Real-Time Control of Networked Control Systems via Ethernet

Kun Ji and Won-jong Kim*

Abstract: In this paper, we discuss real-time control of networked control systems (NCSs) and practical issues in the choice of the communication networks for this purpose. An appropriate integration of control systems, real-time environments, and network communication systems allows the optimization of the quality-of-control (QoC) in NCSs. We compare several prevailing network types that may be used in control applications to offer a guideline of choosing a proper network. A real-time operating environment is also presented as an important ingredient of NCS design. To evaluate its feasibility and effectiveness, a real-time NCS containing a ball magnetic levitation (Maglev) setup is implemented via an Ethernet. Based on the experimental results, it is concluded in this paper that real-time control via Ethernet is a practical and feasible solution to NCS design.

Keywords: Networked control systems, control networks, real-time system, data-packet loss.

1. INTRODUCTION

Control systems where sensors, controllers, actuators, and other system components communicate over a network are referred to as NCSs [1]. The use of a communication network offers significant advantages in terms of reliability, enhanced resource utilization, reduced wiring, easier diagnosis and maintenance, and reconfigurability. However, implementing closed-loop control over a communication network introduces communication delays that inevitably degrade control performance and could even cause instability. Depending on network protocols and scheduling methods, network-induced delays have different characteristics and can be constant, time varying, or stochastic. In the design process, the interaction of the control system with the network must be considered in order to use the communication resources effectively.

The QoC of NCSs can be improved by using a modified control design strategy that takes into account the limited bandwidth of the network. For a given control law, the difference in performance between continuous-time and digital control is determined by the sampling period [2]. As the sampling period approaches zero, the performance of the digital system approaches that of a continuous-time system. For an NCS, performance is a function of not only the sampling period, but also the traffic load on the network [3]. An effective way to reduce the negative effect of delays on the performance of NCSs is to reduce network traffic. Using deadbands to reduce communication in NCSs was proposed as a solution for network traffic reduction in [4]. Yook *et al.* [5] formulated a method that offers network traffic reduction in exchange for added computational cost of using estimators to predict the states of other systems on the network. In [6], a traffic-smoothing method was proposed to reduce the packet-collision ratio. At each node, a traffic smoother is installed between the transport layer and the Ethernet medium access control (MAC) layer and regulates its packet stream using a certain traffic-generation rate. Hong [7] proposed a scheduling method that limits the loop delay as well as increases the utilization of network resources. The fundamental idea of this scheduling is to assign different sampling periods for different control loops based on the availability of network bandwidth. The work of extending the concept of the maximum allowable delay bound in [7] to the multidimensional cases was proposed in [8].

Regarding controller design, stability regions and the stability of NCSs were proposed using a hybrid systems technique in [9] and [10]. Ray *et al.* [11] formulated a state-estimation filter to account for random delays in the measurements. Ji *et al.* [12] extended the stochastic optimal estimator in [13] to be a multi-step-ahead state estimator to compensate for the time delay longer than one sampling period and successive data-packet losses simultaneously. In [14], a remote fuzzy-logic controller (RFLC) was proposed to compensate for the network-induced delays for a

Manuscript received June 22, 2005; revised September 15, 2005; accepted September 20, 2005. Recommended by Editor Keum-Shik Hong. The authors thank Ajit Ambike and Stephen Paschall for their contribution in this research project. This work is based upon work supported by the National Science Foundation under Grant No. CMS-0116642.

Kun Ji and Won-jong Kim are with the Department of Mechanical Engineering, Texas A&M University, College Station, TX 77843-3123, U.S.A. (e-mails: {kunji, wjkim} @tamu.edu).

* Corresponding author.

single-input-single-output (SISO) plant.

The functionality of an NCS can be described as the sequence of four main operations (sampling, computation, data transmission, and actuation) that have to be repeatedly executed, keeping a strict timing, in order to deliver the expected performance. Not only the network could induce delays, but also the devices connected to the network could introduce latencies if they are not in a real-time operating environment. Therefore, to achieve the timing constraints and guarantee the performance of NCSs, network-traffic optimization algorithms and real-time control design for the devices connected to the network must be developed. Thus the successful design and implementation of an NCS requires an appropriate integration of co-design of the real-time control system and the network communication system. The design and implementation of closed-loop real-time control over network is the main focus of this paper.

This paper is organized as follows. The problem statement is presented in Section 2. In Section 3, the characteristics of common network protocol types are elaborated. Real-time control system design is presented in Section 4. A real-time control system for ball Maglev control is implemented via an Ethernet in Section 5.

2. SYSTEM MODEL AND PROBLEM STATEMENT

2.1. Modeling of NCSs

The general framework of an NCS is shown in Fig. 1. It can be a SISO system or a MIMO system.

Under the consideration of network-induced time-delay and data-packet-dropout phenomena in the data transmission, the dynamics of the NCS shown in Fig. 1 can be described by the following model.

$$\begin{aligned} &\dot{x}(t) = Ax(t) + Bu(t) + v(t), \\ &y(t) = Cx(t) + Du(t) + w(t), \\ &u(t) = u(h_k), \qquad t \in [h_k + \tau_k, \quad h_{k+1} + \tau_{k+1}), \end{aligned} \tag{1}$$

where $x(t) \in R^n$ is the state, $u(t) \in R^m$ is the control input, $y(t) \in R^q$ is the output, and A, B, C, and D are known real constant matrices of appropriate dimensions. The sensor sampling period is h, h_k denotes a certain sampling instant with $h_k = i_k h$, $k = 0, 1, 2, \ldots$, where k is the index of the sampling instants, and $i_k \in \{0, 1, 2, 3, \ldots\}$ is the index of the arrived packets at the actuator node. It is not required that $i_k < i_{k+1}$, which is discussed in Remark 1 below, thus i_k can be different from k. The parameter τ_k denotes the time duration from the instant k when sensor samples data to the instant when the actuator actuates the control input. We have $\tau_k = t_{sc} + t_{ca}$, i.e., the sum of the sensor-to-controller and controller-to-actuator delays for a fixed control law [9]. The process noise $v(t)$ and the sensor noise $w(t)$ are zero-mean white Gaussian noises (WGNs).

