not easy to use Matlab to simulate an NCS with QoS-based sampling in which some network QoS measurements and time-stamping information are used online (Colandairaj et al., 2007).

The discrete-event network simulator NS2 (UCB/LBNL/VINT Groups, 2011) is a widely used software package for comprehensive modelling and simulation of network protocols and dynamics. However, it is not directly applicable to simulations of NCSs with continuous-time plant models. To solve this problem, we have recently developed an NS2 based hybrid NCS simulation environment by creating a scheduling synchronization mechanism as well as three types of control related agents: controller, actuator and sensor (Tian et al., 2009), which are in addition to many other existing built-in network agents in NS2. For simplicity, the actuator and sensor nodes have been uniformly described by "plant" nodes. Also, the time cost for executing control algorithms in the controller is set to be 0 in this paper as in other existing publications on packet-based NCS control, e.g., Chaillet and Bicchi (2008); Zhao et al. (2009a,b). If the actual control computation time delay is not negligible, it can be lumped into the plant model. The detailed implementation and usage of the scheduling synchronization mechanism and the new NS2 modules can be found in (Tian et al., 2009). This paper will use this hybrid NS2 simulation environment for case studies.

### *5.2 Simulation Procedures*

Our simulation procedures are discussed below. At the same time when network protocols, network traffic, and network dynamics are simulated, the following steps are undertaken to simulate the control system dynamics:

- Firstly, the plant/sensor node samples the plant state. The sampled data and corresponding time-stamping information are coded and sent to the controller in a single measurement data packet.
- Secondly, after receiving the measurement data packet, the controller node computes a sequence of $n$ predictive control values corresponding to a series of pos-

sible sensor-to-actuator delays within a predefined deadline $\sigma$, which is set to be $2T_s$ in this paper. Then, it sends these control values to the plant/actuator node in a single packet.
- Finally, the plant/actuator node receives the control packet, computes an appropriate control signal according to actual sensor-to-actuator delay, and updates the plant state. In the presence of packet dropout, the plant/actuator predicts the control signal from past control values stored in the queue $Q_2$ and and updates the plant state accordingly.

### *5.3 Network Architecture and Traffic Load*

Corresponding to the network structure described in Section 2, the network topology modelled in NS2 in this Section is shown in Fig. 5. The switch interconnects the plant local area network (LAN) and the controller LAN. Both LANs have been set with the same network capacity of 1Mbps.

As shown in Fig. 5, the pair of the controller node and plant node with combined sensor and actuator forms a control loop over a data network. It is noted that the "plant" node shares the network medium with four other nodes for other control loops in the same LAN. The controller also shares the network resources with four other nodes for other control tasks in the same LAN.

In order to validate the effectiveness of the proposed compensation approach, simulations under different levels of traffic load are performed. As all nodes in the network shown in Fig. 5 can generate traffic, it is easy to simulate various NCS scenarios under different levels of traffic load. In our simulations, the traffic load is generated with reference to the example to be discussed later in Section 6.

### *5.4 Dynamics of Network-Induced Delays*

Understanding the behaviour of network-induced delays in an NCS is crucial for designing a delay compensation scheme that uses QoS measurements. The NCS performance obtained by using the compensation scheme will depend on how accurately a prediction of QoS parameters can be achieved (Colandairaj et al., 2007). For the compensation approach presented in this paper, the settings of a series of possible sensor-to-actuator delays is important for improving the NCS performance. Effort has been made in the literature to predict the distribution of network-induced delays in NCSs (Tian and Levy, 2008a; Tian et al., 2006).

As mentioned earlier, a deadline is predefined as $\sigma = 2T_s$ in our approach. Then, it is assumed that the network-induced delays are distributed in the range between 0 and $\sigma = 2T_s$. Using the NCS network topology designed in Section 5.1, we have

analyzed the dynamics of network-induced delays and packet losses. For a twenty-second simulation, we have obtained the distribution of network-induced delays $\tau$ as shown in Fig. 6. The control period is set to be 30ms, and thus the deadline is $\sigma = 60$ms. The measured packet loss rate is 7.946% under 81.377% traffic load.

### *5.5 Characterizing the QoC Performance*

There are many popular control performance indices to quantify the control performance of a control system. As two of the most popularly used performance indices, the integral of absolute error (IAE) and the integral of timed absolute error (ITAE) are adopted in this paper to measure the control performance improvement. They are respectively defined as

$$IAE = \int_0^\infty |e(t)|dt,\ ITAE = \int_0^\infty t|e(t)|dt. \tag{10}$$

A smaller IAE or ITAE index usually represents better control performance. It is worth mentioning that discretization is carried out in this paper for actual computation of both IAE and ITAE in Eq. (10).

## 6 Example 1: Control of Open-loop Stable Processes

To demonstrate the effectiveness of the compensation approach presented in this paper, this section conducts case studies for networked control of an open-loop unstable second-order plant. The case study configurations discussed in Section 5 are directly used in this section.

