1,000-fps Visual Feedback Control of an Active Vision System over a High-Load Network

Daisuke Wako, Shingo Kagami and Koichi Hashimoto
Graduate School of Information Sciences, Tohoku University
6-6-01 Aramaki Aza Aoba, Aoba-ku, Sendai, Japan.
{wako,swk,koichi}@ic.is.tohoku.ac.jp

Abstract

A 1,000-fps visual tracking system over a real-time network based on the standard Ethernet and UDP/IP technologies is presented. The system consists of a high-speed vision system, a host PC that executes motor-control tasks on an RTOS, and an easy-to-construct configuration for real-time communication to deliver the high-speed visual feature information obtained by the vision system to the host PC. Rather than introducing OS-dependent technologies for real-time communication, we have developed a dedicated network processing unit that handles all of the UDP/IP and Ethernet processing instead of the host PC. The visual information packets are forwarded in priority to the other background traffic by off-the-shelf Gigabit Ethernet switches with the IEEE802.1Q/p QoS mechanism. Experimental results show that the end-to-end delay through three Ethernet switches is below 180 μs with 20-μs jitters even under a high network load.

1 Introduction

Visual servoing technologies have been widely applied in various fields such as robot control, industrial automation and medical applications. As demands on control performance increase, it has been recognized that conventional vision sensors are not always fast enough, and much higher frame rates, which vary from several hundreds to thousands frames per second (fps) depending on situations, are required in particular for feedback control of mechanical systems [1, 2].

In many cases, high-speed visual servoing systems are implemented as tightly coupled systems. On the other hand, cooperative visual detection and tracking by distributed cameras have also been extensively investigated during the last decade. Nevertheless, very few studies on high frame rate visual servoing using networked cameras have been reported. Although high frame rate vision systems connected through a network have been used in commercial optical motion capture systems, to the knowledge of the authors, evaluation of real-time communication latency of those systems has not been reported, possibly because it is not so important for motion capture applications, which do not include visual feedback.

Kagami et al. presented a 1,000-fps networked vision system and evaluated its real-time communication latency [3]. The reported system contains an embedded microprocessor with a real-time operating system (RTOS) that controls issuing timing of real-time packets conveying visual feature information. However, it incorporated no mechanism to guarantee real-time communications and the evaluation was done in the environment without significant background traffic.

Meanwhile, it has been strongly demanded to achieve hard real time communications required for robot control and industrial automation over the widely-used Ethernet and TCP/IP network. In response to this demand, the standard for the industrial Ethernet IEC 61784-2 was issued in 2007, and various kinds of communication profile families such as PROFINET and EtherCAT have been standardized and are already commercially available [4, 5].

However, these systems are mainly prepared as replacement for existing industrial fieldbuses, and there seems to exist a high barrier for them to be used, for example, by researchers in the fields of robotics and computer vision. These systems, though standardized, do not offer interoperability between different communication profile families, and typically users have to consider spending a sizable sum of money on equipments so that their network interfaces and switches are unified with a single profile family. In addition, choices for hardware and software of end nodes, i.e. computers for vision and control processing, are restricted to ones that support the profile family.

In this paper, we discuss a more easy-to-construct configuration for real-time communication that does not impose severe restriction on choices for hardware and software of the end nodes, suitable for connecting high-speed sensors and actuators in LAN environments. The presented system consists of a high-speed vision system that enables 1,000-fps image acquisition, a host PC that executes motor-control tasks on an RTOS, and an easy-to-construct real-time communication network to deliver the high-speed visual feature information obtained by the vision system to the host PC. A target tracking system with a 1,000-fps visual feedback rate[1] is constructed on this system and its control performance when the network is cluttered with heavy background traffic is evaluated as well as end-to-end communication delays.

2 Design Strategy

2.1 Isolating Network Processing from the End Node

In order to reduce variation of communication delays in Ethernet LANs, we must reduce variation of both

[1] As it is difficult to specify the acceptable maximum end-to-end delay because it depends on situations, we employ a millisecond, the communication interval, as the immediate goal.

delays that occur in the end nodes and that occur in Ethernet switches. Firstly we describe our strategy to reduce the variation of the delays in the end nodes.

