Gain Adaptation of Networked DC Motor Controllers Based on QOS Variations

Mo-Yuen Chow, *Senior Member, IEEE*, and Yodyium Tipsuwan, *Student Member, IEEE*

***Abstract*—Connecting a complex control system with various sensors, actuators, and controllers as a networked control system by a shared data network can effectively reduce complicated wiring connections. This system is also easy to install and maintain. The recent trend is to use networked control systems for time-sensitive applications, such as remote dc motor actuation control. The performance of a networked control system can be improved if the network can guarantee Quality-of-Service (QoS). Due to time-varying network traffic demands and disturbances, QoS requirements provided by a network may change. In this case, a network has to reallocate its resources and may not be able to provide QoS requirements to a networked control application as needed. Therefore, the application may have to gracefully degrade its performance and perform the task as best as possible with the provided network QoS. This paper proposes a novel approach for networked dc motor control systems using controller gain adaptation to compensate for the changes in QoS requirements. Numerical and experimental simulations, and prototyping, are presented to demonstrate the feasibility of the proposed adaptation scheme to handle network QoS variation in a control loop. The effective results show the promising future of the use of gain adaptation in networked control applications.**

***Index Terms*—Adaptive control, communication networks, control systems, dc motors, decentralized control, distributed control, real-time systems, stability.**

I. Introduction

THE dc motors have been widely utilized in many industrial applications and have a large interest in the industrial electronics community for control applications. Thus, a dc motor system will be used to illustrate the gain adaptation in a networked control system through out this paper. In an application composed of a small number of dc motors, such as a robotic manipulator, a conveyor belt, or a computer numerical machine (CNC) milling machine, dc motors can be connected to controllers, drivers and sensors by directly wiring these devices together to perform closed-loop control without much complication. Nevertheless, direct wiring becomes more complicated and is not cost effective for installation and maintenance of a large number of dc motors in large-scale or complex systems such as automobiles, aircrafts, and manufacturing plants. The wiring can be organized systematically by applying a shared data network instead of hardwired connections. Moreover, this network-based wiring provides much more modularity, remote-control capability, and ease in diagnosis for the control systems.

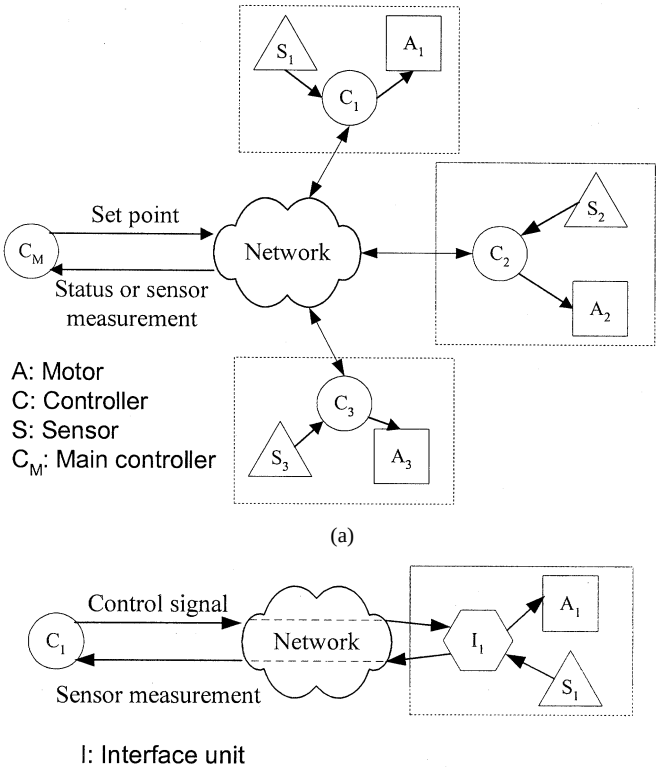


Fig. 1. Control system configurations using a shared data network. (a) Hierarchical sructure. (b) Direct structure.

Conceptually, there are two approaches to utilize a data network [1], [2] for control of dc motors. In the first approach, each dc motor has its own controller. The controller and the dc motor are physically located close to each other. This controller receives a set point remotely from the main controller through the network, and then uses the given set point to perform closed-loop control locally as shown in Fig. 1(a). A status or a sensor measurement of the DC motor is sent back to the main controller via the network. This approach provides modularity for each dc motor system. Each system is easy to be reconfigured. However, poor interaction between the main controller and each dc motor controller is a major drawback of this approach.

On the other hand, as shown in Fig. 1(b), the second approach uses a network as a medium to directly transfer control signals and sensor measurements between a dc motor and a controller. Closed-loop control of the dc motor is communicated over the

Manuscript received December 18, 2001; revised January 15, 2003. Abstract published on the Internet July 9, 2003. This work was supported in part by the Royal Thai Government.

The authors are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695-7911 USA (e-mail: chow@eos.ncsu.edu; ytipsuw@unity.ncsu.edu).

Digital Object Identifier 10.1109/TIE.2003.817576

network. Each dc motor in this case is attached to a simple interface unit (middleware). This unit converts a data frame from the controller to an actual control signal and the sensor measurement to a data frame for sending back to the controller. In fact, this interface can also be thought of as a simple remote controller. Systems formulated by this approach are so-called network-based or networked control system [3], [4], which can provide better interaction and higher flexibility for controlling dc motors.