### *6.1 Plant Model and Setpoint Step Changes*

Consider a typical second-order plant (Tipsuwan and Chow, 2003)

$$G(s) = \frac{2029.826}{(s + 26.29)(s + 2.296)} \tag{11}$$

A direct current (DC) drive process can be described using this plant model.

For setpoint tracking, step changes from 50rad/s to 100rad/s in setpoint are introduced at $t = 0$s and 8s, respectively, and step changes from 100rad/s to 50rad/s in setpoint are also activated at $t = 4$s and 12s, respectively.

*6.2 PI Controller and its Settings*

While various controller can be designed for networked control of the open-loop stable second-order process in Eqn. (11), a PI controller is employed in our case studies. The continuous-time PI controller is described by

$$G_c(s) = K_c\,[1 + 1/(T_i s)]\,, \tag{12}$$

where $K_c$ and $T_i$ are controller gain and integral time, respectively.

The PI controller can be tuned by using various tuning rules (Åström and Hägglund, 1995). Two of the popularly used PI tuning rules are IAE tuning and ITAE tuning. For a first-order plus delay continuous-time process

$$G(s) = \frac{k_p}{T_p s + 1} e^{-\tau_p s} \tag{13}$$

where $k_p$, $T_p$ and $\tau_p$ are process gain, time constant, and delay, respectively, the IAE tuning of the PI controller for setpoint tracking gives

$$K_c = \frac{0.758}{k_p}\left(\frac{T_p}{\tau_p}\right)^{0.861},\ T_i = \frac{T_p}{1.020 - 0.323\frac{\tau_p}{T_p}}. \tag{14}$$

When considering the sensor-to-actuator delay $\tau$ in an NCS, we follow the idea of Tian and Levy (2008a) to modify the controller parameters to

$$K_c = \frac{0.758}{k_p}\left(\frac{T_p}{\tau_p + \tau}\right)^{0.861},\ T_i = \frac{T_p}{1.020 - 0.323\frac{(\tau_p+\tau)}{T_p}}. \tag{15}$$

In order to employ this IAE tuning (15), which is for the first-order plus delay processes (13), the second-order process (11) needs to be reduced to the first-order plus delay form in Eqn. (13). This implies that the three model parameters $k_p, T_p$ and $\tau_p$ in Eqn. (13) are required to be estimated from Eqn. (11). While there are several ways to estimate these parameters, the simple half rule proposed by Skogestad (2003) is employed. The resulting model parameters are $k_p = 33.628$; $T_p = 0.455$ and $\tau_p = 0.019$.

From these model parameters, the PI controller settings can be derived by using the IAE tuning rules (15) for setpoint. Table 1 shows optimal controller settings for a series of possible sensor-to-actuator delays under the settings of $T_s = 30$ms, $\sigma = 2T_s$ and $n = 5$.

For digital control, the continuous-time version of the PI controller in Eqn 12 has to be converted to a discrete-time form. A method of discretizing the PI controller

is given by (Colandairaj et al., 2007; Gabel and Litz, 2004):

$$\begin{cases} P(k) = K_c e(k), \\ I(k) = I(k-1) + T_s \frac{e(k)+e(k-1)}{2T_i}, \\ U(k) = P(k) + I(k) \end{cases} \tag{16}$$

where $e$ means the deviation of the plant output from its setpoint. This form of discrete-time PI controller is adopted in our case studies. It is worth mentioning that other forms of discrete-time PI controller are also available.

### *6.3 Computation of Predictive Control Values*

Corresponding to the five possible sensor-to-actuator delays, the five sets of PI controller parameters in Table 1 are used to compute five predictive control values for delay compensation. The predictive control values are sent to the actuator in a single packet, and are put into queue $Q_1$ if the control packet arrives at the actuator before the deadline $\sigma$. The detailed computation algorithms have been proposed in Section 4.

In order to cope with packet losses in NCSs, the packet loss compensation scheme is also activated and the corresponding algorithms described in Section 4 are employed to predict lost control values. The number of predictive control values for the packet loss compensation is also set to be $m = 5$.

### *6.4 Results Analysis*

Fig. 7 depicts some simulation results for typical NCS scenarios with a continuous-time plant in Eq. (11), a discrete-time controller in Eq. (16) and Table 1, and a relatively high network utilization of 81.377%. Under this traffic load, the packet loss ratio is about 7.946%. The actual sensor-to-actuator delays are depicted in Fig. 6. It is shown in Fig. 7 that compensating for variable network-induced delays and packet losses, the proposed compensation approach improves the NCS performance noticeably with compressed overshoot and shorter settling time.

