Available online at www.sciencedirect.com
SCIENCE DIRECT®
PERGAMON
Control Engineering Practice 11 (2003) 1099–1111
CONTROL ENGINEERING PRACTICE
www.elsevier.com/locate/conengprac

Control methodologies in networked control systems

Yodyium Tipsuwan, Mo-Yuen Chow*

Advanced Diagnosis and Control Lab, Department of Electrical and Computer Engineering, North Carolina State University, Daniels Hall, Raleigh, NC 27695-7911, USA

Received 12 February 2003; accepted 16 February 2003

Abstract

The use of a data network in a control loop has gained increasing attentions in recent years due to its cost effective and flexible applications. One of the major challenges in this so-called networked control system (NCS) is the network-induced delay effect in the control loop. Network delays degrade the NCS control performance and destabilize the system. A significant emphasis has been on developing control methodologies to handle the network delay effect in NCS. This survey paper presents recent NCS control methodologies. The overview on NCS structures and description of network delays including characteristics and effects are also covered.

Keywords: Communication control applications; Communication networks; Communication protocols; Distributed control; Factory automation; Delay analysis; Delay compensation

1. Introduction

The research and developments on shared data networks have a long history. Principle data networks such as Slotted ALOHA (Stallings, 2000), and ARPANET (Minoli & Schmidt, 1999), which were specially developed around 30–40 years ago, evolved to widely used modern network protocols like Ethernet and Internet for general usages, respectively. Data networking technologies provide several benefits on linking data points like computers. Networks enable remote data transfers and data exchanges among users, reduce the complexity in wiring connections and the costs of medias, and provide ease in maintenance.

Because of these attractive benefits, many industrial companies and institutes have shown interest in applying networks for remote industrial control purposes and factory automation. As a result of extensive research and development, several network protocols for industrial control have been released. For example, Controller Area Network (CAN) was originally developed in 1983 by the German company Robert Bosch for use in car industries, and is also being used now in many other industrial control applications. Another example of industrial networks is Profibus developed by six German companies and five German institutes in 1987. Profibus is a broadcast bus protocol that operates as a multi-master/slave system. Many other industrial network protocols including Foundation Fieldbus and DeviceNet were also developed about the same time period. Most of these protocols are typically reliable and robust for real-time control purposes.

Meanwhile, the technologies on general computer networks especially Ethernet have also progressed very rapidly. With the decreasing price, increasing speed, widespread usages, numerous software and applications, and well-established infrastructure, these networks become major competitors to the industrial networks for control applications (Kaplan, 2001). Furthermore, the popularity of the Internet has brought these networks into various organizations. Thus, the control applications can utilize these networks to connect to the Internet in order to perform remote control at much farther distances than in the past without investing on the whole infrastructure. Although the industrial networks have been enhanced for Internet connectivity, the

*Corresponding author. Tel.: +1-919-5155405; fax: +1-919-5155108.
E-mail address: chow@eos.ncsu.edu (M.-Y. Chow).

0967-0661/03/$ - see front matter © 2003 Published by Elsevier Ltd.
doi:10.1016/S0967-0661(03)00036-4

cheaper price and widespread usages of the general networks are still attractive for use in control applications.

Regardless of the type of network used, the overall networked control system (NCS) performance is always affected by network delays since the network is tied with the control system. Delays are widely known to degrade the performance of a control system. Network delays may not significantly affect an open-loop control system such as on-off relay systems in industrial plants. However, the open-loop control configuration may not be appropriate and adequate for time-sensitive high performance control applications such as telerobotics and telesurgery. These applications require feedback data sent across the network in order to correct the output error. Existing constant time-delay control methodologies may not be directly suitable for controlling a system over the network since network delays are usually time-varying, especially in the Internet. Therefore, to handle network delays in a closed-loop control system over a network, an advanced methodology is required.

This survey paper provides recent control methodologies for a closed-loop control system over a data network. This closed-loop system configuration is known as a *network-based control system* (Kim, Park, & Kwon, 1996) or NCS (Walsh, Ye, & Bushnell, 1999c). The two terms are somewhat interchangeable depending on different authors' preferences. The methodologies described in this paper have been applied and have shown promising results in many applications ranging from DC motors (Tipsuwan & Chow, 2001; Kim, Park, & Kwon, 1998) to automobiles (Boustany et al., 1992; Özgüner et al., 1992), aircrafts (Ray, 1987), mobile robots (Wargui, Tadjine, & Rachid, 1996; Tipsuwan & Chow, 2002), robotic manipulator (Tarn & Xi, 1998), and distance learning (Kondraske et al., 1993; Overstreet & Tzes 1999). This paper provides the overview of NCS including system configuration, network delay characteristics, and the effects of networked delays in Section 2. The control methodologies for NCS will then be described in Section 3. The paper is concluded in Section 4.

2. Overview of NCS

2.1. NCS configuration

There are two general NCS configurations listed as follows:

Direct structure. The NCS in the direct structure is composed of a controller and a remote system containing a physical plant, sensors and actuators. The controller and the plant are physically located at different locations and are directly linked by a data network in order to perform remote closed-loop control as illustrated in Fig. 1.

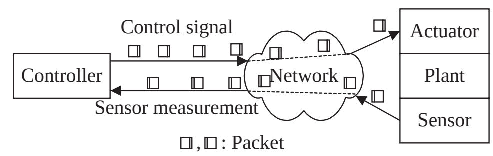

Fig. 1. NCS in the direct structure.

The control signal is encapsulated in a frame or a packet and sent to the plant via the network. The plant then returns the system output to the controller by putting the sensor measurement into a frame or a packet as well. In a practical implementation, multiple controllers can be implemented in a single hardware unit to manage multiple NCS loops in the direct structure. Some examples of NCS in the direct structure are a distance learning lab (Overstreet & Tzes, 1999) and a DC motor speed control system (Tipsuwan & Chow, 2001).

Hierarchical structure. The basic hierarchical structure consists of a main controller and a remote closed-loop system as depicted in Fig. 2.

Periodically, the main controller computes and sends the reference signal in a frame or a packet via a network to the remote system. The remote system then processes the reference signal to perform local closed-loop control and returns to the sensor measurement to the main controller for networked closed-loop control. The networked control loop usually has a longer sampling period than the local control loop since the remote controller supposes to satisfy the reference signal before processes the newly arrival reference signal. Similar to the direct structure, the main controller can be implemented to handle multiple networked control loops for several remote systems. This structure is widely used in several applications including mobile robots (Tipsuwan & Chow, 2002), and teleoperation (Tarn & Xi, 1998).

The use of either the direct structure or the hierarchical structure is based on application requirements and designer's preferences. For example, a robotic manipulator usually requires several motors at the joints of the robot to simultaneously and smoothly rotate together. It may be more convenient and more robust to use an existing robot controller and formulate the networked control problem in the hierarchical structure. On the other hand, a designer may require a networked DC motor speed control system (Tipsuwan & Chow, 2001) to have a faster control response over the network. The direct structure may be preferred in this case.

This survey paper mainly focuses on the fundamental and control methodologies for NCS in the direct structure. Nevertheless, control and analysis methodologies for the

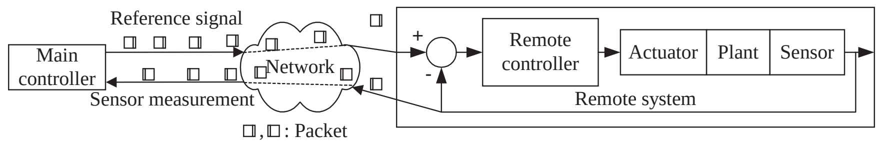

Fig. 2. NCS in the hierarchical structure.

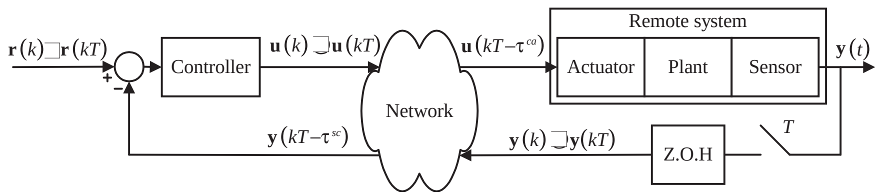

Fig. 3. General NCS configuration and network delays for NCS formulations.

direct structure could also be applied for the hierarchical structure by treating the remote closed-loop system as a pure plant. In this case, the remote closed-loop system is represented by a state-space model or a transfer function similar to the plant.