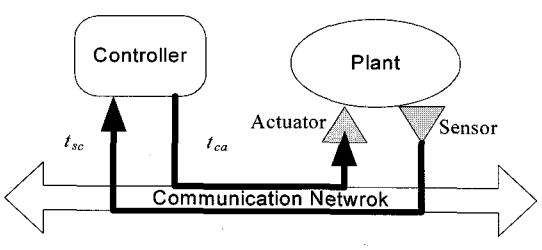


Fig. 1. NCS framework.

Let t_0 denote the starting time of the control system. Then we define the following initial condition function.

$$x(t) = \theta(t), \quad t \in [t_0 - \tau, \quad t_0), \tag{2}$$

where τ is the upper bound of τ_k, and we assume $\tau = Hh$.

Remark 1: If $\{i_0, i_1, i_2, \ldots, i_k\} = \{0, 1, 2, \ldots, k\}$, then there is no data-packet loss in the data transmission. Otherwise, the missing integers in the set $\{0, 1, 2, \ldots, k\}$ indicate the lost data-packets. If $i_k > i_{k+1}$, then there is out-of-order data transmission between the i_k-th data packet and the i_{k+1}-th data packet.

Two types of controller are considered.

1. Full-state-feedback controller:

$$u(t) = Kx(h_k), \quad t \in [h_k + \tau_k, \quad h_{k+1} + \tau_{k+1}), \tag{3}$$

2. Estimated-state-feedback controller:

$$\begin{aligned} &\dot{\hat{x}}(t) = (A - LC)\hat{x}(t) + \begin{bmatrix} B - LD & L \end{bmatrix} \begin{bmatrix} u(t) \\ y(t) \end{bmatrix}, \\ &u(t) = K\hat{x}(h_k), \quad t \in [h_k + \tau_k, \quad h_{k+1} + \tau_{k+1}), \end{aligned} \tag{4}$$

where $K \in R^{m \times n}$ is a constant matrix.

Remark 2: The control input $u(t)$ to the plant is piecewise constant during a sampling interval $[h_k, \quad h_{k+1})$. There can be at most $H + 1$ control inputs in one sampling interval.

In the sampling interval $[h_k, \quad h_{k+1})$, the control inputs arrive at the actuator node at the random instants $h_k + t_k^j$, where $0 \le t_k^j \le h, j \le H$. We use the following equation to discretize the continuous-time plant dynamic model (1) [2].

$$x(t) = \exp(A(t - t_0))x(t_0) + \int_{t_0}^{t} \exp(A(t - s))Bu(s)ds \tag{5}$$

Substituting (5) to (1) with $t = h_{k+1}$ and $t_0 = h_k$ yields

$$x(i_{k+1}) = A_d(h)x(i_k) + \sum_{j=0}^{l} B_{i_k}^{j}(h)u(i_{k-i}) + v(i_k), \qquad (6)$$

$$y(i_k) = Cx(i_k) + Du(i_k) + w(i_k), \quad l = 1, 2, ..., H,$$

where

$$A_d(h) = e^{Ah}, B_{i_k}^{j}(h) = \int_{t_k^j}^{t_k^{j-1}} \exp(A(h-s))dsB, t_k^l = 0, t_k^{-1} = h,$$

$$v(i_k) = \int_{h_k}^{h_{k+1}} e^{A(h_{k+1}-s)}v(s)ds, \quad w(i_k) = w(h_k),$$

$v(i_k)$ and $w(i_k)$ are still zero-mean WGNs.

2.2. Control network and real-time control system co-design

In NCSs, decision and control functions can be distributed on the network. Thus, a new constraint must be accommodated in the design of an NCS—the limited bandwidth of the communication network. To reduce the time delay caused by device processing, these control functions should have certain deadlines. If some of these deadlines are missed, the stability and performance of the NCS could be negatively affected. Thus there is a need of real-time operating systems for the devices to ensure these time-constrained events do not miss their deadlines.

3. CONTROL NETWORK SELECTION

It is necessary to understand the protocol message prioritization as well as the delay characteristics of the control networks when designing NCSs. Generally, the requirement for the control network is that a message should be transmitted successfully within a bounded time delay [15]. Compared with data networks, control networks have some distinguishing characteristics:

1) Most of the communications between controllers and sensors/actuators have a fixed sampling period and data are transmitted continuously, and thus the transmission rate is high.
2) Data size of each message is relatively small.
3) Since time delay has a serious effect on system performance, the real-time requirement of control networks is much more critical than that of data networks.

Control networks for NCSs' purpose are based on the following protocols: Ethernet (IEEE 802.3), Token Bus (IEEE 802.4), Token Ring (IEEE 802.5), CAN (ISO 11898), and Wireless (IEEE 802.11). The characteristics of the prominent control network types (Token-based, CAN-based, Ethernet-based, and Wireless-based) are presented in this section to explain how they influence the NCS performance. A more detailed comparison among Ethernet, Token-based, and CAN protocols can be found in [15].

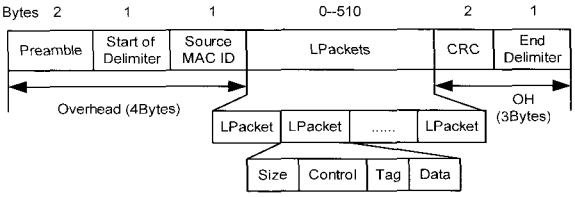


Fig. 2. Message frame of ControlNet [15].