When a multi-task operating system is used in an end node, the way the delays occur strongly depends on the OS since most of the network protocol processing is typically done in the OS kernel. While some of RTOS are equipped with dedicated real-time protocol stacks, others, such as a widely-used free RTOS RTLinux/free, do not allow a real-time task to access the network directly, but instead it has to be accessed via a non-real-time task, which introduces severe unpredictable delays. To avoid these unpredictable delays, several modifications to the kernels have been proposed and successfully applied (e.g. [6]).

In any of the cases, the mechanisms to support real-time communications impose restriction on the choices for OS in the end node and their versions. It sometimes forces users, for example, to give up introducing a new peripheral device, or to keep using an old PC.

To avoid these undesirable situations, we isolate the network protocol processing from the end node, and have them executed in a dedicated external processing unit. The dedicated processing unit contains its own microprocessor and Ethernet interface, and it can periodically send and receive UDP/IP packets independently from the end-node host. Employing UDP/IP as a transport protocol offers flexibility in choices for implementations of communication peers — for example, users can employ the system proposed in this paper as a robot control node and a system running another RTOS with a real-time UDP/IP protocol stack as a vision sensor node.

It would make no sense if the interface between the dedicated network processing unit and the end-node host were complicated and implemented in a way highly depending on the host OS. Thus they are connected to each other through a simple shared memory so that the host can send and receive data only with program read/write operations, that is, without using interrupts.

A similar design concept can be seen in netX by Hilscher GmbH [7]. netX is a network processor that supports many fieldbuses and real-time Ethernet profile families, on which a real-time kernel called rcX runs. Although it is rather overspec for our goal, in which only the standard Ethernet and UDP/IP are used with consumer-use off-the-shelf network switches, it will be possible to implement our architecture on it as an alternative to the iTRON/SH-4 implementation described below.

2.2 Introducing Gigabit Ethernet Switches with QoS Control

Secondly we describe our strategy to reduce the variation of the delays that occur in the network switches. We do not develop a new hardware for this issue, but introduce an off-the-shelf Gigabit Ethernet (GbE) switch with the IEEE 802.1Q/p quality-of-service (QoS) control mechanism, which has been standardized mainly aiming at multimedia communications such as video and voice streaming, and evaluate its effectiveness in motor control applications.

It should be noted that the reason we employ GbE switches is not for the sake of bandwidth, but for ensuring the latency. Even when strict priority queuing, in which a higher-priority packet is always preferentially served, is employed, a highest-priority packet may be kept waiting until the packet just being transmitted when the highest-priority packet arrived is completed. Thus the worst-case delay of the highest-priority packet per hop is the time spent for a maximum-size packet to be transmitted at the wire speed.

This worst-case delay per hop at the 100-Mbps wire speed is as long as approximately 120 μs, which causes nonnegligible variation of the end-to-end delay compared to the packet interval, which is assumed to be a millisecond in this paper, when the packet goes through several hops. By employing GbE switches, the worst-case delay per hop is reduced to 12 μs, and thus the end-to-end delay will be within a millisecond in a LAN with a realistic number of hops. Note that we assume jumbo frames are not used in the LAN. Even if there are many streams of real-time traffic with the highest priority, it will not be a big problem as long as each of the real-time packets are sufficiently small and the number of the real-time streams is not so large.

3 Implementation

Our test implementation consists of a high-speed smart camera node, a PC to control an actuator, and a network connecting them to each other. The camera acquires images at 1,000 fps and send the position of a target in the image at each frame to the PC.

The camera node is the high-speed vision system VCS-IV described in the reference [3]. It is based on a CMOS vision chip [8] containing 64×64 pixels within a 5.4×5.4 mm^2 area. Each pixel contains a digital processing element and pixel-level parallel processing is carried out. The extracted image feature information is handled by a set of real-time tasks in a μiTRON 4.0 compatible RTOS, Mispo NORTi 4, running on a Renesus SH-4 240-MHz processor. Image feature values are sent as series of UDP packets to the control PC. Since real-time tasks in this system can send and receive network packets directly, we do not have to employ the isolating strategy described in Section 2.1 on this side.