Several standard industrial networks such as CAN, Profibus, etc. [5]–[7], have been widely used for networked control systems for years. Mostly, these networks have deterministic delays and enough bandwidth to perform networked closed-loop control using available control techniques without causing significant performance degradation or instability. However, investment on expensive network devices of these protocols is a problematic factor in today's competitive market environment.

Because of the affordability, simplicity, rapid development, and widespread usage, general-purpose networks such as Ethernet have been studied and suggested as alternatives for networked control applications [8]. Moreover, their connectivity to the Internet using Internet Protocol (IP) or Asynchronous Transfer Mode (ATM), can provide great benefits for remote access, control, and monitoring [9], [10] in industrial electronics [11], [12] and factory automation [13] applications. Applying networked control on these networks require more sophisticated control algorithms to handle random network delay problems. Various control techniques and stability criteria have been proposed to solve the delay problems. For examples, Luck and Ray suggested using buffers to reduce the variation of network delays [14]; Chan and Özgüner also used buffers [15], but they included some information about the amount of data in a queue to improve the results; Nilsson and Wittenmark formulated the effects of network delays as a Linear-Quadratic-Gaussian (LQG) problem and applied optimal control to handle the delays [16]; Walsh, *et al.* applied nonlinear control and perturbation theory to treat a network delay as a vanishing perturbation [17]; Hong introduced sampling time scheduling methods to obtain a large sampling time such that a networked control system remains stable [18]. Nevertheless, these algorithms require many assumptions, such as no communication error, which may not be feasible in real-world applications.

The networked control system performance does not only depend on the control algorithm used, but also on the network conditions. Several network conditions such as bandwidth, end-to-end delay, and packet loss rate are major impacts on networked control systems. The overall control performance can be improved if QoS (Quality-of-Services) requirements of these conditions can be provided. QoS can be viewed as bounds and limits of an end-to-end network application requirement on network conditions. The concepts of QoS have led to the development of many protocols such as ATM, RSVP, IP V6, and MPLS, which can provide even better platforms for networked control applications.

Due to changes in network user demands to or disturbances in network environments, such as the loss of a link, the availability of network resources may change unexpectedly. Therefore, the end-to-end control devices may have to renegotiate with the network resource counterparts for reallocation. If the QoS requirements cannot be provided as needed, the networked system may need to lower its performance and use the available QoS requirements to perform a control task as best as it can. In many cases, when anomalies happen, the first issue is to stabilize the closed-loop control system (with a network in the loop in this case), then aim for the best control performance under the given QoS conditions. Recent works on this issue have been extended to real-time QoS negotiation and graceful performance degradation [19], and the application of control theory on middleware QoS resource management and scheduling [20], [21].

This paper proposes a novel approach for networked DC motor control by using controller gain adaptation to compensate for the changes in QoS requirements through a real-time QoS negotiation process. In this paper, the gains of a networked PI controller are the main focus. These gains are adapted with respect to the given QoS conditions. A numerical simulation is set up to investigate the network delay issues and the performance of the adaptation scheme under fully control environments. We then develop a hardware prototype to verify the adaptation scheme experimentally. The successful results obtained from both numerical and experimental prototyping show the promising future of gain adaptation in networked control technologies for industrial applications and factory automation.

II. Problem Formulation

A networked control system can be divided into three parts: 1) the remote systems and remote controllers; 2) the central controller; and 3) the data network. A general block diagram of the networked control system under investigation is shown in Fig. 2. Each component is described in the following sections.

A. Remote System and Remote Controller

Each distributed remote controller can be assumed to have enough computing power to do relatively simple preprogrammed control—such as converting the control signal received from the central controller via the network into a pulsewidth-modulation (PWM) signal to drive a motor (the remote process). Each remote controller can send local measurements, such as motor current, speed, temperature, and local environment information, back to the central controller via the network. Each remote process has its own system dynamics that can be described by the state-space description [11] shown in (1), where the state vector $\mathbf{x}_R = [x_{R1}, \ldots, x_{Rn}]^T \in X^n$, the state space; $\mathbf{p}_R = [p_{R1}, \ldots, p_{Rq}]^T \in \Re^q$ are the system parameters; the input vector $\mathbf{u}_R = [u_{R1}, \ldots, u_{Rr}]^T \in U^r$, the input space; $t \in \Re^+$ is the time parameter; and $\mathbf{f}_R \in \Re^n$ is the state transfer function of the remote plant

$$\dot{\mathbf{x}}_R = \mathbf{f}_R(\mathbf{x}_R, \mathbf{p}_R, \mathbf{u}_R, t). \tag{1}$$

Depending on the design of the networked control system, the remote controller, C_R, performs a certain task, such as regulating the performance of the plant P_R, as described by (2)