3.1. Prominent candidate control network types

3.1.1 ControlNet as a token passing network

ControlNet is a deterministic network protocol used for time/mission critical applications. The message frame of ControlNet is shown in Fig. 2 [15]. In ControlNet, all nodes are arranged on a ring. A token is passed around the ring, and the node that holds the token is allowed to transmit data. This transmission continues until it is finished or a time limit is reached, then the token is regenerated and passed on to the next node. Thus the maximum waiting time in a node before sending a message is the token rotation time. Message collisions never occur since only one node can transmit a message at one time. In general, ControlNet is very efficient at high network loads and its deterministic nature defines a maximum bound on network-induced delays which makes time delay analysis easier. However at low network loads, a significant amount of time is spent passing the token around the logical ring [15]. In the case of an emergency, a node cannot gain access to transmit a message until the token finishes its rotation around the logical ring. Under light to moderate network loads, ControlNet provides good and fair performance during increased loads. A refined token-based algorithm that assures fair accesses with deterministic waiting time delays was proposed in [16].

3.1.2 CAN and DeviceNet

In the CAN protocol, data are transmitted with message frames shown in Fig. 3 [17], and a message may be transmitted periodically, sporadically, or on demand [17]. Each message is given a priority that determines its network access, and collisions do not destroy messages since the message with higher priority is delivered. Each message used in the CAN has a unique identifier defining the type of data, and the identifier also automatically implies the message priority for bus access [17]. However, the identifier does not indicate the destination or source address information in a message. If a node wants to transmit a message, it waits until the bus is free and then

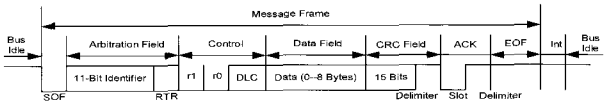


Fig. 3. Message frame of CAN [17].

broadcasts the identifier of its message. Each node has an acceptance filter to decide whether to receive that message or not. A message is accepted only if an identifier of the message object is matched with the incoming message identifier [17]. Thus, a bound on the time delay for higher-priority messages can be defined and used in analysis. CAN is not suitable for transmitting messages of large size although large messages can be transmitted using fragmentation [15]. DeviceNet is based on the CAN-bus protocol but does not use the same physical-layer interface as ISO 11898. It is developed originally for the automotive industry, and each message has a unique identifier defining the type of data such as engine speed, temperature, pressure or any other data. DeviceNet has a slow data rate of only 500 kbps with up to 64 devices on the bus [15].

3.1.3 Ethernet-based networks

Ethernet does not support message prioritization and is not a deterministic protocol. Ethernet uses the carrier-sense multiple access with collision detection (CSMA/CD) mechanism to resolve the problem of contention in case of simultaneous data transmission [15]. If two nodes transmit data packets simultaneously, the packets collide. If a collision is detected, the two transmitting nodes wait a random length of time to retry transmission. The random length of time is determined by the standard binary-exponential-backoff (BEB) algorithm. If 16 collisions are detected, the node stops transmitting, and then data-packet losses occur [18]. Thus in modeling, both time delays and packet loss need to be considered. Since Ethernet uses a simple algorithm for network operation, delays are small at low network loads [15]. Unlike ControlNet or DeviceNet, communication bandwidth is not wasted in message arbitration and message collisions lead to message loss in Ethernet [15]. The large frame size also makes Ethernet better suited to transmit large-size data with low frequency. The message frame of Ethernet is shown in Fig. 4.

Ethernet's contention-based mechanism makes it impossible to predict the network-induced delay. Switched Ethernet can provide deterministic delays by eliminating message collisions, but its high price has restricted its implementation in industry [6]. Several software changes have been suggested so that the network-induced delay could be bounded. Kweon *et al.* [6] developed a traffic-smoothing method to

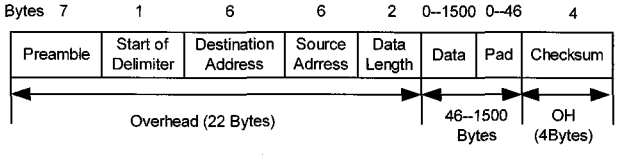


Fig. 4. Message frame of Ethernet [18].

decrease the packet-collision ratio on the network which requires minimal changes in the OS kernel. The traffic smoother regulates the node's packet stream using a certain traffic-generation rate to eliminate collisions effectively. Venkatramani and Chiueh [19] proposed the approach of a timed-token bus to provide bandwidth guarantees. They proposed a software-based protocol called RETHER (Real-time Ethernet).

3.1.4 Wireless networks

Wireless networks have also been investigated as control networks. The performance analysis of the wireless-medium-access-control (WMAC) protocol and the remote frame medium access control (RFMAC) protocol for a wireless controller area network (WCAN) was presented in [17]. In [20], the wireless Ethernet (802.11) standard was modified and used to prioritize the carrier sense multi-access with collision avoidance (CSMA/CA), and thus message collisions were reduced. Later work [21] applied the wireless 802.11 control and scheduling algorithm to the control of a physical plant. Ploplys *et al.* [22] developed a distributed-control system for a Furuta pendulum over a wireless communication network based on 802.11b.

3.2. NCS via Ethernet

Ethernet is potentially the most practical network solution because of its low cost, availability, and higher communication rates, although it is not designed to transmit short messages with real-time requirements. In terms of implementation, Ethernet is not yet ready for manufacturing automation because its hardware was not designed to withstand stress, vibrations, or noise [23], and it has unpredictable delay and packet loss characteristics. However, under low traffic loads, Ethernet delivers fast data transmissions with almost no latency [15]. As a result, an Ethernet network structure may be suitable for control when the network is relatively uncrowded. An example using Ethernet in network control is given in [24], where the user datagram protocol (UDP) over a switched Ethernet was shown to exhibit good performance characteristics that were sufficient for substation automation. Based on the NCS framework shown in Fig. 1, we designed and implemented a real-time closed-loop control system via an Ethernet and present experimental results in Section 5.