2.2. Delays in-the-loop

Since an NCS operates over a network, data transfers between the controller and the remote system will induce network delays in addition to the controller processing delay. Fig. 3 shows network delays in the control loop, where **r** is the reference signal, **u** is the control signal, **y** is the output signal, k is the time index, and T is the sampling period. Most of networked control methodologies use the discrete-time formulation shown in Fig. 3. Fig. 4 shows the corresponding timing diagram of network delay propagations.

Network delays in an NCS can be categorized from the direction of data transfers as the sensor-to-controller delay τ^{sc} and the controller-to-actuator delay τ^{ca}. The delays are computed as

$$\tau^{sc} = t^{cs} - t^{se}, \tag{1}$$

$$\tau^{ca} = t^{rs} - t^{ce}, \tag{2}$$

where t^{se} is the time instant that the remote system encapsulates the measurement to a frame or a packet to be sent, t^{cs} is the time instant that the controller starts processing the measurement in the delivered frame or packet, t^{ce} is the time instant that the main controller encapsulates the control signal to a packet to be sent, and t^{rs} is the time instant that the remote system starts processing the control signal. In fact, both network delays can be longer or shorter than the sampling time T. The controller processing delay τ^{c} and both network delays can be lumped together as the control delay τ for ease of analysis. This approach has been used in some networked control methodologies. Although the controller processing delay τ^{c} always exists, this delay is usually small compared to the network delays, and could be neglected. In addition, the sampling periods of the main controller and of the remote system may be different in some cases.

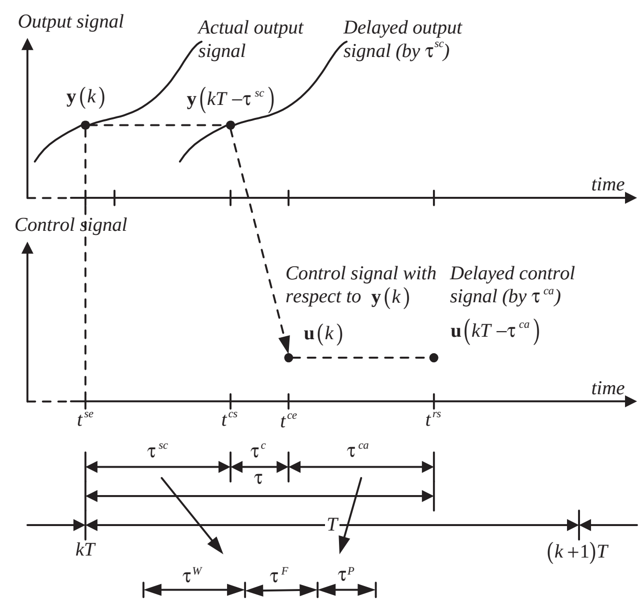

Fig. 4. Timing diagram of network delay propagations.

The delays τ^{sc} and τ^{ca} are composed of at least the following parts (Lian, Moyne, & Tilbury, 2001).

- *Waiting time delay* τ^{W}. The waiting time delay is the delay, of which a source (the main controller or the remote system) has to wait for queuing and network availability before actually sending a frame or a packet out.
- *Frame time delay* τ^{W}. The frame time delay is the delay during the moment that the source is placing a frame or a packet on the network.

- *Propagation delay* τ^P. The propagation delay is the delay for a frame or a packet traveling through a physical media. The propagation delay depends on the speed of signal transmission and the distance between the source and destination.

These three delay parts are fundamental delays that occur on a local area network. When the control or sensory data travel across networks, there can be additional delays such as the queuing delay at a switch or a router, and the propagation delay between network hops. The delays τ^{sc} and τ^{ca} also depend on other factors such as maximal bandwidths from protocol specifications, and frame or packet sizes.

Higher layer network protocols such as TCP may require retransmission if an error occurs in a packet, or a switch or a router drops the packet. This incident is a trade-off for an NCS. Even though some control or sensory signals are lost due to network transmissions, some NCS may operate acceptably. In this case, retransmission may be undesirable because the NCS may be severely affected by the extending delays as a result from retransmission.

2.3. Delay characteristics

The delay characteristics on NCS basically depend on the type of a network used, which are described as follows.

Cyclic service network. In local area network protocols with cyclic service such as IEEE 802.4, SAE token bus, PROFIBUS, IEEE 802.5, SAE token ring, MIL-STD-1553B, and FIP, control and sensory signals are transmitted in a cyclic order with deterministic behaviors. Thus, the delays are periodic and can be simply modeled as a periodic function such that $\tau_k^{sc} = \tau_{k+1}^{sc}$ and $\tau_k^{ca} = \tau_{k+1}^{ca}$, where τ_k^{sc} and τ_k^{ca} are the sensor-to-controller delay and the controller-to-actuator delay at the sampling time period k (Halevi & Ray, 1988). The models work perfectly in the ideal case. In practice, NCS may experience small variations on periodic delays due to several reasons. For examples, the discrepancies in clock generators on a controller and a remote system may result in delay variations.

Random access network. Random access local area networks such as CAN and Ethernet involve with more uncertain delays. The significant parts of random network delays are the waiting time delays due to queuing and frame collision on the networks. When an NCS operates across networks, several more factors can increase the randomness on network delays such as the queuing time delays at a switch or a router, and the propagation time delays from different network paths. In addition, a cyclic service network connected to a random access network also results in random delays.

In the networking area, random network delays have been modeled by using various formulations based on probability and the characteristics of sources and destinations. The techniques range from simple approaches such as the Poisson process to more sophisticated approaches such as Markov chain (Shakkottai, Kumar, Karnik, & Anvekar, 2001), fluid flow model (Filipiak, 1988), ARMA model (Li & Mills, 2001), etc. These techniques have been brought to NCS formulations in several studies, but may have to be modified or reformulated for specific networked control methodologies. For example, Markov chain is applied (Krtolica et al., 1994; Nilsson, 1998), and simple independent transfer-to-transfer probability distribution models are used (Nilsson, 1998) as follows.

$$f_\tau(\tau_k^{ca}) = \delta(\tau_k^{ca} - a)\cdot(1-p_{ca}) + \delta(\tau_k^{ca} - b)\cdot p_{ca}, \tag{3}$$

$$f_\tau(\tau_k^{sc}) = \begin{cases} \delta(\tau_k^{sc} - a)\cdot(1-p_{sc}), & \tau_k^{sc} = a, \\ p_{sc}/(b-a^+), & \tau_k^{sc} \in (a,b], a<b, \\ 0, & \tau_k^{sc} \notin [a,b], \end{cases} \tag{4}$$

where $\delta(\cdot)$ is the Dirac delta function, a and b are constants, and $p_{sc}, p_{ca} \in [0,1]$ are parameters of the network.

2.4. Effects of delays in-the-loop

2.4.1. Performance degradation

Delays in a control loop are widely known to degrade system performances of a control system, so are the network delays in an NCS. The closed-loop proportional-integral (PI) control system with delays in Fig. 5(a) is used to briefly illustrate system performance degradations by delays in-the-loop, where $R(s)$, $U(s)$, $Y(s)$, and $E(s) = R(s) - Y(s)$ are the reference, control, output, and error signals in Laplace domain according to the reference, control, output, and error signals in time domain, respectively.