$$\mathbf{u}_R = \mathbf{g}_R(\alpha_R, \cdot) \tag{2}$$

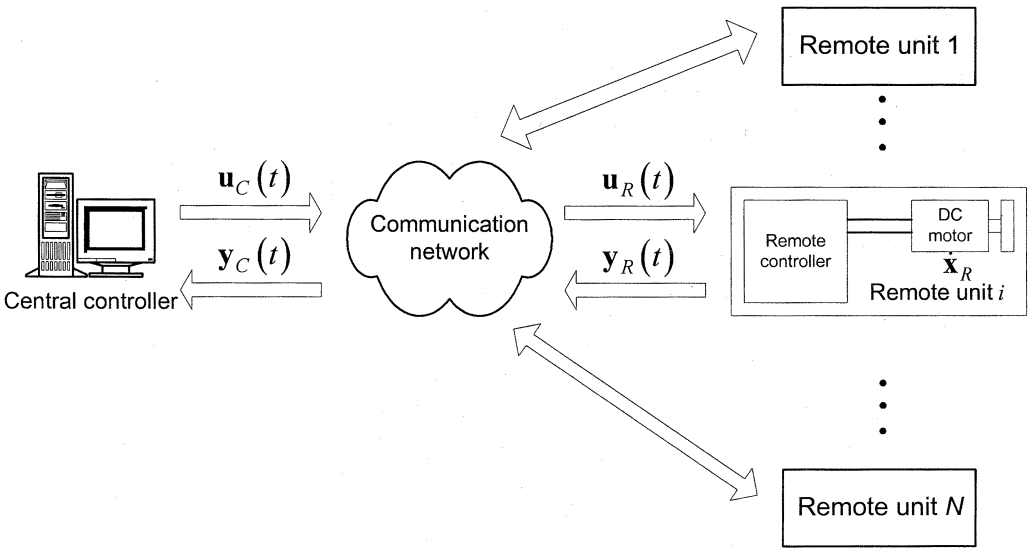


Fig. 2. An overall real-time networked control system.

where $\alpha_R = [\alpha_{R1}, \ldots, \alpha_{Ra}]^T$ is the adjustable controller parameter vector and $(\cdot)$ represents other appropriate information. The combination of the remote controller and the remote plant can be viewed as a remote system, S_R, which has its own system dynamics that can be described by a set of differential equations

$$\dot{\mathbf{x}}_R = \mathbf{f}_R(\mathbf{x}_R, \mathbf{p}_R, \mathbf{g}_R(\alpha_R, \cdot), t). \tag{3}$$

We denote the quality of service provided by the network as a function of time, as $QoS(t)$, which can be varied.

B. Central Controller

The central controller can be a highly sophisticated controller that requires lots of computing power and memory, and is therefore not suitable to be installed at the remote site. The central controller is powerful and can provide advanced real-time control laws to all remote units, including fault diagnosis and accommodation control, network traffic condition monitoring and adaptation to the QoS provided by the network. The central controller will provide the control signal $\mathbf{u}_C(t)$ to each of the remote systems. The control signal is assumed to be the result of optimizing a cost function C, described as

$$\min_{\mathbf{u}_C(t)} C(\mathbf{x}_R, \cdot). \tag{4}$$

Let z^{-t} be a time delay operator, and let $QoS(t)$ be the current QoS provided by the network. We define

$$\mathbf{u}_R(t) = \mathbf{u}_C\left(z^{-\tau_R}, QoS(t)\right) \tag{5}$$
$$\mathbf{y}_C(t) = \mathbf{y}_R\left(z^{-\tau_C}, QoS(t)\right) \tag{6}$$

where τ_R is the time delay in transmitting a signal from the central site to the remote site, and τ_C is the time delay in transmitting a signal from the remote site to the central site. The time delays $z^{-\tau}$ and $QoS(t)$ are functions of network variables such as the type and number of signals to be transmitted, the network traffic congestion condition, the network throughput, the network protocol used, the network management/policy used, and the controller processing time.

III. Case Study

In order to focus our discussion on how the QoS conditions can affect the closed-loop control performance, and how the central control can adapt its control parameters to compensate for the QoS variation and deficiency to provide the *best* control performance with a given QoS requirement, we select a simple central controller and a simple remote system (end-user to end-user) connected via a shared network with different QoS levels as a case study. With the simple networked control system, the effects of the QoS conditions are more obvious for illustration than a complicated system.

A. Remote System Description

This section briefly describes the dynamics of our remote process: a dc motor driving a load. The load can be a robot arm or an unmanned electric vehicle, for instance. The loop equation for the electrical circuit is [11], [12]

$$u(t) = e_a = L\frac{di_a}{dt} + Ri_a + e_b. \tag{7}$$

The mechanical torque balance based on Newton's law is

$$J\frac{d\omega}{dt} + B\omega + T_l = T_e = Ki_a \tag{8}$$

where $u = e_a$ is the armature winding input voltage; $e_b = K_b\omega$ is the back-electromotive-force (EMF) voltage; L is the armature winding inductance; i_a is the armature winding current; R is the armature winding resistance; J is the system moment of inertia; B is the system damping coefficient; K and K_b are the torque constant and the back-EMF constant, respectively; T_l is the load torque; and ω is the rotor angular speed.

By letting $x_1 = i_a$ and $x_2 = \omega$, the electromechanical dynamics of the dc motor can be described by the following state-space description:

$$\dot{x}_1 = -\frac{R}{L}x_1 - \frac{K_b}{L}x_2 + \frac{1}{L}u \tag{9}$$

TABLE I
DC MOTOR PARAMETERS

J	Inertia	42.6 e-6 Kg-m^2
L	Inductance	170 e-3 H
R	Resistance	4.67 Ω
B	Damping coefficient	47.3 e-6 N-m-sec/rad
K	Torque Constant	14.7 e-3 N-m/A
K_g	Back-EMF Constant	14.7 e-3 V-sec/rad

$$\dot{x}_2 = \frac{K}{J}x_1 - \frac{B}{J}x_2 - \frac{1}{J}T_l. \tag{10}$$

The parameters of the motor used in this paper are shown in Table I.