3.3. UDP and TCP

Selection of network protocol for communication is an important part of NCS design. The two dominant choices are TCP and UDP. TCP was specifically designed to provide a reliable end-to-end byte streams over any unreliable network [18]. TCP provides various services like stream data transfer, reliability, efficient flow control, full duplex operation,

Table 1. General properties of TCP and UDP.

	TCP	UDP
Message Boundaries	No	Yes
Connection Oriented	Yes	No
Positive Acknowledgement	Yes	No
Data Checksum	Yes	Optional
Timeout and Retransmission	Yes	No
Duplicate Detection	Yes	No
Flow Control	Yes	No

multiplexing, etc. Handshaking signals are used for making and breaking the connection. Parameters such as sequence numbers are initialized to help ensure ordered delivery and robustness. If time delay is encountered, the data is retransmitted from the sender. Timers and acknowledgment messages are used to detect this time delay or data loss. Check sums are used to detect data corruptions that might occur during the data transfer. Although TCP is a very reliable protocol, it has some disadvantages. Due to various services such as error checking and ordered and reliable data delivery, it has large overheads leading to time delay in the communication. In the event of congestion, the data are lost more frequently, so more retransmissions are done by TCP, increasing the overheads. For closed-loop control over the network the added reliability provided by TCP may not be worth the cost of the network delays it introduces.

UDP is an alternative provided by the TCP/IP protocol suite. Data transfer with UDP is not connection oriented. UDP does not provide additional services such as ensuring ordered data delivery and robustness as provided by TCP. It is therefore known as a best-effort network protocol. Although UDP is less reliable, it has fewer overheads and introduces less network delays. Ploplys *et al.* [22] concluded that the UDP, an unreliable but faster protocol, was better suited for real-time control over a dedicated wireless computer network. The general properties of TCP and UDP are compared in Table 1.

4. REAL-TIME OPERATION ENVIRONMENT FOR NCS

4.1. Selection of real-time operating environment

After the natures of available networks were studied, it was concluded in the previous section that an appropriate real-time OS is required to reduce the time delay caused by device processing and to successfully implement a distributed architecture. Commercially available OSs such as Windows 2000, various versions of Unix and Linux are not real-time OSs. The Linux real-time application interface (RTAI) [25] was developed as a real-time operating environment solution at Dipartimento di Ingeneria Aerospaziale Politecnico di Milano (DIAPM). RTAI modifies the Linux kernel to make it a real-time operating environment. RTAI offers the same services as the Linux kernel core, adding the features of a real-time OS. Compared to the commercially available real-time OSs, RTAI's performance is very competitive [25]. Table 2 summarizes the typical performance of RTAI. Last but not the least, RTAI is open-source software and free under the terms of the GNU (GNU is Not Unix) Lesser General Public License.

Table 2. RTAI's performance.

Context switch time	4 μs
Maximum periodic task rate	100 kHz
One-shot task rate	30kHz

Two timing tests were performed to observe the difference of the performances between RTAI and non-real-time OSs. The following two paragraphs present the results of these two tests.

The smallest amount of time that can be precisely measured on an OS is known as its clock resolution. The time required to read a clock is typically much less than its resolution, and many consecutive clock access functions can be executed before the value returned by the clock changes. This principle was used in the first test. The number of times the clock was accessed before the change in value returned by the clock access function was recorded and then plotted. Fig. 5 represents the results of the first test on Windows 2000, Redhat Linux 7.3, and Redhat Linux 7.3 with RTAI 24.1.12. Significant variations in Fig. 5(a) and (b) indicate some other OS activities that are not deterministic. In Fig. 5(c), the straight line denotes that there was no significant non-deterministic OS activity in Linux with RTAI.

When the value returned by the clock access function changes, the difference between this new value returned and the previous value returned is the clock resolution. In the second test, the clock resolution was calculated and plotted over several iterations. It was found that there were variations in the values returned as clock resolution. Thus, maximum value is considered the clock resolution for the corresponding OS. Fig. 6 presents the results of the second test on Windows 2000, Redhat Linux 7.3, and Redhat Linux 7.3 with RTAI 24.1.12, respectively. Whereas the clock resolution of Redhat Linux 7.3 was found to be uniform, the resolutions of Windows 2000 and RTAI were not. From Fig. 6, we can see that the clock resolution of RTAI is much better than that of Windows and Linux alone. Although the spikes in Fig 6(c) denote the variation in the clock resolution from 1 μs to 4.1 μs, the clock resolution reported is consistently less than 5 μs. This allowed us to measure time intervals as small as 5 μs accurately. These two simple tests demonstrated the non-real-time characteristics of the two popular OSs: Windows 2000 and Linux.

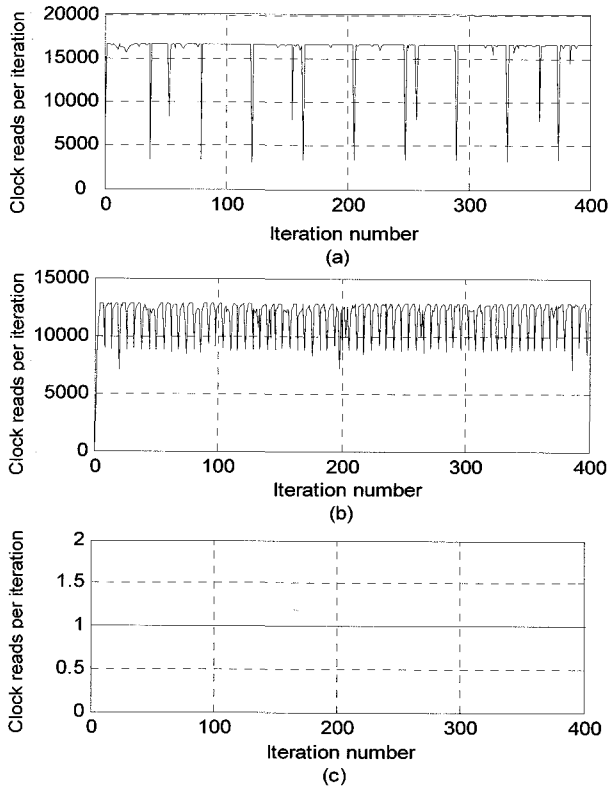


Fig. 5. Plots of the number of clock reads per iteration for the first timing test on (a) Windows 2000, (b) Redhat Linux 7.3, and (c) Redhat Linux 7.3 with RTAI 24.1.12.