The transfer functions of the controller and the plant are described, respectively, as follows:

$$G_C(s) = \frac{\beta K_P(s + (K_I/K_P))}{s},$$
$$K_P = 0.1701, \quad K_I = 0.378, \tag{5}$$

$$G_P(s) = \frac{2029.826}{(s+26.29)(s+2.296)}, \tag{6}$$

where $G_C(s)$ is a PI controller, K_P is the proportional gain, K_I is the integral gain, $G_P(s)$ is the plant of a DC motor (Tipsuwan & Chow, 1999), β is a parameter to adjust K_P and K_I. In this case, $\beta = 1$. As shown in Fig. 5(b), obvious system performance degradations are the higher overshoot and the longer settling time when the delays $\tau^{ca} = \tau^{sc} = \tau/2$ are longer. Other kinds of performance degradations can be evaluated based on different performance measures. Analyses on the effects

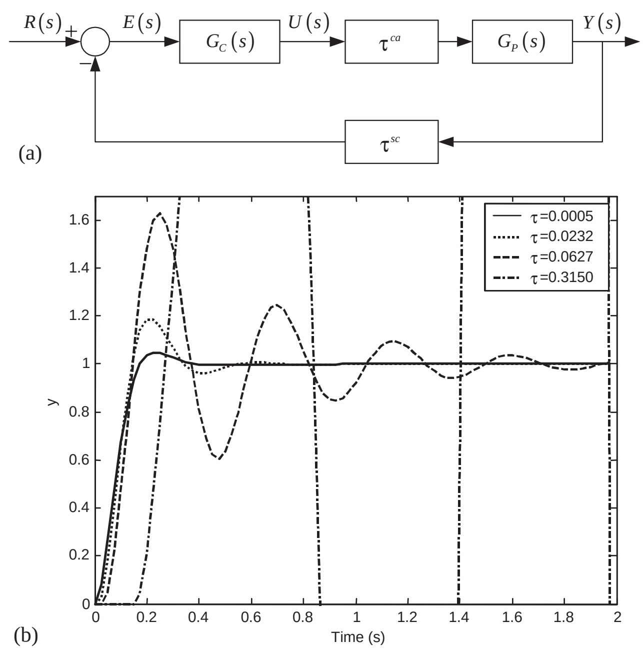

Fig. 5. System performance degradations caused by delays in-the-loop: (a) closed-loop control system example. (b) Step response with respect to various τ, where $\tau^{ca} = \tau^{sc} = \tau/2$ are constant, and $\beta = 1$.

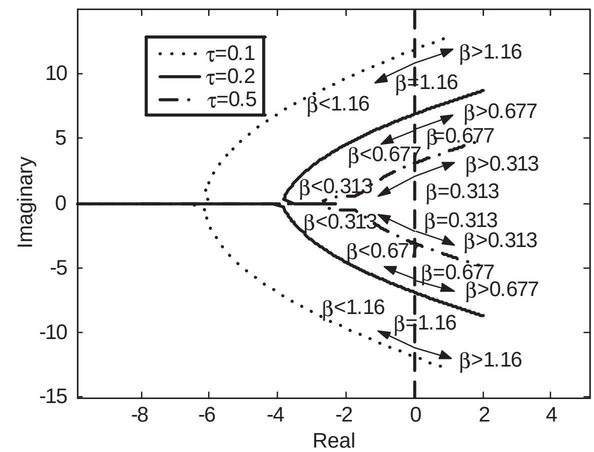

Fig. 6. Primary branches of the root locus of the system in Fig. 5 (a) with respect to β, where $\tau^{ca} = \tau^{sc} = \tau/2$ are constant.

of delays on system performance measures can be used for developing appropriate networked control methodologies (Yook, Tilbury, & Soparkar, 2000; Tipsuwan & Chow, 2001).

2.4.2. Destabilization

Delays in-the-loop including network delays in an NCS can destabilize the system by reducing the system stability margin. Again, the system in Fig. 5(a) is used to illustrate how the delays can reduce the stability region. Fig. 6 shows the branches of the root locus of the system in Fig. 5(a) with respect to the parameter β. In this case, increasing β is equivalent to increasing K_P and K_I while maintaining the same ratio between both controller gains. Only primary branches are shown because they are sufficient to approximate the stability region (Kuo, 1987).

As shown in Fig. 6, when the delay τ is longer, the primary branches of the root locus bend toward the right of the imaginary axis, and β, at the point at which the branches cross the imaginary axis, is smaller. This result indicates the narrower stability region since the PI controller has the smaller range of feasible values to use for stabilizing the closed-loop control system.

There have been several studies to derive stability criteria for an NCS in order to guarantee that the NCS can remain stable in a certain condition. However, there is no generic stability analysis that can be applied on every NCS. Most of stability analysis techniques are subject to network configurations, network protocols, assumptions, and control techniques used.

Simple stability analysis for a discrete-time delayed system (Åström & Wittenmark, 1990) can be applied to a constant delay NCS. A periodic delay NCS requires more sophisticated analysis based on various system formulations. For example, an NCS on a periodic delay network (Halevi & Ray, 1988) is stable if all eigenvalues of a specific formulation are contained in a unit circle. Another formulation (Hong, 1995) uses a general frequency domain analysis for checking stability, but the stability criterion is limited to a single-dimensional system.

Stability analysis for an NCS with random network delays is more challenging, since more advanced algorithms are usually required. Varieties of techniques have been used for different NCS formulations. For example (Nilsson, 1998; Krtolica et al., 1994), stabilities of NCS were analyzed based on stochastic stability analysis, but with different formulations. Nonlinear control and perturbation theories were applied for NCS stability analysis (Walsh, Ye, & Bushnell, 1999c) using the Bellman–Gronwall lemma. A hybrid system technique is used to analyze the stability of an NCS (Zhang, Branicky, & Phillips, 2001).

3. Recent networked control methodologies

Due to network delay concerns, the methodologies to control an NCS have to maintain the stability of the system in addition to controlling and maintaining the system performance as much as possible. Various methodologies have been formulated based on several types of network behaviors and configurations in conjunction with different ways to treat the delay problems. Some assumptions may be required. For example:

- Network transmissions are error-free.
- Every frame or packet always has the same constant length.
- The difference between the sampling times of the controller and of the sensor, called time skew Δ_k, is constant.
- The computational delay τ^c is constant and is much smaller than the sampling period T.
- The network traffic cannot be overloaded.
- Every dimension of the output measurement or the control signal can be packed into one single frame or packet.

Some methodologies are denoted by some specific terminologies defined by the authors of this paper in order to unify and distinguish them.

3.1. Augmented deterministic discrete-time model methodology

Halevi and Ray (1988) proposed a methodology named here as the augmented deterministic discrete-time model methodology to control a linear plant over a periodic delay network. The structure of the augmented discrete-time model is straightforward and easy. In addition, this methodology can be modified to support non-identical sampling periods of a sensor and a controller as mentioned in Liou and Ray (1990). The linear plant used in this methodology has the following form

$$\mathbf{x}(k+1) = \mathbf{\Phi}\mathbf{x}(k) + \mathbf{\Gamma}\mathbf{u}(k), \tag{7}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k), \tag{8}$$

where $\mathbf{\Phi} = \exp(\mathbf{A}T)$, $\mathbf{\Gamma} = \int_0^T \exp(\mathbf{A}\zeta)\,\mathrm{d}\zeta\mathbf{B}$, and $\{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$ is the realization of the system. With requiring a set point to be zero, the dynamics of the linear controller used in this methodology can be described by

$$\boldsymbol{\xi}(k+1) = \mathbf{F}\boldsymbol{\xi}(k) - \mathbf{G}\mathbf{z}(k), \tag{9}$$

$$\mathbf{u}(k) = \mathbf{H}\boldsymbol{\xi}(k) - \mathbf{J}\mathbf{z}(k), \tag{10}$$

where $\boldsymbol{\xi}$ is the controller state vector, $\mathbf{z}(k) = \mathbf{y}(k-i), i = \{1, \ldots, j\}$ is the past measurement at the instant when $\mathbf{u}(k)$ is processed by the controller, and $\mathbf{F}$, $\mathbf{G}$, $\mathbf{H}$, and $\mathbf{J}$ are constant matrices describing the dynamics of the controller. The control $\mathbf{u}$ in (10) is the output of this controller.

The main idea to handle network delays in this methodology is to combine and rearrange (7)–(10) into an augmented state-space equation as follows:

$$\mathbf{X}(k+1) = \mathbf{\Omega}(k+1)\mathbf{X}(k), \tag{11}$$

where $\mathbf{X}(k) = [\mathbf{x}^{\mathrm{T}}(k), \mathbf{y}^{\mathrm{T}}(k-1), \ldots, \mathbf{y}^{\mathrm{T}}(k-j), \boldsymbol{\xi}^{\mathrm{T}}(k), \mathbf{u}^{\mathrm{T}}(k-1), \mathbf{u}^{\mathrm{T}}(k-l)]^{\mathrm{T}}$ is the augmented state vector, and $\mathbf{\Omega}(k+1)$ is the augmented state transition matrix computed from $\mathbf{\Phi}$, $\mathbf{\Gamma}$, $\mathbf{C}$, $\mathbf{F}$, $\mathbf{G}$, $\mathbf{H}$, and $\mathbf{J}$.