Quantitative control performance evaluation is also carried out. The results are tabulated in Table 2. It is seen from this table that by applying the proposed compensation approach with the step algorithm, the IAE index is reduced from 43.533 down to 33.876, representing an improvement of as much as over 22%. When the linear algorithm is employed the proposed approach also improvs the IAE index by 17%.

The ITAE index reduces the weighting of the large initial error and penalizes the

small error occurring later more heavily. It is shown in Table 2 that by employing our compensation approach with respective linear and step algorithms, the ITAE index can be reduced as much as 23% and 32%, respectively. Such improvements are considered to be significant.

Moreover, to confirm that the proposed compensation approach behaves with improved NCS control performance, the IAE and ITAE indices are quantitatively evaluated under different levels of NCS traffic load. The results are given in Figs. 8 and 9. It is observed that the IAE and ITAE indices are improved at all discrete points of the traffic load conditions in our simulations. The performance improvement becomes more significant when the traffic load becomes heavy, e.g., larger than 80% of the network capacity. This is because the network-induced delays and packet losses become severer under heavier traffic load conditions.

## 7 Example 2: Control of Open-Loop Unstable Processes

From the control point of view, control of open-loop unstable processes are more challenging than that of open-loop stable ones. Aiming to demonstrate the effectiveness of the proposed predictive compensation approach for a wide range of industrial processes, this section carries out case studies for networked control of an open-loop unstable plant. All case study configurations are directly taken from Section 5. In addition, plant and controller settings which will be discussed below.

Consider the following open-loop unstable process (Jhunjhunwala and Chidambaram, 2001):

$$G(s) = \frac{k_p}{T_p s - 1} e^{-\tau_p s}, \tag{17}$$

with $k_p = 1$, $T_p = 1.0$ and $\tau_p = 0.4$ .

A proportional-integral-derivative (PID) controller is adopted for this process:

$$G_c(s) = K_c \left[1 + 1/(T_i s) + T_d s\right], \tag{18}$$

where $K_c$, $T_i$ and $T_d$ are controller gain, integral and derivative times, respectively.

Similar to the IAE tuning method employed in example 1, a widely-used servo tuning method for unstable first-order plus delay processes (17) has been adopted in example 2 (Jhunjhunwala and Chidambaram, 2001). The IAE tuning of the PID controller for setpoint tracking gives:

$$\begin{aligned} &K_c = \tfrac{1.397}{k_p}\left(\tfrac{T_p}{\tau_p}\right)^{0.769},\ T_i = 0.856 T_p e^{2.044\frac{\tau_p}{T_p}}, \\ &T_d = 0.5643\tau_p + 0.0075 T_p. \end{aligned} \tag{19}$$

In consideration of the sensor-to-actuator delay $\tau$ in an NCS, the controller parameters can be modified to:

$$\begin{aligned} &K_c = \tfrac{1.397}{k_p}\left(\tfrac{T_p}{\tau_p+\tau}\right)^{0.769},\ T_i = 0.856T_p e^{2.044\frac{(\tau_p+\tau)}{T_p}}, \\ &T_d = 0.5643(\tau_p+\tau) + 0.0075T_p. \end{aligned} \tag{20}$$

The PID controller in Eq. (18) is implemented in the following discrete-time form:

$$\begin{cases} U(k) = K_c e(k) + I(t_k) + D(t_k), \\ I(k) = I(t_{k-1}) + \frac{K_c}{T_i} T_s \frac{e(k)+e(k-1)}{2}, \\ D(k) = K_c T_d \frac{e(k)-e(k-1)}{T_s}, \end{cases} \tag{21}$$

where $e$ means the deviation of the plant output from its setpoint; $U$ represents control signal; $I$ and $D$ are integral and derivative actions, respectively. The subscript $k$ indicates the $k$th period.

Table 3 shows the five sets of PID controller parameters corresponding to the five possible sensor-to-actuator delays. As in the first case study presented in Section 6.3, the five sets of PID controller parameters are used to compute five predictive control values for delay compensation.

In this case study, a setpoint tracking test has been carried out where the servo responses for a unit step change in the set point at $t = 1$s are evaluated. The simulation results will be discussed in the following section.

### *7.1 Results Analysis*

Some simulation results for typical NCS scenarios with a continuous-time plant in Eq. (17) and a discrete-time controller in Eq. (21) are shown in Fig. 10 and 11. For delay compensation the five sets of the PID controller are shown in Table 3 while for packet loss compensation the predicted control values are computed from Eqs. (7), (8) and (10). The network in the NCS scenarios considered in this case study is under a relatively heavy traffic load where the network utilization is about 57.86% and the packet loss ratio is about 11.4%. As shown in Fig. 10, the servo response is becoming unstable and the servo response reaches as much as 7 units. However, Fig. 11 clearly shows that the proposed compensation approach gives improved servo responses for networked control of this open-loop unstable process.