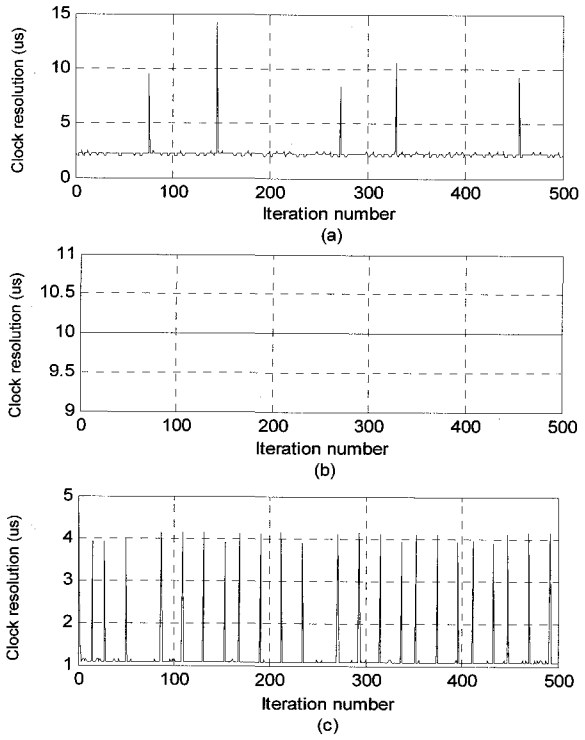


Fig. 6. Plots of the clock resolutions obtained for the second timing test on (a) Windows 2000, (b) Redhat Linux 7.3., and (c) Redhat Linux 7.3 with RTAI 24.1.12.

4.2. Linux control and measurement device interface (comedi).

Comedi [26] is a free software project for tools, libraries, and drivers for various forms of data acquisition, and provides a collection of drivers for a variety of common data-acquisition plug-in boards. It is used to provide the hardware-software interface. It works with the standard Linux kernel as well as the real-time extensions such as RTLinux and RTAI.

5. CASE STUDY

In this section, we provide a case study of network configuration on a ball Maglev system. Based on this test bed, we implement a real-time closed-loop NCS via an Ethernet.

5.1. Ball maglev test bed

Fig. 7 shows a photograph of the ball Maglev test bed [27]. The objective of this test bed is to levitate a steel ball at a predetermined steady-state equilibrium position with an electromagnet. The control input for the set up is the output current from the pulse width modulation (PWM) power amplifier and the system output is the ball position that is measured by an optical position sensor.

The mathematical model between the PWM output (V) and the position sensor output (Y) is described by a second-order transfer function [27].

$$G(s)=\frac{Y(s)}{V(s)}=\frac{-0.02792}{0.0086s^2} \tag{7}$$

A state-space model with sampling period of 3 ms is given as

$$\begin{aligned} x(i_{k+1}) &= \begin{bmatrix} 1 & 0 \\ 0.006 & 1 \end{bmatrix} x(i_k) + \begin{bmatrix} 0.003 \\ 0.000009 \end{bmatrix} u(i_k) + v(i_k), \\ y(i_k) &= \begin{bmatrix} 0 & -1.6233 \end{bmatrix} x(i_k) + w(i_k). \end{aligned} \tag{8}$$

Using this ball Maglev test bed and the Ethernet local area network (LAN) in our lab, we constructed a real-time NCS shown in Fig. 8. The system configuration is the same as shown in Fig. 1. The

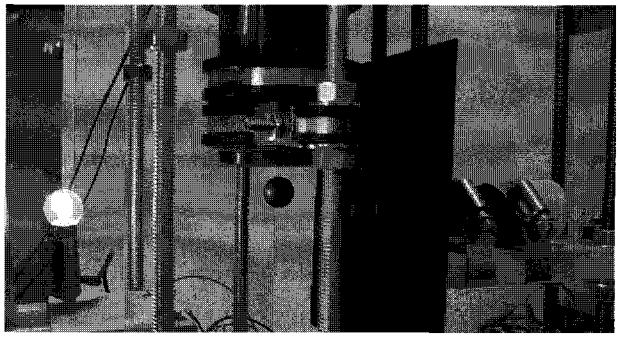


Fig. 7. Ball Maglev system [27].

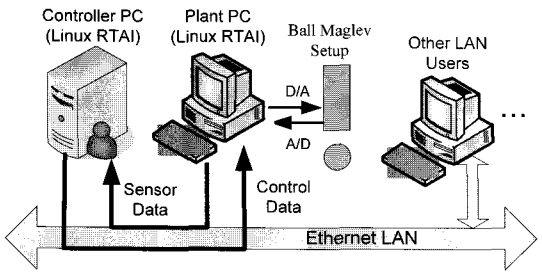


Fig. 8. Real-time NCS over Etherent.

plant PC with NI PCI-6025E as the data-acquisition card enables the ball Maglev test bed to send out sensor data and receive control data through the LAN. The controller PC receives the sensor data, computes the control data, and then sends out the control data through the same LAN. Linux with RTAI is implemented on both PCs to ensure the time-constrained events like sampling and actuating do not miss their deadlines. The average round-trip time delay between the client PC and the server PC was measured and found to be about 230 μs and with a standard deviation of about 200 μs. But there were some sporadic delays as long as 6 ms which is twice longer than the system sampling period.