For periodic delays, there exists a positive integer M such that $\tau^{sc}_{k+M} = \tau^{sc}_k$. Using this property, Halevi and Ray (1988) proved that the system in (11) is asymptotically stable if all eigenvalues of $\Xi^M_k = \prod_{j=1}^{M} \mathbf{\Omega}(k+M-j)$ are contained within the unit circle. Ray and Halevi also suggested an approach to improve the networked control methodology by appropriately selecting Δ_k (Ray & Halevi, 1988).

3.2. Queuing methodology

Queuing mechanisms can be used to reshape random network delays on an NCS to deterministic delays such that the NCS becomes time-invariant. The methodologies to control an NCS that is based on queuing mechanisms are defined here as the queuing methodologies. These methodologies have been developed by utilizing some deterministic or probabilistic information of an NCS for the control algorithm formulation.

An early queuing methodology was developed by Luck and Ray (1990, 1994) denoted here as the deterministic predictor-based delay compensation methodology. This methodology uses an observer to estimate the plant states and a predictor to compute the predictive control based on past output measurements. The control and past output measurements are stored in a FIFO (First-In-First-Out) queue and a shift register defined as Q_1 and Q_2, where the sizes of Q_1 and Q_2 are μ and θ, respectively, as depicted in Fig. 7.

The steps for applying the delay compensation methodology are listed as follows:

- Using the set of past measurements $Z(k) = \{y(k-\phi), y(k-\phi-1), \ldots\}$ in Q_2, where ϕ is the number of packets in Q_2, the observer estimates the plant state $\hat{\mathbf{x}}(k-\theta+1)$.
- The predictor uses $\hat{\mathbf{x}}(k-\theta+1)$ to predict the future state $\hat{\mathbf{x}}(k+\mu)$.
- The controller computes the predictive control $\mathbf{u}(k+\mu)$ from $\hat{\mathbf{x}}(k+\mu)$, and then sends $\mathbf{u}(k+\mu)$ to be stored in Q_1.

Since the performances of the observer and the predictor highly depend on the model accuracy, the dynamic model of the plant has to be very precise.

Chan and Özgüner (1995) developed another queuing methodology for controlling an NCS on random delay networks. This methodology, named here as the probabilistic predictor-based delay compensation methodology, utilizes probabilistic information along with the number of packets in a queue to improve state prediction. Nevertheless, this queuing methodology itself is not really a control algorithm, but is more likely

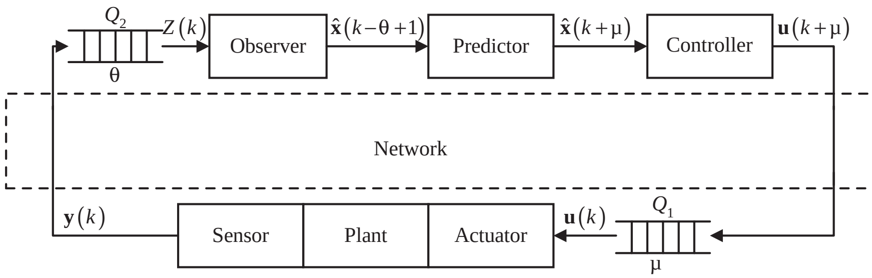

Fig. 7. Configuration of NCS in the deterministic predictor-based delay compensation methodology.

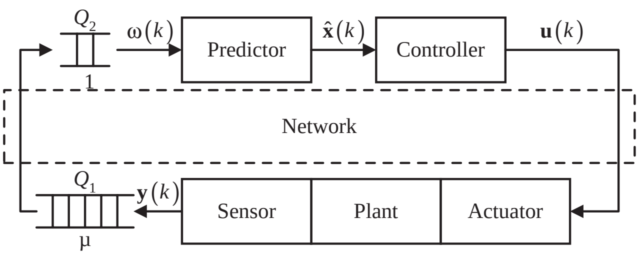

Fig. 8. Configuration of NCS in the probabilistic predictor-based delay compensation methodology.

a scheme to predict state variables. The configuration of the NCS in probabilistic predictor-based delay compensation methodology is illustrated in Fig. 8.

As shown in Fig. 8, the queue Q_1 at the sensor has a capacity of μ, while the shift register Q_2 can store only one packet. The output $\mathbf{y}(k)$ is stored in Q_1 waiting to be sent to Q_2 when the network is available for a transmission. To describe the compensation methodology, let the number of packets stored in Q_1 and the output from Q_2 be defined as i and $\boldsymbol{\omega}(k)$, respectively. At the sampling time k, if the sensor cannot send $\mathbf{y}(k)$ before Q_2 is read, $\boldsymbol{\omega}(k)$ is set to the previous value $\boldsymbol{\omega}(k-1)$. Otherwise, $\boldsymbol{\omega}(k)$ can be identical to any value in $\{\mathbf{y}(k), \mathbf{y}(k-1), \ldots, \mathbf{y}(k-\mu)\}$. However, the possible choices of $\boldsymbol{\omega}(k)$ can be reduced to either $\mathbf{y}(k-i)$ or $\mathbf{y}(k-i+1)$, if $i = 1, \ldots, \mu$, defined as the delay index, is known. This condition requires that the value of i has to be attached to every packet of $\mathbf{y}(k)$. The predictor then estimates the current state $\hat{\mathbf{x}}(k)$ by

$$\hat{\mathbf{x}}(k) = \mathbf{P}_0\left(\boldsymbol{\Phi}^{i-1}\omega(k) + \mathbf{W}_i\right) + \mathbf{P}_1\left(\boldsymbol{\Phi}^{i}\boldsymbol{\omega}(\mathrm{k}) + \mathbf{W}_{i+1}\right), \quad (12)$$

$$\mathbf{W}_i = \begin{cases} 0, & i = 1, \\ [\boldsymbol{\Gamma}, \boldsymbol{\Phi\Gamma}, \ldots, \boldsymbol{\Phi}^{i-2}\boldsymbol{\Gamma}] \cdot [\mathbf{u}^{\mathrm{T}}(k-1), \mathbf{u}^{\mathrm{T}}(k-2), \ldots, \mathbf{u}^{\mathrm{T}}(k-i+1)]^{\mathrm{T}}, & i \neq 1, \end{cases} \quad (13)$$

where $\mathbf{P}_0$ and $\mathbf{P}_1$ are weighting matrices. The weighting matrices are computed from the probabilities of the occurrences of $\mathbf{y}(k-i)$ and $\mathbf{y}(k-i+1)$. These equations require full state information (i.e., $\mathbf{y}(k) = \mathbf{x}(k)$). If the full state information is not available, an observer can also be applied with minor modification. With the predictive states, a control law from various control algorithms can be applied in this methodology.

3.3. Optimal stochastic control methodology

Nilsson (1998) proposed the optimal stochastic control methodology to control an NCS on random delay networks. The optimal stochastic control methodology treats the effects of random network delays in an NCS as a Linear–Quadratic–Gaussian (LQG) problem. Other than the assumptions mentioned earlier, this methodology assumes that $\tau < T$.

The dynamics of a remote system plant in this methodology is described by

$$\mathbf{x}(k+1) = \boldsymbol{\Phi}\mathbf{x}(k) + \boldsymbol{\Gamma}_0(\boldsymbol{\tau}_k)\mathbf{u}(k) + \boldsymbol{\Gamma}_1(\boldsymbol{\tau}_k)\mathbf{u}(k-1) + \mathbf{v}(k), \quad (14)$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{w}(k), \quad (15)$$

where $\boldsymbol{\tau}_k = \left[\tau_k^{sc}, \tau_k^{ca}\right]^{\mathrm{T}}$ indicates network delays at the sampling time k, $\boldsymbol{\Phi} = \exp(\mathbf{A}T)$, $\boldsymbol{\Gamma}_0(\boldsymbol{\tau}_k) = \int_0^{T-\tau_k^{sc}-\tau_k^{ca}} \exp(\mathbf{A}\zeta)\,\mathrm{d}\zeta\mathbf{B}$, and $\boldsymbol{\Gamma}_1(\boldsymbol{\tau}_k) = \int_{T-\tau_k^{sc}-\tau_k^{ca}}^{T} \exp(\mathbf{A}\zeta)\,\mathrm{d}\zeta\mathbf{B}$. The stochastic processes $\mathbf{v}(k)$ and $\mathbf{w}(k)$ are uncorrelated Gaussian white noises with zero means. These equations are modified from the constant delay system (Åström & Wittenmark, 1990).