Table 4 shows IAE and ITAE values for the servo response by using the proposed compensation approach. It is observed from this table that the IAE and ITAE values are reduced significantly when the proposed compensation approach with linear or step algorithm. It is indicated that the proposed compensation approach for

network-induced delay and packet loss gives improved NCS control performance of not only open-loop stable process but also open-loop unstable processes.

## 8 Conclusions

As a co-design in both network and controller, a predictive compensation approach has been proposed in this paper to tackle the problem of simultaneous network-induced delays and packet losses along the way from sensors to NCS controller and then to actuators. Different from the majority of the existing NCS control methods in which the controller design has been a focus, the proposed approach for compensation for variable network-induced delays and packet losses improves the NCS control performance through adjusting control actions according to actually measured sensor-to-actuator delays. In order to do so, a sequence of predictive control values are computed for a series of possible sensor-to-actuator delays. Compared with the existing packet-based NCS control methods, the proposed approach does not tightly rely on precise process models and full understanding of NCS network dynamics, providing a useful tool for networked control of modern complex industrial processes. The paper has described in detail how a sequence of predictive redundant control values are computed in the controller for a series of possible sensor-to-actuator delays within a predefined deadline, how these values are transmitted over the network and stored in the actuator, and how a control signal is computed from the control values stored in the actuator. The effectiveness of the proposed approach has been demonstrated through case studies.

## Acknowledgements

This work was supported in part by Australian Government's Department of Innovation, Industry, Science and Research (DIISR) under International Science Linkages (ISL) grant number CH070083.

## References

Aidan, O. (2006). Performance improvement using simple PID controller tuning formulae. In *The 3rd IET International Conference on Power Electronics, Machines and Drives - PEMD'06*, pages 276–280, Dublin, Ireland.

Antsakls, P. and Baillieul, J. (2007). Guest editorial: Special issue on technology of networked control systems. *Proceedings of the IEEE*, 95(1):5–8.

Åström, K. J. and Hägglund, T. (1995). *PID Controllers: Theory, Design, and*

*Tuning*. Instrument Society of America, Research Triangle Park, NC, 2nd edition edition.

Baillieul, J. and Antsaklis, P. J. (2007). Control and communication challenges in networked real-time systems. *Proceedings of the IEEE*, 95(1):9–28.

Branicky, S., Liberatore, V., and Phillips, S. M. (2003). Networked control system co-simulation for co-design. In *Proceedings of the 2003 American Control Conference*, volume 4, pages 3341–3346, Denver, Colorado.

Cervin, A., Henriksson, D., Lincoln, B., Eker, J., and Arzen, K.-E. (2003). How does control timing affect performance? *IEEE control systems magazine*, 23(3):16–31.

Chaillet, A. and Bicchi, A. (2008). Delay compensation in packet-switching networked controlled systems. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 3620–3625, Cancun, Mexico.

Colandairaj, J., Irwin, G. W., and Scanlon, W. G. (2007). Wireless networked control systems with QoS-based sampling. *IET Control Theory and Applications*, 1(1):430–438.

Decotignie, J.-D. (2005). Ethernet-based real-time and industrial communications. *Proceedings of the IEEE, Special Issue on Industrial Communication Systems*, 93(6):1102–1117.

El-Farra, N. H. and Mhaskar, P. (2008). Special issue on "Control of networked and complex process systems". *Computers and Chemical Engineering*, 32(9):1963.

Gabel, O. and Litz, L. (2004). QoS-adaptive control in NCS with variable delays and packet losses - a heuristic approach. In *43rd IEEE Conference on Decision and Control*, volume 2, pages 1586–1591, Altantis, Paradise Island, Bahamas.

Gupta, R. A. and Chow, M.-Y. (2010). Networked control system: Overview and research trends. *IEEE Transactions on Industrial Electronics*, 57(7):2527–2535.

Heemels, W. P. M. H., Teel, A. R., van de Wouw, N., and Nesic, D. (2010). Networked control systems with communication constraints: Tradeoffs between transmission intervals, delays and performance. *IEEE Transactions on Automatic Control*, 55(8):1781–1796.

Hespanha, J. P., Naghshtabrizi, P., and Xu, Y. (2007). A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162.

Jhunjhunwala, M. K. and Chidambaram, M. (2001). Pid controller tuning for unstable systems by optimization method. *Chemical Engineering Communications*, 185(1):91–113.

Koivo, H. N. and Reijonen, A. (2004). Tuning of PID controllers for varying time-delay systems. In *Proceedings of the IEEE International Conference Mechatronics - ICM'04*, pages 446–451, Denver, Colorado.

Luck, R. and Ray, A. (1994). Experimental verification of a delay compensation algorithm for integrated communication and control systems. *Journal of Electronics and Control*, 59(6):1357–1372.