To keep the illustration simple, we assume that the remote controller is used to simply convert the control voltage data sent from the central controller into a PWM signal to drive the dc motor. The remote control value u_R can be mathematically expressed as

$$u_R(t) = u_C(t - \tau_R) \tag{11}$$

where τ_R is the time delay to transmit the control signal u_C from the central controller to the remote controller. The remote controller also sends the monitored signals $y_R(t)$ of the remote system back to the central controller, $y_C(t)$, and these two signals are related as

$$y_C(t) = y_R(t - \tau_C), \tag{12}$$

where τ_C is the time delay to transmit the measured signal from the remote controller to the central controller.

In fact, there are also processing delays, denoted as τ_{PC} and τ_{PR}, at the central and remote controllers, respectively. However, both τ_{PC} and τ_{PR} could be approximated as small constants, or even neglected because these delays are usually small compared to τ_C and τ_R.

B. Central Control System Description

The central controller will monitor the QoS conditions of each remote system link and provide appropriate control signals to each remote system. In this paper, the central controller uses a PI control algorithm to compute the control to the remote system for step tracking, based on the monitored system signals sent from the remote system via the network link. The proportional–integral (PI) controller used has the form [11], [12]

$$u(t) = K_P e(t) + K_I \int_0^t e(s)\,ds \tag{13}$$

where K_P is the proportional gain; K_I is the integral gain; $r(t)$ is the reference signal for the system to track; $y(t)$ is the system output; and $e(t) = r(t) - y(t)$ is the error function. In our case, $y = \omega$ is the motor speed, and $u(t)$ is the input voltage to the motor system.

C. Network Link QoS Conditions

There are different ways to define Quality-of-Service for end-to-end (from the central control to a specific remote system) user conditions. Two of the most popular QoS measures, which are used in this paper, are defined as follows.

- QoS_1 denotes the point-to-point (from the central controller to the remote controller) network throughput; it is used to indicate how fast the signal can be sampled and sent as a packet through the network.
- QoS_2 denotes the point-to-point maximal delay bound of the largest packet; it is used to indicate how long of a packet is expected to be delivered from the central controller to the remote controller.

One factor of interest in this paper is the sampling time h. In this paper, we set the periodic sampling time h with respect to the following criterion:

$$h > \tau_C + \tau_{PC} + \tau_R + \tau_{PR}. \tag{14}$$

This criterion is used to guarantee that packet transfers between the central and remote controllers can be processed in a sampling period. QoS_1 and QoS_2 are significant factors to determine h with knowledge of packet sizes used in order to satisfy the criterion. With given QoS_1 and QoS_2, τ_C and τ_R can be estimated by computing delays such as transmission delay and propagation delay. If this criterion cannot be satisfied, the central controller will not have the measured signals to process. This situation is not discussed in this paper.

IV. SIMULATION SETUPS

Both numerical and experimental simulations are set up to illustrate the effects of QoS variations on networked control, as explained in the following sections.

A. Numerical Simulation

In the numerical simulation scenario, the networked DC motor control system is simulated using MATLAB/SIMULINK under fully controlled environments. The motor equations in (9) and (10) are used as the main model, and it is controlled by the PI controller in (13) with the insertions of network delays according to (11) and (12). These delays can be varied according to different effects of interests. The numerical system setup is illustrated in Fig. 3.

B. Experimental Simulation

A simple peer-to-peer networked DC motor control system is set up to experimentally demonstrate the effects of interests. The block diagram of the networked system is shown Fig. 4(a), and the actual system setup is depicted in Fig. 4(b).

The central controller is implemented on a PC running RT-Linux, while the remote controller is implemented on a Siemens C-515C microcontroller board. Mainly, the remote controller is used for two tasks. The first task is to convert the instantaneous motor speed measurement to a packet and send it to the central controller. This packet is then used at the central controller to compute the input voltage by the PI control algorithm. The second task is to convert the input voltage in a packet form from the central controller to a duty cycle of a PWM signal, and then send it to the motor.

The network between the central and remote controllers in the setup is simulated by an RS-232 serial link. The baudrate of the serial link is assigned as QoS_1. Three packet formats used in this case are shown in Fig. 5.