The goal of the optimal stochastic control methodology is to minimize the following cost function in the case that full state information is available:

$$J(k) = E\left[\mathbf{x}^{\mathrm{T}}(N)\mathbf{Q}_N\mathbf{x}(N)\right] + E\sum_{k=0}^{N-1}\begin{bmatrix}\mathbf{x}(k)\\ \mathbf{u}(k)\end{bmatrix}^{\mathrm{T}}\mathbf{Q}\begin{bmatrix}\mathbf{x}(k)\\ \mathbf{u}(k)\end{bmatrix}, \quad (16)$$

where $E[\cdot]$ is the expected value, and $\mathbf{Q}_N$ and $\mathbf{Q}$ are weighting matrices. The control law for the optimal state feedback is derived by using dynamic programming and is described as

$$\mathbf{u}(k) = -\mathbf{L}(k, \boldsymbol{\tau}_k)\begin{bmatrix}\mathbf{x}(k)\\ \mathbf{u}(k-1)\end{bmatrix}, \tag{17}$$

where $\mathbf{L}$ is the optimal gain matrix after solving the formulated LQG problem. The network delay $\boldsymbol{\tau}_k$ is assumed to be independent. The past information of the delay is also required. If the full state information is not available, an optimal estimator such as the Kalman filter can be applied for (17). Nevertheless, this case requires the past information of output and input $\{\mathbf{y}(0), \ldots, \mathbf{y}(k), \mathbf{u}(0), \ldots, \mathbf{u}(k-1)\}$ in conjunction with the past information of the delay. Another control law to use with the delays modeled by a Markov Chain is also derived in the same study. Based on (16), the optimal stochastic control methodology has shown to give better performance than the deterministic predictor-based delay compensation methodology.

3.4. Perturbation methodology

Walsh, Beldiman, Ye, and Bushnell (1999a, 1999c) used non-linear and perturbation theory to formulate network delay effects in an NCS as the vanishing perturbation of a continuous-time system under the assumption that there is no observation noise. This methodology, denoted here as the perturbation methodology, can be applied on an NCS on periodic delay networks and random delay networks at the sensor-to-controller transmission. However, these networks are restricted to be *priority-based networks*, which can assign different priorities to data transmissions. These priorities can be managed by priority scheduling algorithms proposed in Walsh, Beldiman, and Bushnell (1999b). In addition, this methodology requires a very small sampling time so that an NCS can be approximated as a continuous-time system. A control loop in the perturbation methodology consists of a nonlinear controller and a nonlinear plant, but the analysis and derivations used can be similarly applied to linear systems, as described in Walsh, Ye, and Bushnell (1999c). Fig. 9 shows the block diagram of the NCS in this methodology.

The dynamics of the NCS in the perturbation methodology is represented by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{e}(t), t), \tag{18}$$

where $\mathbf{x}(t) = \left[\mathbf{x}_p^{\mathrm{T}}(t), \mathbf{x}_c^{\mathrm{T}}(t)\right]^{\mathrm{T}}$ is the augmented state vector containing the plant state vector $\mathbf{x}_p(t)$ and the controller state vector $\mathbf{x}_c(t)$. The error of the NCS is described by

$$\mathbf{e}(t) = \mathbf{y}(t) - \hat{\mathbf{y}}(t), \tag{19}$$

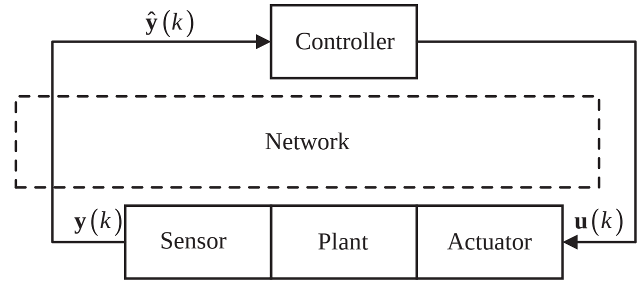

Fig. 9. Configuration of NCS in the perturbation methodology.

where $\mathbf{y}(t)$ is the plant output, and $\hat{\mathbf{y}}(t) \triangleq \mathbf{y}(t-\tau^{sc})$ is the most updated output which the controller receives. Also, $\mathbf{e}(t)$ is assumed to have certain dynamics as follows.

$$\dot{\mathbf{e}}(t) = \mathbf{g}(t, \mathbf{x}(t), \mathbf{e}(t)). \tag{20}$$

The dynamics equation (20) is treated as the vanishing perturbation to derive a delay bound ρ such that the NCS remains stable if $\tau^{sc} < \rho$.

3.5. Sampling time scheduling methodology

Hong (1995) developed the sampling time scheduling methodology to appropriately select a sampling period for an NCS such that network delays do not significantly affect the control system performance, and the NCS remains stable. This methodology is originally used for multiple NCS on a periodic delay network, in which all connections of every NCS on the network are known in advance. However, it was also modified to apply on random delay networks such as CAN (Hong & Kim, 2000). This methodology requires $\tau < T$, and is applicable to only a single-dimensional NCS.

To briefly describe the sampling time scheduling methodology, let the number of NCS on the network be M. The sampling times of all M NCS on the network are calculated from the sampling time of the most sensitive NCS based on the general frequency domain analysis on its worst-case delay bound. The most sensitive NCS, denoted as NCS_1, has the shortest delay bound defined as φ_1. The sampling time scheduling algorithm is formulated from the window concept illustrated in Fig. 10, where L and σ are the transmission periods of a pure data message and its overhead, respectively; T_1 is the sampling time of NCS_1, and r is the number of data messages that can be served by the network during the worst-case network traffic. The sampling time T_1 is computed from

$$T_1 = \frac{\varphi_1 + L}{3}. \tag{21}$$

In order to find the sampling times of other NCS on the same network, these systems have to be indexed from the worst-case delay bounds of the systems in an ascending order as $NCS_2, \ldots, NCS_M$. For example, the

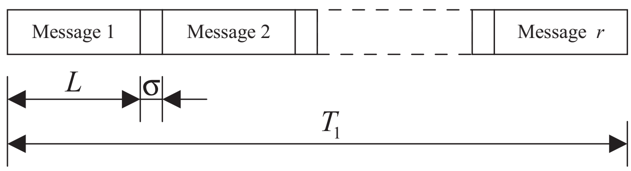

Fig. 10. Windows of data transmissions in the sampling period T_1 of the sampling time scheduling methodology.

system NCS_2 has the worst-case delay bound longer than the worst-case delay bound of NCS_1, but is shorter than then the worst-case delay bound of NCS_3. The sampling times of $NCS_2,\ldots,NCS_M$ are determined from T_1 using different rules with respect to network conditions. In a generic case, all other sampling times are multiples of T_1 as expressed by

$$T_i = k_i T_1, \quad i = 2, 3, \ldots, M, \tag{22}$$

$$k_i = \Lambda\left(\frac{\varphi_i - (T_1 - L)}{2T_1}\right), \tag{23}$$

where T_i is the sampling time of NCS_i, and $a = \Lambda(b)$ indicates that $a = 2^{v_i}, v_i \in \{0, 1, 2, \ldots\}$, which is the "closest" to, but does not exceed b.

In a special case, in which the number of NCS and other resources connected on the same network is less than r, the sampling times of $NCS_2,\ldots,\ NCS_M$ are determined by

$$T_i = \frac{\varphi_i - (T_1 - L)}{2}, \quad i = 2, 3, \ldots, M. \tag{24}$$

In addition, the optimality of the network utilization can be achieved by this methodology, which is an advantage among other methodologies. The condition for the optimality is

$$2\sum_{i=1}^{M} \frac{T_1}{T_i} = r. \tag{25}$$

Kim, Kwon, and Park (1996, 1998) enhanced the concept of sampling time scheduling to develop another algorithm for the multi-dimensional NCS. In this work, the delay bound of each system is obtained from different stability criteria. The dynamics of such a multi-dimensional NCS is briefly expressed as follows:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{A}_1\mathbf{x}(t - \tau^{sc}) \\ &\quad + \mathbf{A}_2\mathbf{x}(t - \tau^{sc} - \tau^{c}) + \mathbf{A}_3\mathbf{x}(t - \tau^{ca} - \tau^{c}),\end{aligned} \tag{26}$$

where $\mathbf{x}(t) = \left[\mathbf{x}_p^{\mathrm{T}}(t), \mathbf{x}_c^{\mathrm{T}}(t)\right]^{\mathrm{T}}$, $\mathbf{x}_P(t)$ is the plant state vector, $\mathbf{x}_c(t)$ is the controller state vector. The matrices $\mathbf{A}$, $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$ are calculated from the realizations of the plant and the controller. Two existing asymptotically stability criteria based on Lyapunov function can be used to find the delay bound in this generalized methodology.