Marti, P., Vill, R., Fuertes, J. M., and Fohler, G. (2005). Networked control systems overview. In Zurawski, R., editor, *The Industrial Information Technology Handbook*, volume 1, pages 1–16. CRC Press, South San Francisco, California, USA.

Martins, E. C. and Jota, F. G. (2010). Design of networked control systems with explicit compensation for time-delay variations. *IEEE Transactions on Systems, Man, and Cybernetics*, 40(3):308–318.

Mikael, P., Eriksson, L., and Koivo, H. (2006). Tuning of PID controllers for networked control systems. In *Proceedings of the 32nd Annual Conference on IEEE Industrial Electronics - IECON'06*, pages 4650–4655, Paris, France.

Nilsson, J. (1998). *Real-Time control systems with delays, PhD dissertation*. PhD thesis, Lund Institute of technology.

Peng, C. (2007). Networked, guaranteed cost control for a class of industrial processes with state delay. *Asia-Pacific Journal of Chemical Engineering*, 2(6):650–658.

Peng, C. and Tian, Y.-C. (2009). Delay-dependent robust $H\_\infty$ control for uncertain systems with time-varying delay. *Information Sciences*, 179(18):3187–3197.

Sala, A., Cuenca, Á., and Salt, J. (2009). A retunable pid multi-rate controller for a networked control system. *Information Sciences*, 179(14):2390–2402.

Skogestad, S. (2003). Simple analytical rules for model reduction and PID controller tuning. *Journal of Process Control*, 13:291–309.

Sun, Y. and El-Farra, N. H. (2008). Quasi-decentralized model-based networked control of process systems. *Computers and Chemical Engineering*, 32(9):2016–2029.

Tavakoli, S. and Tavakoli, M. (2003). Optimal tuning of PID controllers for first order plus time delay models using dimensional analysis. In *Proceedings of The fourth international conference on control and automation - ICCA'03*, pages 942–946, Montreal, Canada.

Tavassoli, B., Jabehdar-Maralani, P., and Rezaee, N. (2009). Tuning of control systems over csma networks. *IEEE Transactions on Industrial Electronics*, 56(4):1282–1291.

Tian, G., Fidge, C., and Tian, Y.-C. (2008). High-precision relative clock synchronization using time stamp counters. In *Proceedings of the 13th IEEE International Conference on Engineering of Complex Computer Systems - ICECCS'08*, pages 69–78, Belfast, Northern Ireland.

Tian, G., Fidge, C., and Tian, Y.-C. (2009). Hybrid system simulation of computer control applications over communication networks. In *Proceedings of the IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems - MASCOTS'09*.

Tian, G.-S. and Tian, Y.-C. (2011). Modelling and performance evaluation of the IEEE 802.11 DCF for real-time control. *Computer Networks*, in press. DOI: 10.1016/j.comnet.2011.10.001.

Tian, Y.-C. and Gao, F. (1999). Injection velocity control of thermoplastic injection molding via a double controller scheme. *Industrial and Engineering Chemistry Research*, 38(9):3396–3406.

Tian, Y.-C. and Levy, D. (2008a). Compensation for control packet dropout in networked control systems. *Information Sciences*, 178(5):1263–1278.

Tian, Y.-C. and Levy, D. (2008b). Dealing with network complexity in real-time networked control. *International Journal of Computer Mathematics*,

85(8):1235–1253.

Tian, Y.-C., Levy, D., Tade, M. O., Gu, T. L., and Fidge, C. (2006). Communication architecture design for real-time networked control systems. In *Proceedings of International Conference on Communications, Circuits and Systems*, volume 3, pages 1840–1845, Guilin, China.

Tian, Y.-C., Yu, Z.-G., and Fidge, C. (2007). Multifractal nature of network induced time delay in networked control systems. *Physics Letters A*, 361:103–107.

Tian, Y.-C., Zhao, F., Bisowarno, B. H., and Tadé, M. O. (2003). Pattern-based predictive control for ETBE reactive distillation. *Journal of Process Control*, 13(1):57–67.

Tipsuwan, Y. and Chow, M.-Y. (2003). Control methodologies in networked control systems. *Control Engineering Practice*, 11(1):1099–1111.

UCB/LBNL/VINT Groups (2011). ns Network Simulator. http://www.isi.edu/nsnam/ns/, accessed on 8 Feb 2011.

Xia, Y., Liu, G. P., Fu, M., and Rees, D. (2009). Predictive control of networked systems with random delay and data dropout. *IET Control Theory and Applications*, 3(11):1476–1486.

Zhang, T., Tadé, M. O., Tian, Y.-C., and Zan, H. (2008). High-resolution method for numerically solving PDEs in process engineering. *Computers and Chemical Engineering*, 32(10):2403–2408.

Zhao, Y.-B., Liu, G.-P., and Rees, D. (2008). Networked predictive control systems based on the hammerstein model. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 55(5):469–473.