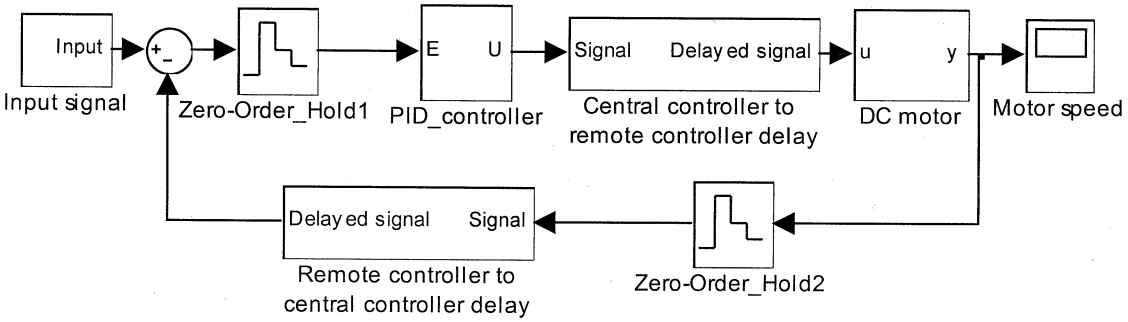
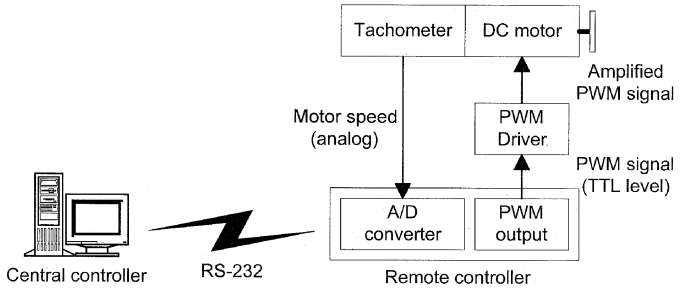
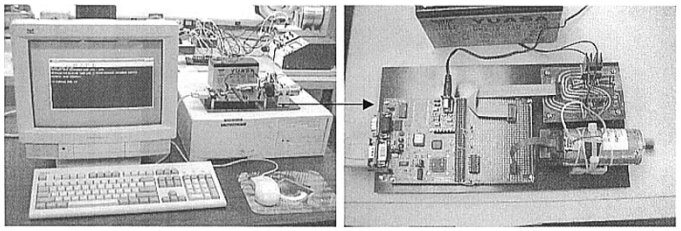


Fig. 3. Block diagram of the networked dc motor control system in the numerical simulation.



(a)



(b)

Fig. 4. (a) Block diagram of a peer-to-peer networked dc motor control system. (b) Actual networked dc motor control system setup.

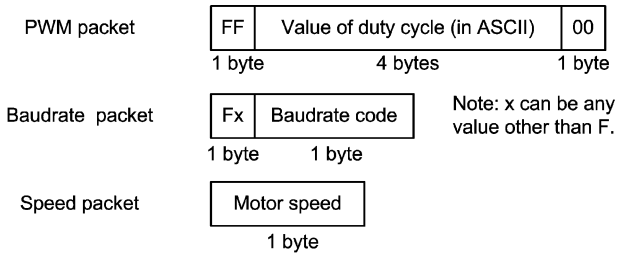


Fig. 5. Packet formats.

A PWM packet is used for sending the control voltage signal as the duty cycle of the PWM signal from the central controller to the remote controller. After the remote controller processes the PWM packet, it returns the motor speed using the speed packet format. The A/D converter used for sampling and quantization of the motor speed in the experimental setup has the resolution of 8 bits. Therefore, one byte is enough to represent the motor speed in the experimental simulation. On the other hand, the central controller sends the baudrate packet to the remote controller when the central and remote controllers are negotiating for a new baudrate used. Therefore, the maximal packet size with respect to our custom protocol is set to be 6 bytes, and

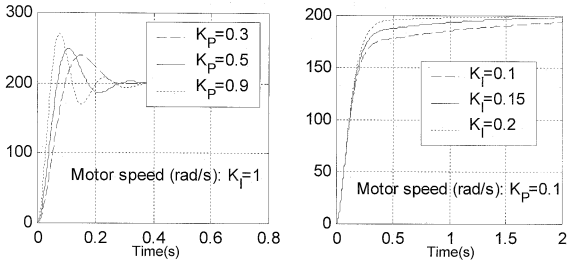


Fig. 6. Networked motor control performance with a sampling time of 2 ms and different PI gain values and without time delays.

QoS_2 in this case is determined by the maximal delay bound of the 6-byte packet transmission.

V. Simulation Results and Discussions

A. System Response With Different PI Gains

In the networked control system under consideration, both the control gains and the network QoS can significantly affect the closed-loop system performance. Fig. 6 shows typical closed-loop responses of a networked control system from the numerical simulation, subject to tracking a 200 rad/s reference signal with respect to different PI gains, given a sampling period of 2 ms and without time delays.

B. System Response With Different QoS

The closed-loop system will respond differently given a controller gain and different network QoS conditions. To demonstrate this case, we assume

$$QoS_2 = Maximal\ packet\ size(bits)/QoS_1. \tag{15}$$

Also, the sum of τ_{PC} and τ_{PR} is estimated to be 0.1 ms based on the equipment settings. Applying these estimations in conjunction with (14), the sampling time is estimated by

$$h > 2QoS_2 + 0.1\ \text{ms}. \tag{16}$$

With the largest packet size of 6 bytes, possible valid sampling times for $QoS_1 = \{4800,\ 9600,\ 19\,200$ and $38\,400\}$ bps can be set to $\{24,\ 12,\ 6,$ and $3\}$ ms, respectively. These settings are applied in the numerical simulation using a 6-byte packet size for the input voltage transmission and a single-byte size for the measured signal transmission. The network delays are set to be random, ranging from 90% to 100% of the maximal delays