3.6. Robust control methodology

Göktas (2000) designed a networked controller in the frequency domain using robust control theory. This methodology is denoted here as the robust control methodology. A major advantage of this methodology is that it does not require a priori information about the probability distributions of network delays. In the robust control methodology, the network delays τ^{ca} and τ^{sc} are modeled as simultaneous multiplicative perturbation. Both delays τ^{sc} and τ^{ca} are also assumed to be bounded and able to be approximated by the fluid-flow model (Filipiak, 1988). The network delay formulation is described as follows:

$$\begin{aligned}\tau^n &= \tfrac{1}{2}(\tau_{\max} + \tau_{\min}) + \tfrac{1}{2}(\tau_{\max} - \tau_{\min})\delta, \quad -1 \leqslant \delta \leqslant 1, \\ &= (1 - \alpha)\tau_{\max} + \alpha\tau_{\max}\delta, \quad 0 \leqslant \alpha \leqslant \tfrac{1}{2},\end{aligned} \tag{27}$$

where τ^n can be τ^{sc} and τ^{ca}, $\tau_{\max}$ is the upper bound of τ^n, $\tau_{\min}$ is the lower bound of τ^n, α and δ are real numbers to be determined based on an application. The first term in (27) represents a constant delay, whereas the second term represents the uncertain delay varying from the first term. The delay in (27) is converted for use in the frequency domain, and approximated by the first-order Padé approximation as

$$\begin{aligned}\mathrm{e}^{-\tau^n s} = \mathrm{e}^{-s(1-\alpha)\tau_{\max}}\mathrm{e}^{-s\alpha\tau_{\max}\delta} &\approx \frac{1 - s\tau^n/2}{1 + s\tau^n/2} \\ &\approx \left(\frac{1 - s(1-\alpha)\tau_{\max}/2}{1 + s(1-\alpha)\tau_{\max}/2}\right) \\ &\quad \times \left(\frac{1 - s\alpha\tau_{\max}\delta/2}{1 + s\alpha\tau_{\max}\delta/2}\right).\end{aligned} \tag{28}$$

The uncertain delay part is then treated as the simultaneous multiplicative perturbation expressed as follows:

$$\left(\frac{1 - s\alpha\tau_{\max}\delta/2}{1 + s\alpha\tau_{\max}\delta/2}\right) = 1 + W_m(s)\Delta. \tag{29}$$

where Δ is the perturbation function, and

$$W_m(s) = \frac{\alpha\tau_{\max}s}{1 + \alpha\tau_{\max}s/3.465}$$

is a multiplicative uncertainty weight which covers the uncertain delay. The factor 3.465 is selected based on a designer's preference. This formulation is then put in H_∞ framework, and μ-synthesis is used to design a continuous time controller $G_C(s)$ for a plant $G_P(s)$. The control loop in the robust control methodology using this formulation is depicted in Fig. 11, where $R(s)$, $U(s)$, $Y(s)$, and $E(s) = R(s) - Y(s)$ are the reference, control, output, and error signals in the frequency domain, respectively.

The controller is discretized using the bi-linear transformation on an actual network. The author also suggested an approach to apply the robust control methodology with network Quality-of-Service on an

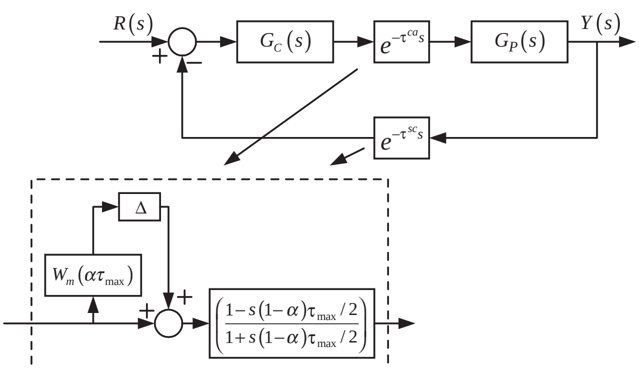

Fig. 11. Configuration of NCS in the robust control methodology.

ATM network in order to achieve the maximal tolerable error on a mobile robot application.

3.7. Fuzzy logic modulation methodology

Almutairi et al., (2001) proposed the fuzzy logic modulation methodology for an NCS with a linear plant and a modulated PI controller to compensate the network delay effects based on fuzzy logic (Zadeh, 1973). In this methodology, the PI controller gains are externally updated at the controller output with respect to the system output error caused by network delays. Thus, the PI controller needs not to be redesigned, modified, or interrupted for use on a network environment. A DC motor speed control problem is used to illustrate the proposed methodology. The system configuration of the fuzzy logic modulator methodology is depicted in Fig. 12, where $r(t), e(t), y(t)$, are the reference, error, and output of the system. The output of the PI controller is defined as $u_{PI}(t)$, and the modified PI controller output by the fuzzy logic modulation methodology is defined as $u_C(t)$.

The fuzzy logic modulation methodology can be implemented in a unit called the fuzzy logic modulator, which modifies the control $u_{PI}(t)$ by

$$u_C(t) = \beta\, u_{PI}(t) = \beta K_P e(t) + \beta K_I \int_{t_0}^{t} e(\xi)\,\mathrm{d}\xi. \tag{30}$$

The multiplicative factor β is used to externally adjust the controller gains at the output without interrupting the original PI controller. The value of β is selected from two fuzzy rules based on the network delay effects as follows:

If $e(t)$ is SMALL, then $\beta = \beta_1$,
If $e(t)$ is LARGE, then $\beta = \beta_2$,

where $0<\beta_1<\beta_2<1$. The membership functions of $e(t)$ are depicted in Fig. 13, where μ_{SMALL} and μ_{LARGE} are the membership functions representing the degrees of memberships for the linguistic variable SMALL and LARGE, respectively; α_1 and α_2 are factors to adjust the shapes of the membership functions. The shapes of the membership functions and the values of β_1 and β_2 are fine tuned by on-line and off-line optimization using the steepest descent algorithm based on cost functions. The cost functions for the on-line optimization are:

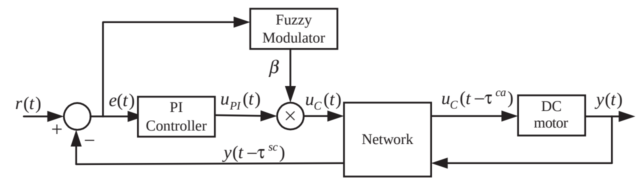

Fig. 12. Configuration of NCS in the fuzzy logic modulation methodology.

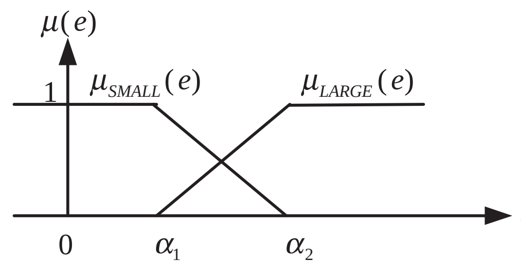

Fig. 13. Membership functions of $e(t)$.

$$J(k) = e^2(k), \tag{31}$$

$$J(k) = \sum_{i=k-m}^{k} e^2(i), \tag{32}$$

where the costs (31) and (32) indicate the instantaneous error, and the summing error evaluated from a moving window with the size of m. On the other hand, the off-line optimization uses the different cost function as follows:

$$\mathbf{J} = \lambda \mathbf{J}_1 + (1-\lambda)\mathbf{J}_2, \tag{33}$$

where

$$\mathbf{J}_1(\mathbf{p}) = \frac{\sum_{k=0}^{N} e(k)^2}{|\mathbf{J}_1(\mathbf{p})|_\infty}, \tag{34}$$

$$\mathbf{J}_2(\mathbf{p}) = \frac{\sum_{i=1}^{M} \Delta e_b(i)^2}{|\mathbf{J}_2(\mathbf{p})|_\infty} \tag{35}$$

and $\{\Delta e_b(i)\} = \{\Delta e(k) | e(k)\Delta e(k) > 0\}$. The cost $\mathbf{J}_1$ places the penalty on the system response time and poor convergence; the cost $\mathbf{J}_2$ gives the extra penalty on the system overshoot, undershoot, and oscillatory behaviors, and λ is a weighting factor. The parameter vector $\mathbf{p}$ represents the membership function parameters and β_1 and β_2.