Zhao, Y.-B., Liu, G.-P., and Rees, D. (2009a). Design of a packet-based control framework for networked control systems. *IEEE Transactions on Control Systems Technology*, 17(4):859–865.

Zhao, Y.-B., Liu, G.-P., and Rees, D. (2009b). Modeling and stabilization of continuous-time packet-based networked control systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(6):1646–1652.

Zou, Y., Chen, T., and Li, S. (2010). Network-based predictive control of multirate systems. *IET Control Theory and Applications*, 4(7):1145–156.

**Captions of Illustrations**

Figure 1. Block diagram of a typical networked control system (Tian et al., 2006).

Figure 2. Assumed dumbbell network structure in NCSs.

Figure 3. Workflow of the predictive compensation approach.

Figure 4. Possible network-induced delays for delay compensation ($T_s$: control period; $\tau$: measured network-induced delay; $\tau_j$: the $j$th possible delay; $\sigma$: delay deadline beyond which the packet is treated as a packet loss).

Figure 5. Network topology in our simulations.

Figure 6. Sensor-to-actuator transmission delays in a control loop.

Figure 7. System state with and without compensation for delays and packet losses under the traffic load of 81.377%.

Figure 8. IAE improvement under different levels of traffic load.

Figure 9. ITAE improvement under different levels of traffic load.

Figure 10. Servo response without compensation for delays and packet losses (Example 2).

Figure 11. Servo response with compensation for delays and packet losses (Example 2).

Table 1. Optimal controller settings under $T_s = 30$ms, $\sigma = 2T_s$ and $n = 5$ (Example 1).

Table 2. IAE and ITAE indices under the traffic load of 81.377% (Example 1).

Table 3. Optimal PID controller settings under $T_s = 20$ms, $\sigma = 2T_s$ and $n = 5$ (Example 2).

Table 4. IAE and ITAE indices under the traffic load of 57.86 (Example 2)%

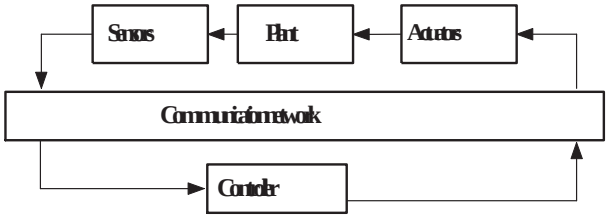


Fig. 1. Block diagram of a typical networked control system (Tian et al., 2006).

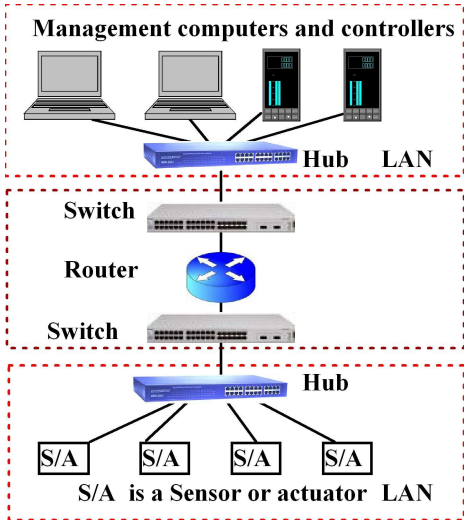


Fig. 2. Assumed dumbbell network structure in NCSs.

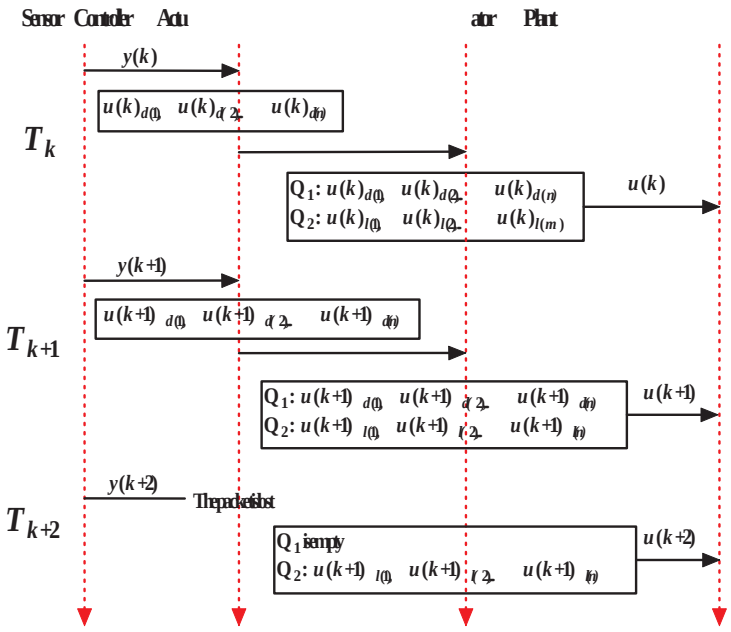


Fig. 3. Workflow of the predictive compensation approach.