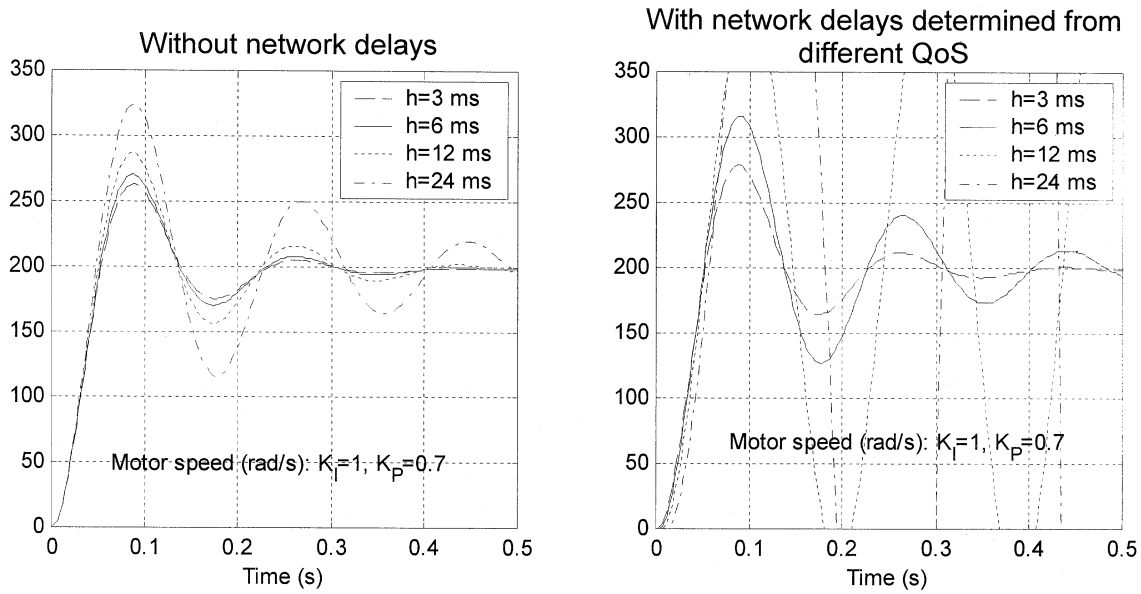


Fig. 7. Networked motor control performance given PI gain values under different QoS conditions.

with respect to the packet size used in different settings. This delay variation is generated using a uniform distribution. Fig. 7 clearly indicates that sampling times and network delays, which are determined from different QoS, can significantly influence the closed-loop control system performance by increasing the maximum overshoots and the settling times in the step responses, sometimes to the point of destabilizing the system.

C. Performance Measure and Gain Adaptation

In many cases, an end-user cannot control the network QoS, while the end-user can monitor the network QoS via appropriate middleware. Thus, the end-user can then adapt its controller gains to provide the best closed-loop control performance possible while the network QoS is varying or deteriorating. In this paper, we use a cost function approach to proactively adapt the PI gains in response to the change in network QoS. Different performance measures can be used to construct the cost function. In this paper, we define the cost function as

$$C = \frac{1}{N}\sum_{k=1}^{N} |r(k) - y(k)| \tag{17}$$

where N is the appropriate time index such that the tracking has arrived at the steady state, $r(k)$ and $y(k)$ are the measurements on r and y at time k. The cost function C considers how different the transient and steady-state system response is from the reference signal. Fig. 8 shows a typical cost for different PI gains under different QoS conditions, from the numerical simulation.

Each arrow points to a cost surface with respect to each QoS setting. The shading in Fig. 8 indicates the overlapped parts of the surfaces. The lighter shading of the surface lines implies more parts of surfaces overlap the lines. The figure clearly indicates that given a QoS and the maximal packet size, there are certain values of K_P and K_I that can provide the optimal cost, hence the optimal system performance. Different types of optimization techniques can be used to find the optimal values of K_P and K_I. For instance, we can construct the

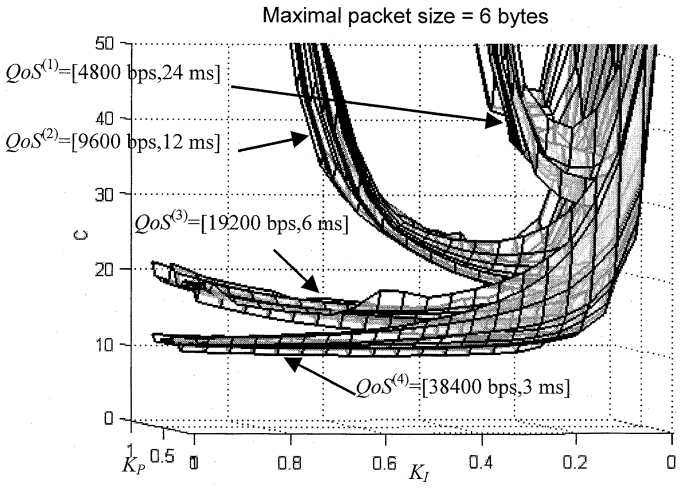


Fig. 8. Cost with respect to different PI gains and QoS.

cost function C with respect to the influential factors K_P, K_I, QoS_1 and QoS_2. Note that if there is a closed-form relationship among network QoS conditions, controller gains, and the cost function, the problem can be formulated as an optimal control problem such as the LQG problem. However, such the relationship is usually difficult to find and may not be able to obtain analytically.