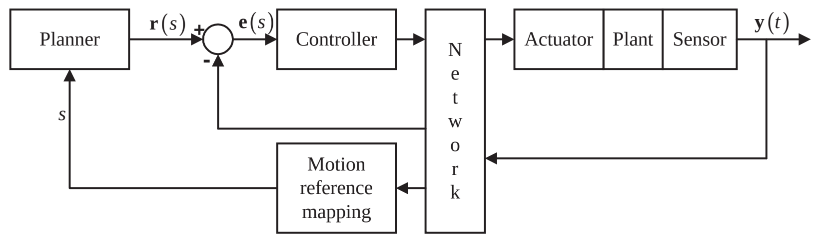

Fig. 14. Configuration of NCS in the event-based methodology.

3.8. Event-based methodology

Tarn and Xi (1998) introduced the event-based methodology for networked control of a robotic manipulator over the Internet. This methodology was originally developed for the hierarchical structure, but could be applied for the direct structure as well. The concept of the event-based methodology is quite different from all the previously mentioned methodologies. Instead of using time, this methodology uses a system motion as the reference of the system. The motion reference defined as s can be, for example, the distance traveled by an end-effector of a robotic manipulator. The motion reference s has to be a non-decreasing function of time in order to guarantee the system stability. The configuration of NCS in the event-based methodology is depicted in Fig. 14.

The output measurement $\mathbf{y}(t)$ sent across a network is used as an input for a motion reference mapping. The mapping converts $\mathbf{y}(t)$ to the motion reference s, which is then used as the input for the planner to compute the reference $\mathbf{r}(s)$. Thus, $\mathbf{r}(s)$ becomes a function of $\mathbf{y}(t)$, and is updated in real-time to compensate all disturbances and unexpected events including network delays. Because the overall system is not based on time, network delays will not destabilize the system.

3.9. End-user control adaptation methodology

Tipsuwan and Chow (2001) proposed the end-user control adaptation methodology. The main concept of end-user control adaptation is to adapt controller parameters (e.g., controller gains) with respect to the current network traffic condition or the current given network Quality-of-Service (QoS). In this methodology, the controller and the remote system are assumed to be able to measure network traffic conditions. The traffic condition measurement in this case could be measured through middleware (Li & Nahrstedt, 1999). The end-user control adaptation methodology is originally designed to cooperate with real-time QoS negotiation scheme (Abdelzaher, Atkins, & Shin, 2000), in which the controller can request and update network QoS requirements from the network. If the desired QoS requirements cannot be granted, the controller will adapt the parameters to aim for the best possible performance. The parameters are optimal with respect to the current traffic condition.

An application used to demonstrate the end-user control adaptation methodology is a DC motor speed control system controlled over a network link with random network delays. The DC motor speed is controlled by a PI controller, and the parameters to be adapted are the proportional gain K_P and the integral gain K_I. The system performance is measured by using the mean-squared error as follows:

$$J = \frac{1}{N}\sum_{k=1}^{N}|r(k) - y(k)|. \tag{36}$$

The network QoS measure used in this case is defined as $QoS^{(n)} = [QoS_1, QoS_2]^{\mathrm{T}}$, where n is the index to indicate a QoS condition, and

- $QoS1$: point-to-point network throughput.
- $QoS2$: point-to-point maximal delay bound of the largest packet.

The optimal controller parameters with respect to controller gains under different $QoS^{(n)}$ are pre-computed by simulations and stored in a look-up table. The controller gains will be updated when the network traffic condition changes. An example of the cost surface of (36) with respect to PI controller gains is shown in Fig. 15.

The authors illustrated the performance of the end-user control adaptation methodology by letting the network condition changes from $QoS^{(1)} = [38400$ bps, 5 ms] to $QoS^{(2)} = [19200$ bps, 8 ms] when the reference changes from 200 rad/s to 300 rad/s at $t = 3.5$ sec. The step response of the actual DC motor speed control system is shown in Fig. 16.

As shown in Fig. 16, the networked DC motor speed control system with control gain adaptation has superior performance than without gain adaptation as indicated by the lower overshoot. The end-user control gain adaptation methodology can also be applied for the hierarchical structure as shown in Tipsuwan and Chow (2002).

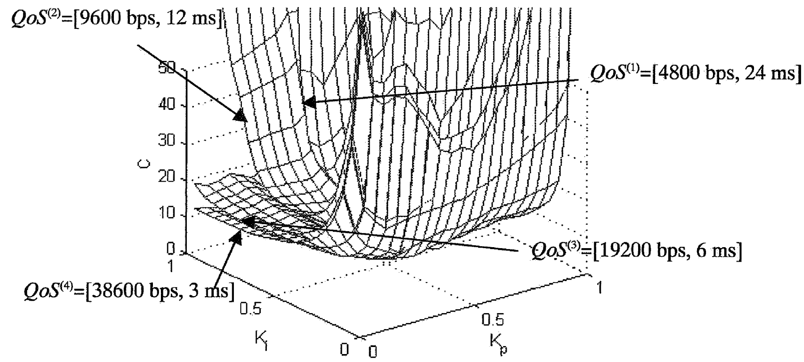

Fig. 15. Cost surface with respect to controller gains under different QoS conditions.

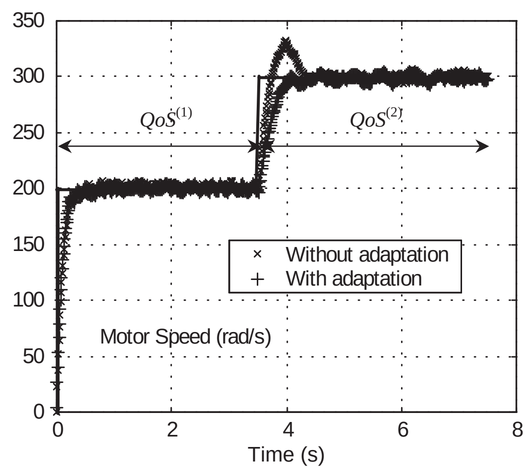

Fig. 16. Step responses of an actual networked DC motor speed control system in the end-user control adaptation methodology; ×: without adaptation, +: with adaptation.

4. Conclusion

This survey paper has introduced the fundamental and recent control methodologies for NCS. An NCS can be designed in the direct structure and/or the hierarchical structure depending on the application requirements and the designer's preferences. Regardless of the structure used, the system performance of NCS will degrade due to the existences of network delays in the control loop. In the worst case, the network delays can destabilize the NCS by reducing the system stability region. Random network delays in-the-loop are more difficult to handle than constant or periodic delays because there is no existing criterion to generally guarantee the stability of an NCS. Stability criteria for NCS are usually subject to specific methodologies and network protocols. Therefore, to design an NCS with a certain networked control methodology, a designer has to clearly understand an application whether it is feasible, acceptable, and reliable enough to be controlled by the methodology under a selected network protocol. There are also additional factors of concern including the price for the network protocol, and the size and distance of the application. The control methodologies described in this paper cover a large variety of systems and protocols. For example:

If a plant in NCS is linear, every methodology can be applied. However, if a plan is nonlinear, only the perturbation methodology, robust control methodology, and event-based control methodology can be used at this stage.

The queuing methodology should not be used on a cyclic service network since the methodology will result in longer delays unnecessarily.

If the network delays are unbounded, and the final time of the system is not critical, the event-based methodology is preferred because the system can remain stable.

The end-user adaptation methodology is preferred when the network QoS can be provided or monitored.