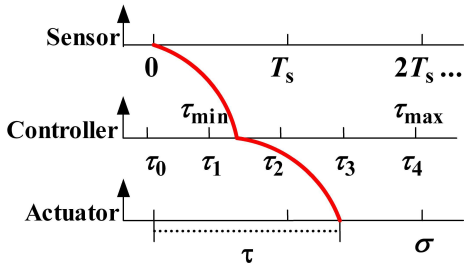


Fig. 4. Possible network-induced delays for delay compensation ($T_s$: control period; $\tau$: measured network-induced delay; $\tau_j$: the $j$th possible delay; $\sigma$: delay deadline beyond which the packet is treated as a packet loss).

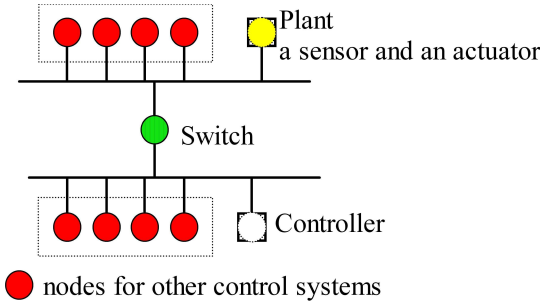


Fig. 5. Network topology in our simulations.

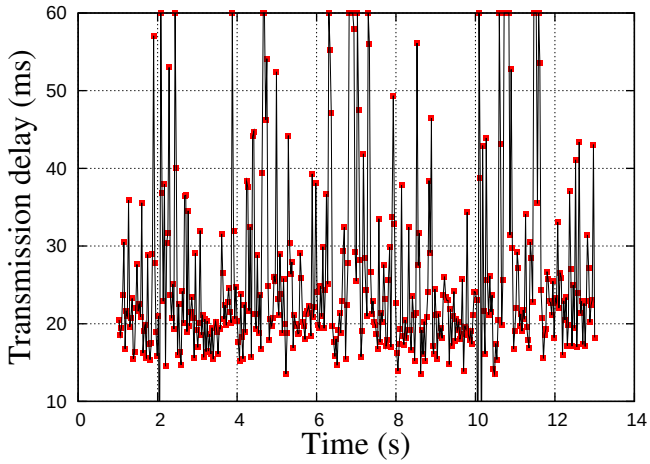


Fig. 6. Sensor-to-actuator delays in a control loop.

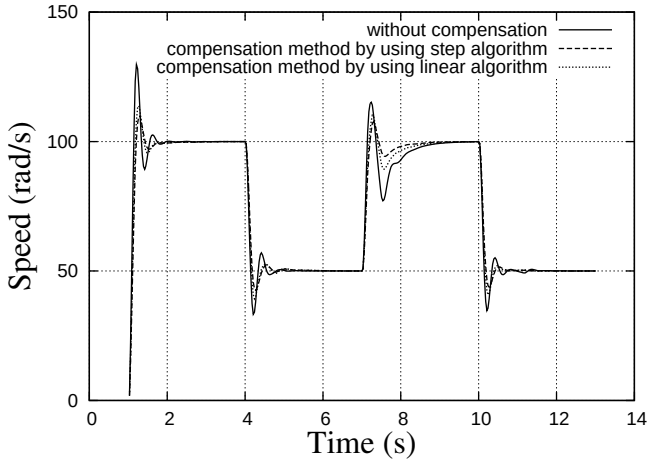


Fig. 7. System state with and without compensation for delays and packet losses under the traffic load of 81.377%.

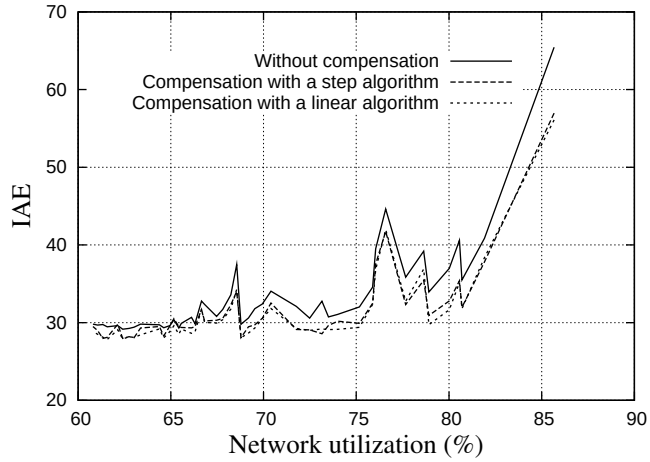


Fig. 8. IAE improvement under different levels of traffic load.