Analytical searching techniques, such as steepest descent, may take a long time to find the optimal solution and may not be suitable for real-time applications. Other searching techniques, such as Newton–Raphson, may provide significant faster searching results, yet it can be sensitive to the cost function model error. In this paper, we use a lookup table technique to perform the searching and to show the feasibility of the proposed proactive gain adaptation algorithm with respect to network QoS. The cost values are stored in a table form with respect to QoS_1, QoS_2, K_P, and K_I. Using the monitored network QoS, we search for the best K_P and K_I from the stored table that can provide the optimal cost. Linear interpolation

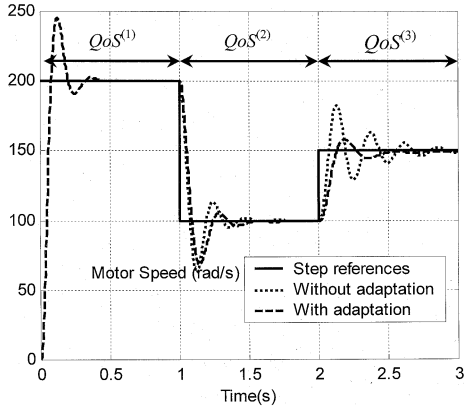


Fig. 9. Typical networked control system performance from the numerical setup with and without gain adaptation when network QoS deteriorates.

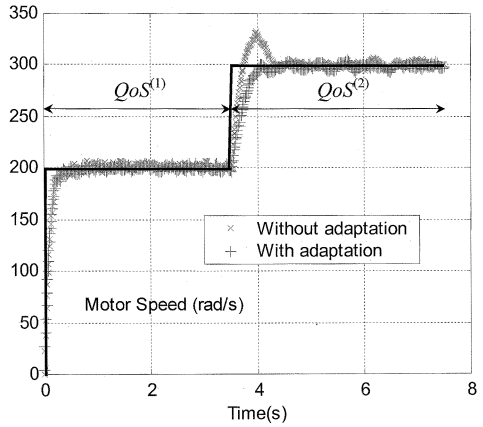


Fig. 10. Typical networked control system performance from the experimental setup with and without gain adaptation when network QoS deteriorates.

is used when the exact entry cannot be found from the table with the given inputs. For other high dimensional problems, the size of the lookup table may be very large due to the curse of dimensionality. Computational intelligence approaches such as fuzzy logic and neural networks can be applied to construct the mapping from the controller gains to the network QoS conditions by using the lookup table as a priori knowledge.

In order to show the effects of network QoS variations on the networked control system, different network QoS and control gains with and without adaptation were simulated and analyzed in both numerical and experimental simulations.

D. Numerical Simulation

Fig. 9 shows a typical networked control response from the numerical simulation when the network QoS deteriorates from $QoS^{(1)} = [QoS_1^{(1)}, QoS_2^{(1)}] = [38\,400\,\text{bps}, 3\,\text{ms}]$ to $QoS^{(2)} = [19\,200\ \text{bps}, 6\ \text{ms}]$, and to $QoS^{(3)} = [9600\,\text{bps}, 12\,\text{ms}]$. Step references are issued for the controller to track when the network changes QoS so that we can obviously see how the network QoS variation affects the networked control response with and without gain adaptation. The solid line represents the reference signal for the networked controller to track; the dotted line represents the closed-loop system response *without* gain adaptation; and the dashed line represents the closed-loop system response *with* gain adaptation to compensate for the QoS deterioration. The numerical simulation results clearly show the advantages and improved performance of using gain adaptation to maintain the system performance while the network QoS is varying or deteriorating.

E. Experimental Simulation

A networked control response from the experimental simulation is depicted in Fig. 10. Due to the processing delays at the central and remote controllers, the network QoS in this case is modified to deteriorate from $QoS^{(1)} = [38\,400\ \text{bps}, 5\ \text{ms}]$ to $QoS^{(2)} = [19\,200\ \text{bps}, 8\ \text{ms}]$ instead, with respect to the new step references. The solid line represents the reference signal for the networked controller to track; the cross-sign line represents the closed-loop system response *without* gain adaptation; and the plus-sign line represents the closed-loop system response *with* gain adaptation to compensate for the QoS deterioration. The experimental results explicitly support the advantages and performance of using gain adaptation.

VI. Conclusion

Networks and their applications are promising for wide deployment of real-time high performance networked control in industrial applications. However, one of the major concerns is the QoS that can be provided by the network and how it affects the performance of the networked control system. This paper has described, formulated, and shown promising results on the use of a proactive gain adaptation in a networked DC motor control system by the end-user in response to network QoS variations and deteriorations. Both the numerical and experimental results have shown that this is a promising and feasible approach in performing network QoS negotiation and graceful performance degradation and in maintaining networked control system availability. The analysis on more advanced issues such as packet losses can improve and strengthen the gain adaptation approach for more practical uses in the future.

References

[1] J. W. Overstreet and A. Tzes, "An Internet-based real-time control engineering laboratory," *IEEE Contr. Syst. Mag.*, vol. 19, pp. 19–34, Oct. 1999.

[2] G. V. Kondraske, R. A. Volz, D. H. Johnson, D. Tesar, J. C. Trinkle, and C. R. Price, "Network-based infrastructure for distributed remote operations and robotics research," *IEEE Trans. Robot. Automat.*, vol. 9, pp. 702–704, Oct. 1993.

[3] G. C. Walsh, O. Beldiman, and L. Bushnell, "Error encoding algorithms for networked control systems," in *Proc. 1999 IEEE Conf. Decision and Control*, vol. 5, Phoenix, AZ, 1999, pp. 4933–4938.

[4] Y. H. Kim, H. S. Park, and W. H. Kwon, "Stability and a scheduling method for network-based control systems," in *Proc. 1996 IEEE Int. Conf. Industrial Electronics, Control, and Instrumentation*, vol. 2, Taipei, Taiwan, R.O.C., 1996, pp. 934–939.