The PC for actuator control has an Intel Pentium 4 processor 3.0 GHz and a 1-GB RAM, which runs Vine Linux 3.2 (kernel 2.4.33) with RTLinux-3.2-rc1. Since this free version of RTLinux does not allow real-time tasks to access network resources directly, we employed the isolating strategy described in Section 2.1.

As the dedicated network processing unit isolated from the host PC, an Alpha Project MS104-SH4 board is employed, which is the same product used as the second-level control board for the vision system VCS-IV. A Renesus SH-4 240-MHz processor (SH7750RF240) and an SMSC Ethernet chip LAN91C111 are implemented on the board and the RTOS NORTi 4 is installed in the same way.

The shared memory between the network processing unit and the host PC is implemented with Interface Corp. PCI-4913 and MAT-4914. The host PC and the network processing unit access this shared memory through the PCI bus and the PC/104 bus, respectively.

Generation of receiving and sending tasks, reception and sending of data, and other management operations can be designated from the host. The API functions for these are all implemented by the read and write primitives of the shared memory without any help of

interrupts. When a receiving task or a sending task is generated, a corresponding ring buffer is allocated in the shared memory. The sending task is waken up by a periodic timer interrupt, reads out the data stored in the ring buffer, and sends a UDP packet containing the data. On the other hand, when a UDP packet arrives at the Ethernet port, the receiving task is waken up by the hardware interrupt, and writes the data delivered by the UDP packet into the ring buffer. The host must poll the ring buffer repeatedly in order to check if any received data are available because use of interrupts is avoided on the host side.

The camera node and the PC are connected with a network consisting of Planex Communications SW-0208G GbE switches. SW-0208G is a consumer-use GbE switch with eight 1-Gbps Ethernet ports supporting IEEE 802.1Q VLAN and IEEE 802.1p QoS control. When a packet of the real-time traffic arrives at the first switch, the VLAN tag corresponding to the highest priority is set to the packet, and it is queued strictly in priority by all the switches until it arrives at the destination node.

Note that the links between the end nodes (the camera and the PC) and the edge switches are 100-Mbps ones while the links between the switches are 1-Gbps ones. It does not matter because the links connected to the end nodes are almost exclusively used by the real-time traffic[2].

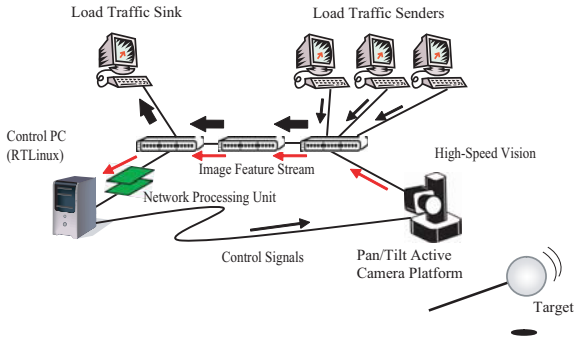

Figure 1: Experimental setup.

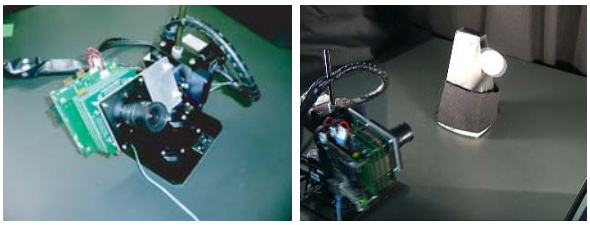

Figure 2: High-speed vision system mounted on the active camera platform.

4 Experiments

4.1 Experimental Setup

Figure 1 shows the setup for the experiments, assuming a scenario of servoing a remote actuator with visual information from a camera. For a simple demonstration, we use a pan/tilt active camera platform as an actuator example, and mount the camera on it. This configuration, in which the actuator is not actually in a remote site, may seem strange, but will be sufficient for proof of principle.