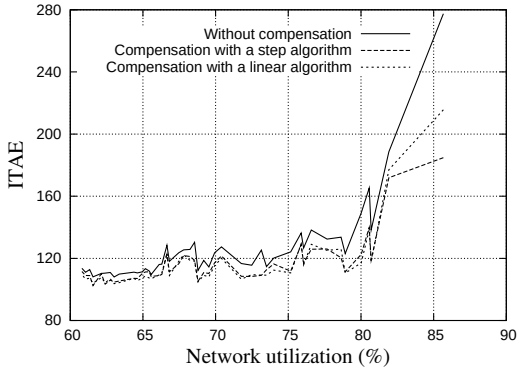


Fig. 9. ITAE improvement under different levels of traffic load.

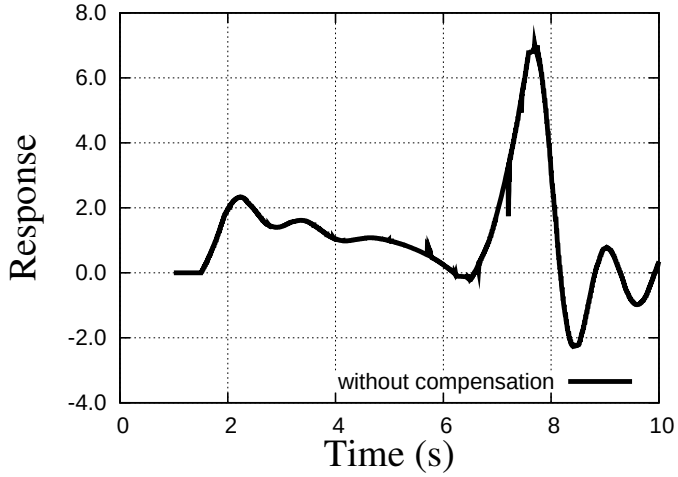


Fig. 10. Servo response without compensation for delays and packet losses (Example 2).

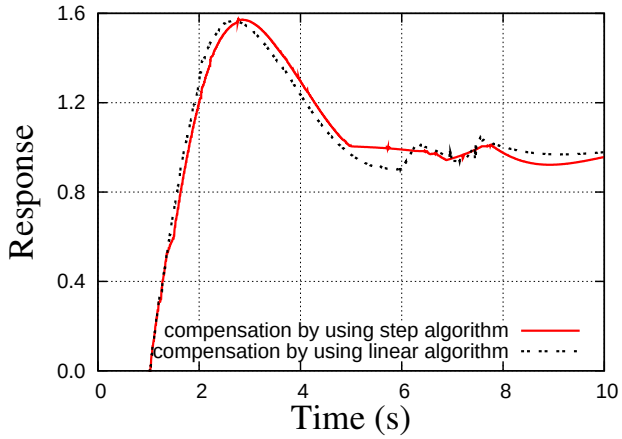


Fig. 11. Servo response with compensation for delays and packet losses (Example 2).

Table 1
Optimal controller settings under $T_s$ = 30ms, $\sigma = 2T_s$ and $n$ = 5 (Example 1).

| Predicted delay | $K_c$ | $T_i$ |
|---|---|---|
| $\tau_0$ ( 0ms) | 0.170100 | 0.450000 |
| $\tau_1$ (15ms) | 0.143359 | 0.462737 |
| $\tau_2$ (30ms) | 0.115682 | 0.467813 |
| $\tau_3$ (45ms) | 0.097442 | 0.473002 |
| $\tau_4$ (60ms) | 0.084458 | 0.478307 |

Table 2
IAE and ITAE indices under the traffic load of 81.377% (Example 1).

| Compensation Method | IAE | IAE Improvement | ITAE | ITAE Improvement |
|---|---|---|---|---|
| No compensation | 43.553 | - | 197.440 | - |
| - linear algorithm | 36.133 | > 17% | 151.872 | > 23% |
| - step algorithm | 33.876 | > 22% | 134.121 | > 32% |

Table 3
Optimal PID controller settings under $T_s$ = 20ms, $\sigma = 2T_s$ and $n$ = 5 (Example 2).

| Predicted delay | $K_c$ | $T_i$ | $T_d$ |
|---|---|---|---|
| $\tau_0$ ( 0ms) | 2.8050 | 1.9514 | 0.2321 |
| $\tau_1$ (10ms) | 2.7993 | 1.9588 | 0.2360 |
| $\tau_2$ (20ms) | 2.7731 | 1.9789 | 0.2389 |
| $\tau_3$ (30ms) | 2.7474 | 1.9992 | 0.2417 |
| $\tau_4$ (40ms) | 2.7222 | 2.0198 | 0.2445 |

Table 4
IAE and ITAE indices under the traffic load of 57.86% (Example 2).

| Compensation Method | IAE | ITAE |
|---|---|---|
| No compensation | Unstable | Unstable |
| - linear algorithm | 81.232 | 184.891 |
| - step algorithm | 78.242 | 163.018 |