A digital lead-lag controller designed in [28] to stabilize the ball maglev set up is given as

$$D(z) = 4.15\times10^4\left[\frac{z^2-1.754z+0.769}{z^2-0.782z-0.13}\right]. \quad (9)$$

5.2. Timing of events and predictors for time delays and packet losses

Clock synchronization is a complicated process generating additional network traffic to deliver the synchronized clock signals [29]. A comparatively simpler approach is developed in this paper to coordinate the events in an NCS. Our networked-control architecture is based on a combination of a time-driven sensor and actuator and an event-driven controller. Figs. 9 and 10 show two example timing diagrams of communication processes between the client PC and the server PC with packet losses and time delays. The signals labeled y denote the sensor data transferred from the client to the server, and the signals labeled u denote the control signal data transferred from the server PC to the client PC. The subscripts of these labels denote the sampling-period index (–7, ..., 1) and also indicate whether the data is a prediction (p) or not. For example, y_2 denotes the sensor data of the second sampling period, u_3 denotes the control signal data for the third sampling period, and u_{2p} denotes the control signal predicted for the second sampling period. Eighth-order predictors based on an auto-regressive (AR) model were designed for up to 4-step-ahead predictions of the sensor data after

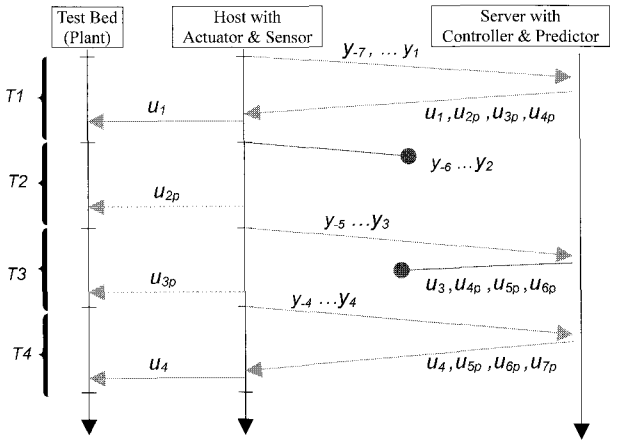


Fig. 9. Example timing diagram of the NCS communication process with packet losses. A line with a round tip denotes that a data packet was lost.

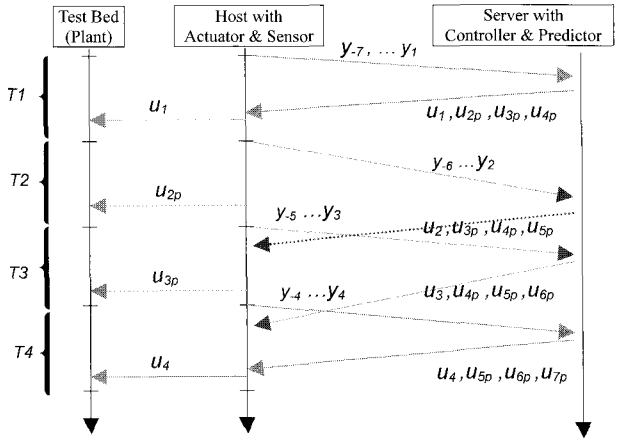


Fig. 10. Example timing diagram of the NCS communication process with time delays. Dotted lines denote the delayed data packet transmission.

numerous design iterations. Then the 4-step-ahead prediction for the control data is calculated using the predicted sensor data using (9).

If the control signal arrives in time, they are used immediately for actuation. If not, the predicted value of the control signal that arrived in the previous packet will be used. The effectiveness of this multi-step- ahead prediction method significantly depends on the accurate system modeling and the parameter vectors for predictors vary with different controlled systems. From Figs. 9 and 10, we can see that the effect of packet loss is no worse than that of time delay and the multi-step-ahead prediction algorithm can deal with packet loss and time delay simultaneously.

5.3. UDP packet composition

The composition of the 68-byte-long Internet Protocol (IP) sensor data-packet transmitted from the

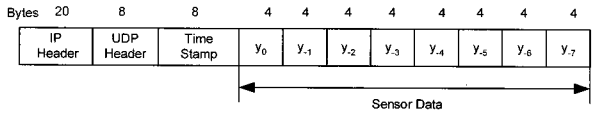


Fig. 11. The frame of sensor data-packet.

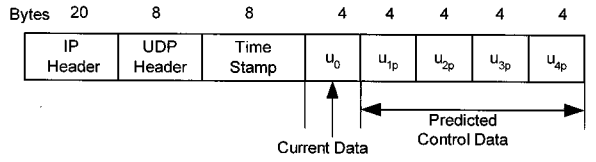


Fig. 12. The frame of control data-packet.

client to the server is shown in Fig. 11. The composition of the 56-byte-long IP control data-packet transmitted from the server to the client is shown in Fig. 12.

A time stamp is taken on the client side at sampling and then transmitted in the sensor data packet to the server side, the server controller adds it in the control data packet transmitted back to the client side. Then the client side uses this time stamp to identify whether the arrived control data packet is the expected packet or an outdated packet. If a packet is outdated, it is simply discarded in the current scheme. Thus the out-of-order data transmission problem is also dealt with in the implementation of our NCS test bed.

5.4. NCS quality of control

With the implementation of the NCS test bed shown in Fig. 8, several experiments were performed to verify the QoC. The first set of experiments was step response. The system step response is shown in Fig. 13.