Even though the methodologies described in this paper are mostly applied on wired local area networks, the networked control applications and research can be extended on progressing network technologies including wireless networks, ad hoc networks, and Internet technologies. Furthermore, certain issues in NCS can be investigated such as the effect of packet loss on NCS, QoS requirements of NCS, stability analysis on various wired and wireless protocols in order to advance and strengthen networked control applications. With the rapid spreading of network applications to every place including homes, offices, and manufacturing plants, the research and reward in NCS could be substantial in the near future.

References

Abdelzaher, T. F., Atkins, E. M., & Shin, K. G. (2000). QoS negotiation in real-time systems and its application to automated flight control. *IEEE Transactions on Computers*, *49*(11), 1170–1183.

Almutairi, N.B., Chow, M.-Y., & Tipsuwan, Y., 2001. Network-based controlled DC motor with fuzzy compensation. *The 27th annual conference of the IEEE industrial electronics society* (*IECON 01*), Vol. 3 (pp. 1844–1849). Denver, CO.

Åström, K. J., & Wittenmark, B. (1990). *Computer-controlled systems: Theory and design*. Englewood Cliffs, NJ: Prentice-Hall.

Boustany, N., Folkerts, M., Rao, K., Ray, A., Troxel, L., & Zhang, Z., 1992. A simulation based methodology for analyzing network-based intelligent vehicle control systems. *Proceedings of the intelligent vehicles symposium* (pp. 138–143). Detroit, MI.

Chan, H., & Özgüner, Ü. (1995). Closed-loop control of systems over a communications network with queues. *International Journal of Control*, *62*(3), 493–510.

Filipiak, J. (1988). *Modelling and control of dynamic flows in communication networks*. Berlin: Springer.

Göktas, F., 2000. *Distributed control of systems over communication networks*. Ph.D. dissertation, University of Pennsylvania.

Halevi, Y., & Ray, A. (1988). Integrated communication and control systems: Part I—analysis. *Journal of Dynamic Systems, Measurement and Control*, *110*, 367–373.

Hong, S. H. (1995). Scheduling algorithm of data sampling times in the integrated communication and control systems. *IEEE Transactions on Control Systems Technology*, *3*(2), 225–230.

Hong, S. H., & Kim, W.-H. (2000). Bandwidth allocation scheme in CAN protocol. *IEE Proceedings—Control Theory and Applications*, *147*(1), 37–44.

Kaplan, G. (2001). Ethernet's winning ways. *IEEE Spectrum*, *38*(1), 113–115.

Kim, Y.H., Park, H.S., & Kwon, W.H., 1996. Stability and a scheduling method for network-based control systems. *The 22nd annual conference of the IEEE industrial electronics society* (*IECON 96*), Vol. 2 (pp. 934–939). Taipei, Taiwan.

Kim, Y.H., Park, H.S., & Kwon, W.H., 1998. A scheduling method for network-based control systems. *Proceedings of the 1998 American control conference*, Vol. 2 (pp. 718–722). Philadelphia, PA.

Kondraske, G. V., Volz, R. A., Johnson, D. H., Tesar, D., Trinkle, J. C., & Price, C. R. (1993). Network-based infrastructure for distributed remote operations and robotics research. *IEEE Transactions on Robotics and Automation*, *9*(5), 702–704.

Krtolica, R., Özgüner, Ü., Chan, H., Göktas, H., Winkelman, J., & Liubakka, M. (1994). Stability of linear feedback systems with random communication delays. *International Journal of Control*, *59*(4), 925–953.

Kuo, B. C. (1987). *Automatic control systems*. Englewood Cliffs, NJ: Prentice-Hall.

Li, Q., & Mills, D. L. (2001). Jitter-based delay-boundary prediction of wide-area networks. *IEEE/ACM Transactions on Networking*, *9*(5), 578–590.

Li, B., & Nahrstedt, K. (1999). A control-based middleware framework for quality-of-service adaptations. *IEEE Journal on Selected Areas in Communications*, *17*(9), 1632–1650.

Lian, F.-L., Moyne, J. R., & Tilbury, D. M. (2001). Performance evaluation of control networks: Ethernet, Controlnet, and DeviceNet. *IEEE Control Systems Magazine*, *21*(1), 66–83.

Liou, L.-W., & Ray, A. (1990). Integrated communication and control systems: Part III—nonidentical sensor and controller sampling. *Journal of Dynamic Systems, Measurement, and Control*, *112*, 357–364.

Luck, R., & Ray, A. (1990). An observer-based compensator for distributed delays. *Automatica*, *26*(5), 903–908.

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

Minoli, D., & Schmidt, A. (1999). *Internet architectures*. New York: Wiley.

Nilsson, J., 1998. *Real-time control systems with delays*. Ph.D. dissertation, Lund Institute of Technology.

Overstreet, J. W., & Tzes, A. (1999). An Internet-based real-time control engineering laboratory. *IEEE Control Systems Magazine*, *19*(5), 19–34.

Özgüner, Ü., Göktas, H., Chan, H., Winkelman, J., Liubakka, M., & Krotolica, R., 1992. Automotive suspension control through a computer communication network. *Proceedings of the 1992 IEEE conference on control applications*, Vol. 2 (pp. 895–900). Dayton, OH.

Ray, A. (1987). Performance evaluation of medium access control protocols for distributed digital avionics. *Journal of Dynamic systems, Measurement, and Control*, *109*, 370–377.

Ray, A., & Halevi, Y. (1988). Intergrated communication and control systems: Part II—design considerations. *Journal of Dynamic Systems, Measurement, and Control*, *110*, 374–381.

Shakkottai, S., Kumar, A., Karnik, A., & Anvekar, A. (2001). TCP performance over end-to-end rate control and stochastic available capacity. *IEEE/ACM Transactions on Networking*, *9*(4), 377–391.

Stallings, W. (2000). *Data & computer communication*. Upper Saddle River: Prentice-Hall.

Tarn, T.-J., & Xi, N., 1998. Planning and control of internet-based teleoperation. *Proceedings of SPIE: Telemanipulator and telepresence technologies V*, Vol. 3524 (pp. 189–193). Boston, MA.

Tipsuwan, Y., & Chow, M.-Y., 1999. Fuzzy logic microcontroller implementation for DC motor speed control. *The 25th annual conference of the IEEE industrial electronics society (IECON 99)*, Vol. 3 (pp. 1271–1276). San Jose, CA.

Tipsuwan, Y., & Chow, M.-Y., 2001. Network-based controller adaptation based on QoS negotiation and deterioration. *The 27th annual conference of the IEEE industrial electronics society (IECON 01)*, Vol. 3 (pp. 1794–1799). Denver, CO.

Tipsuwan, Y., & Chow, M.-Y., 2002. Gain adaptation of networked mobile robot to compensate QoS deterioration. *The 28th annual conference of the IEEE industrial electronics society (IECON 02)* 4 (pp. 3146–3151). Sevilla, Spain.

Walsh, G.C., Beldiman, O., & Bushnell, L., 1999a. Asymptotic behavior of networked control systems. *Proceedings of the 1999 IEEE international conference on control applications*, Vol. 2 (pp. 1448–1453). Kohala Coast, HI.

Walsh, G.C., Beldiman, O., & Bushnell, L., 1999b. Error encoding algorithms for networked control systems. *Proceedings of the 38th IEEE conference on decision and control*, Vol. 5 (pp. 4933–4938). Phoenix, AZ.

Walsh, G.C., Ye, H., & Bushnell, L., 1999c. Stability analysis of networked control systems. *Proceedings of the 1999 American control conference*, Vol. 4 (pp. 2876–2880). San Diego, CA.

Wargui, M., Tadjine, M., & Rachid, A., 1996. Stability of real time control of an autonomous mobile robot. *The fifth IEEE international workshop on robot and human communication* (pp. 311–316). Tsukuba, Japan.

Yook, J.K., Tilbury, D.M., & Soparkar, N.R., 2000. A design methodology for distributed control systems to optimize performance in the presence of time delays. *Proceedings of the 2000 American control conference*, Vol. 3 (pp. 1959–1964). Chicago, IL.

Zadeh, L. A. (1973). Outline of a new approach to the analysis complex systems and decision processes. *IEEE Transactions on Systems Man and Cybernatics* (SMC-3), 28–44.

Zhang, W., Branicky, M. S., & Phillips, S. M. (2001). Stability of networked control systems. *IEEE Control Systems Magazine*, *21*(1), 84–99.