[5] J. M. Lee, S. Lee, M. H. Lee, and K. S. Yoon, "Integrated wiring system for construction equipment," *IEEE/ASME Trans. Mechatron.*, vol. 4, pp. 187–195, June 1999.

[6] P. Sink, "A comprehensive guide to industrial networks," *Sensors*, vol. 18, no. 6, pp. 28–43, 2001.

[7] W. T. Strayer and A. C. Weaver, "Performance measurement of data transfer services in MAP," *IEEE Network*, vol. 2, pp. 75–81, May 1988.

[8] G. Kaplan, "Ethernet's winning ways," *IEEE Spectr.*, vol. 38, pp. 113–115, Jan. 2001.

[9] A. C. Weaver, J. Luo, and X. Zhang, "Monitoring and control using the Internet and Java," in *1999 IEEE Int. Conf. Industrial Electronics, Control, and Instrumentation*, vol. 3, San Jose, CA, 1999, pp. 1152–1158.

[10] A. Malinowski, T. Konetski, B. Davis, and D. Schertz, "Web-controlled robotic manipulator using Java and client-server architecture," in *1999 IEEE Int. Conf. Industrial Electronics, Control, and Instrumentation*, vol. 2, San Jose, CA, 1999, pp. 827–830.

[11] Y. Tipsuwan and M.-Y. Chow, "Fuzzy logic microcontroller implementation for DC motor speed control," in *1999 IEEE Int. Conf. Industrial Electronics, Control, and Instrumentation*, vol. 3, San Jose, CA, 1999, pp. 1271–1276.

[12] J. T. Teeter, M.-Y. Chow, and J. J. Brickley Jr., "A novel fuzzy friction compensation approach to improve the performance of a DC motor control system," *IEEE Trans. Ind. Electron.*, vol. 43, pp. 113–120, Feb. 1996.

[13] R. Zurawski, "Verifying correctness of interfaces of design models of manufacturing systems using functional abstractions," *IEEE Trans. Ind. Electron.*, vol. 44, pp. 307–320, June 1997.

[14] R. Luck and A. Ray, "An observer-based compensator for distributed delays," *Automatica*, vol. 26, no. 5, pp. 903–908, 1990.

[15] H. Chan and Ü Özgüner, "Closed-loop control of systems over a communications network with queues," *Int. J. Control*, vol. 62, no. 3, pp. 493–510, 1995.

[16] J. Nilsson, B. Bernhardsson, and B. Wittenmark, "Stochastic analysis and control of real-time systems with random time delays," *Automatica*, vol. 34, no. 1, pp. 57–64, 1998.

[17] G. C. Walsh, O. Beldiman, and L. Bushnell, "Asymptotic behavior of networked control systems," in *Proc. IEEE Int. Conf. Control Applications*, vol. 2, Kohala Coast, HI, 1999, pp. 1448–1453.

[18] S. H. Hong, "Scheduling algorithm of data sampling times in the integrated communication and control systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 3, pp. 225–230, June 1995.

[19] T. F. Abdelzaher, E. M. Atkins, and K. G. Shin, "QoS negotiation in real-time systems and its application to automated flight control," *IEEE Trans. Comput.*, vol. 49, pp. 1170–1183, Nov. 2000.

[20] B. Li and K. Nahrstedt, "A control-based middleware framework for quality-of-service adaptations," *IEEE J. Select. Areas Commun.*, vol. 17, pp. 1632–1650, Sept. 1999.

[21] D. C. Schmidt, V. Kachroo, Y. Krishnamurthy, and F. Kuhns, "Developing next-generation distributed applications with QoS enabled DPE middleware," *IEEE Commun. Mag.*, vol. 38, pp. 112–123, Oct. 2000.

Mo-Yuen Chow (S'81–M'82–SM'93) received the B.S. degree in electrical and computer engineering from the University of Wisconsin, Madison, in 1982, and the M.Eng. and Ph.D. degrees from Cornell University, Ithaca, NY, 1983 and 1987, respectively.

Upon completion of the Ph.D. degree, he joined the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, as an Assistant Professor. He became an Associate Professor in 1993 and a Professor since 1999. His core technology is diagnosis and control, artificial neural network and fuzzy logic. Since 1987, he has been applying his core technology to areas including motors, process control, power systems, and communication systems. He has established the Advanced Diagnosis and Control (ADAC) Laboratory at North Carolina State University. He has authored one book, several book chapters, and over 90 journal and conference articles related to his research work.

Prof. Chow is an Associate Editor of the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, an AdCom member of the IEEE Industrial Electronics Society (IES). He served as the IES Vice President for Member Activities during 2000–2001.

Yodyium Tipsuwan (S'99) was born in Chiangmai, Thailand. He received the B.Eng. degree (with honors) in computer engineering in 1996 from Kasetsart University, Bangkok, Thailand, and the M.S. degree in electrical engineering in 1999 from North Carolina State University, Raleigh, where he is currently working toward the Ph.D. degree.

After receipt of the B.Eng. degree, he joined the Department of Computer Engineering, Kasetsart University, as an Instructor, mainly teaching digital system designs and microprocessor-based control systems. During his M.S. studies, he focused on intelligent control and its implementation. His main interests are distributed networked control systems, mobile robotics, and computational intelligence. His current research topic is networked control over IP networks.

Mr. Tipsuwan was awarded the Royal Thai Scholarship in 1998 to continue his graduate studies.