Then we introduced artificial data-packet losses for the purpose of checking the robust stability and performance of our NCS test bed. The system step response with 20% data-packet losses is shown in Fig. 14. The NCS test bed maintained its stability successfully with degraded performance.

The second set of experiments was to make the ball track commanded trajectories. Fig. 15 shows the system response to a sinusoidal position command.

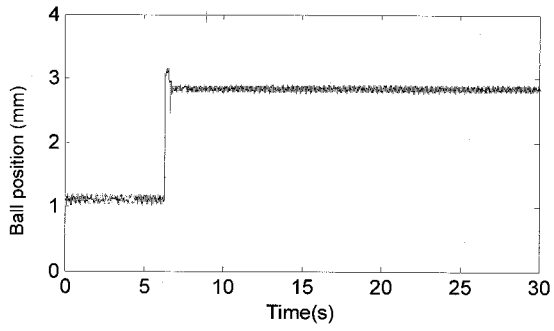


Fig. 13. Step response with the control loop closed over the Ethernet.

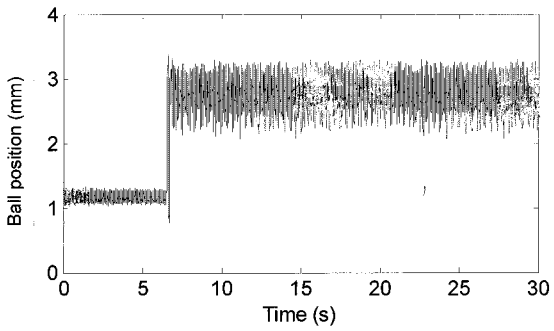


Fig. 14. Step response with 20% data-packet losses.

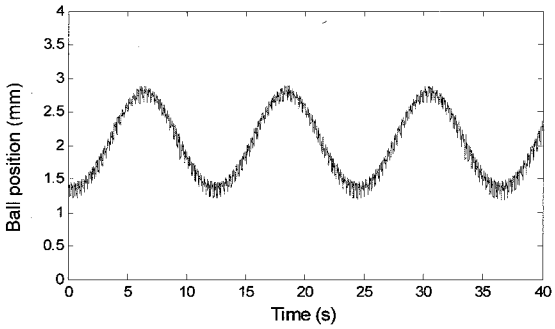


Fig. 15. System response of sinusoidal position command following.

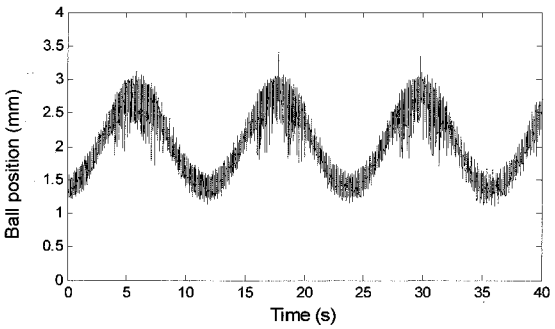


Fig. 16. System response of sinusoidal position command following with 20% packet losses.

Fig. 16 shows the system response to the same sinusoidal command with 20% data-packet losses. Repeating experiments with various command frequencies, we found that the maximum frequency of the command that the system can follow was reduced from 2.8 Hz to 0.35 Hz due to packet losses and imperfect sensor- and control-data prediction.

6. CONCLUSIONS

The main contribution of this paper is to provide a guideline to choose a communication network for NCS design. Network design and real-time OS design were presented as important ingredients of NCS design. The characteristics of the most common network types were discussed to demonstrate how

each of them could be used as a communication medium for NCSs. RTAI with Linux was presented as a good choice as a real-time OS.

We successfully constructed a real-time NCS via Ethernet. With this NCS test bed we experimentally verified the feasibility and effectiveness of our design methodology. A compensation algorithm for network-induced time delays and data-packet losses was also presented and verified. Real-time control via Ethernet is a practical and feasible solution to NCS design.

REFERENCES

[1] Y. Tipsuwan and M.-Y. Chow, "Control methodologies in networked control systems," *Contr. Eng. Practice*, vol. 11, no. 10, pp. 1099-1111, Feb. 2003.

[2] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*, 3rd ed. Reading, Addison-Wesley, MA, 1998.

[3] F.-L. Lian, J. Moyne, and D. Tilbury, "Network design consideration for distributed control systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 2. pp. 297-307, Mar. 2002.

[4] P. G. Otanez, J. R. Moyne, and D. M. Tilbury, "Using deadbands to reduce communication in networked control systems," *Proc. of 2002 American Contr. Conf.*, vol. 4, pp. 3015-3020, May 2002.

[5] J. K. Yook, D. M. Tilbury, and N. R. Soparkar, "Trading computation for bandwidth: Reducing communication in distributed control systems using state estimators," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 4, pp. 503-518, July 2002.

[6] S.-K. Kweon, K. G. Shin, and G. Workman, "Achieving real-time communication over Ethernet with adaptive traffic smoothing," *Proc. IEEE Real-Time Tech. Appl. Symp.*, pp. 90-100, June 1999.

[7] S. H. Hong, "Scheduling algorithm of data sampling times in the integrated communication and control systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 3, no. 2, pp. 225–230, June 1995.

[8] H. S. Park, Y. H. Kim, D. S. Kim, and W. H. Kwon, "A scheduling method for network based control systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 3, pp. 318-330, May 2002.

[9] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," *IEEE Contr. Syst. Mag.*, vol. 21, no. 1, pp. 84-99, Feb. 2001.

[10] G. C. Walsh, Y. Hong, and L. G. Bushnell, "Stability analysis of networked control system," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 3, pp. 438-446, May 2002.

[11] A. Ray, L.-W. Liou, and J. H. Shen, "State estimation using randomly delayed measurements," *ASME J. of Dyn. Syst. Measurement, Contr.*, vol. 115, no. 1, pp. 19-26, Mar. 1993.

[12] K. Ji, W.-J. Kim, and A. Ambike, "Control strategies for distributed real-time control with time delays and packets losses," *Proc. of ASME International Mechanical Eng. Congress and Exposition*, Paper no. 61733, Nov. 2004.

[13] J. Nilsson, *Real-Time Control System with Delays*, Ph.D. dissertation, Lund Inst. Technol., Lund, Sweden, 1998.

[14] K. C. Lee, S. Lee, and M. H. Lee, "Remote fuzzy logic control of networked control system via Profibus-DP," *IEEE Trans. Ind. Electron.*, vol. 50, no. 4, pp. 784-792, Aug. 2003.

[15] F.-L. Lian, J. R. Moyne, and D. M. Tilbury, "Performance evaluation of control networks: Ethernet, ControlNet and DeviceNet," *IEEE Contr. Syst. Mag.*, vol. 21, no. 1, pp. 66-83, Feb. 2001.

[16] B. Englert, L. Rudolph, and A. Shvartsman, "Developing and refining an adaptive token-passing strategy," *Proc. of the 21st International Conference on Distributed Computing Systems*, pp. 597-605, April 2001.

[17] A. Kutlu, H, Ekiz, and E. T. Powner, "Performance analysis of MAC protocols for wireless control area network," *Proc. of Second International Symposium on Parallel Architectures, Algorithms, and Networks*, pp. 494-499, June 1996.

[18] A. S. Tanenbaum, *Computer Networks*, 3rd ed. Prentice-Hall, Upper Saddle River, NJ, 2001.

[19] C. Venkatramani and T. Chiueh, "Supporting real-time traffic on Ethernet," *Proc. of Real-Time Systems Symposium*, pp. 282-286, 1994.

[20] H. Ye, G. Walsh, and L. Bushnell, "Wireless local area networks in the manufacturing industry," *Proc. of American Contr. Conf.*, pp. 2363-2367, June 2000.

[21] H. Ye and G. Walsh, "Real-time mixed-traffic wireless networks," *IEEE Trans. Ind. Electron.*, vol. 48, no. 5, pp. 883-890, Oct. 2001.

[22] N. J. Ploplys, P. A. Kawka, and A. G. Alleyne, "Closed-loop control over wireless network," *IEEE Contr. Syst. Mag.*, vol. 24, no. 3, pp. 58-71, June 2004.

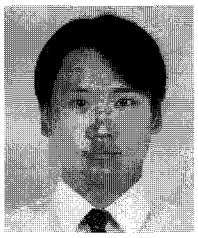
[23] J. D. Decotignie, "A perspective on Ethernet - TCP/IP as a Fieldbus," *Proc. of IFAC Conference on Fieldbus Systems*, pp. 138-142, Nancy, France, 2001.

[24] T. Skeie, S. Johannessen, and C. Brunner, "Ethernet in substation automation," *IEEE Contr. Syst. Mag.*, vol. 22, no. 3, pp. 43-51, June 2002.

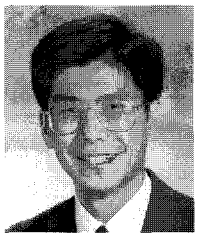
[25] P. Mantegazza, "DIAPM RTAI - real-time application," http://www.rtai.org.

[26] D. Schleef, "Linux control and measurement device interface," http://www.comedi.org.

[27] S. C. Paschall, II, *Design, Fabrication, and Control of a Single Actuator Magnetic Levitation System,* Senior Honors Thesis, Texas A&M University, College Station, TX, May 2002.
[28] A. Ambike, *Closed-Loop Real-Time Control on Distributed Networks*, Masters' Thesis, Texas A&M University, College Station, TX, Aug. 2004.
[29] L. Zhang, Z. Liu, and C. H. Xia, "Clock synchronization algorithms for network measurements," *Proc. of IEEE INFOCOM*, vol. 1, pp. 1160-1169, 2002.



Kun Ji was born in Yangzhou, Jiangsu, P.R.China in 1977. He received the B.S. and M.S. degrees in Mechanical Engineering from Tsinghua University, Beijing, China, in 1999 and 2002, respectively. He is currently a Ph.D. candidate at Texas A&M University, College Station. His research interests focus on analysis, design of networked control systems, real-time control systems, and mechatronic systems. He is a Student Member of IEEE and ASME.



Won-jong Kim received the B.S. (summa cum laude) and M.S. degrees in Control and Instrumentation Engineering from Seoul National University, Seoul, Korea, in 1989 and 1991, respectively, and the Ph.D. degree in Electrical Engineering and Computer Science from Massachusetts Institute of Technology (MIT), Cambridge, in 1997. In September 2000, he joined the Department of Mechanical Engineering, Texas A&M University (TAMU), College Station, where he is currently an Assistant Professor. Following receipt of the Ph.D. degree, he was with SatCon Technology Corporation, Cambridge, MA, for three years. His teaching and research interests focus on analysis, design, and real-time control of mechatronic systems, networked control systems, and nanoscale engineering and technology. He holds three US patents on precision positioning systems. Dr. Kim received the Grand Prize from the Korean Institute of Electrical Engineers' Student Paper Contest in 1988. His 1997 MIT dissertation earned him the Gold Prize from Samsung Electronics' Humantech Thesis Prize. He was a semifinalist of the NIST's Advanced Technology Program 2000 Competition. The NASA granted him the Space Act Award in July 2002. He was appointed a Select Young Faculty Fellow by TAMU College of Engineering and the Texas Engineering Experiment Station twice in September 2003 and October 2005. He is the Chair of the ASME Nanoscale Control Technical Panel and a member of the IEEE Nanotechnology Council. Prof. Kim is a Senior Member of IEEE and a Member of ASME, ASPE, KSEA, Pi Tau Sigma, and Sigma Xi.