The camera detects the centroid of a moving target in the image and sends its image coordinates to the PC at 1,000 fps. The image coordinates of the target centroid are expressed by the 0th-order moment and the 1st-order moments in x and y directions of the target region in the image, each of which is a 32-bit integer. Thus the data size for a frame is 12 bytes and the Ethernet frame size is fixed to 64 bytes, the minimum size defined in the Ethernet specification. Since the details of image processing are not focused in this paper, we used a white the target and kept the background black.

The packet sent from the camera travels through three GbE switches and arrives at the PC. The PC controls the pan/tilt motors of the active camera platform by a PD control law so that the centroid of the target comes to the center of the image. The pan and tilt axes of the active camera platform [9] are direct-drive actuators driven by Yaskawa 100-W Σ-II AC servo motors. The active camera is shown in Figure 2

Three extra PCs are connected to the first (that is, next to the camera) GbE switch, and another extra PC is connected to the third (next to the control PC) switch. The first three are used as senders of background load traffic, and the other one is the sink of the background traffic. One PC generates approximately 500-Mbps traffic at maximum, and thus the 1-Gbps bandwidth between the switches is almost saturated.

[2]Except for packets incoming to the end nodes including a broadcast packet. Although the effects of these packets are not considered in this paper, they should be filtered out if possible.

4.2 Communication Performances

Figure 3 (a) shows the packet receiving interval at the control PC. The packets of the real-time traffic are sent every 1 millisecond, which should be interpreted as an ideal result. For comparison, the corresponding result for the case without the priority control at the switches is shown in Figure 3 (b) and the result for the case without isolating the network protocol processing from the host PC is shown in Figure 3 (c). In all of these three experiments, the extra PCs generated the background load traffic so that the bandwidths of the links between the switches are saturated, and some background processes were randomly executed in the control PC. In the experiment corresponding to Figure 3 (c), the packets of the real-time traffic are received at a built-in Ethernet port in the PC, and a non-real-time task in the RTLinux obtained the data and recorded the time stamps.

These results show that the proposed configuration is effective in reducing the variation of the communication delays. The reason Figure 3 (b) exhibits rather quantized intervals is because this traffic suffered from many packet drops. No packet drops were observed in the results shown in Figure 3 (a) and (c).

The end-to-end delay between the camera and the PC was also evaluated. For this purpose, the offset and the drift between the clock timers of the sender and the receiver was calibrated through 250-times two-way message exchanges between them. Because this calibration procedure has been implemented only within the network processing unit, the end-to-end delay for

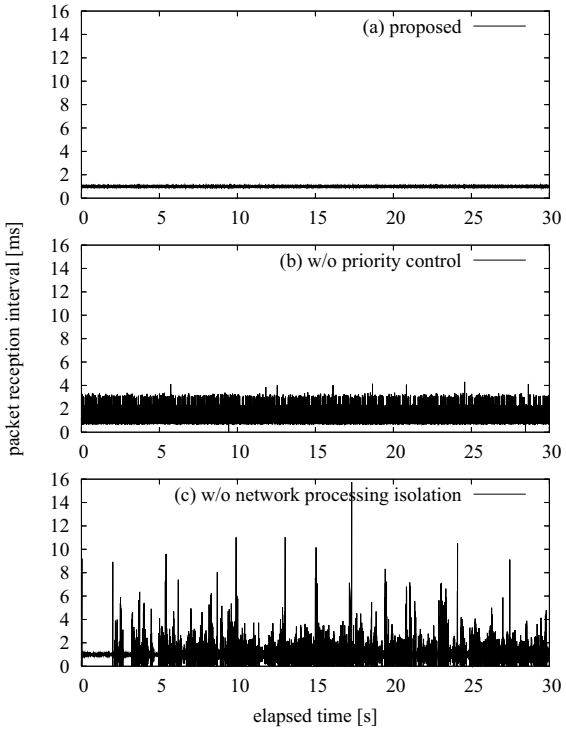

Figure 3: Packet receiving intervals. (a) Proposed system. (b) Without priority control. (c) Without isolation of network processing from the host PC.

the case without isolating the network processing from the host was not measured.

In the case without the priority control, the end-to-end delay varied around from 200 μs to 700 μs. When the priority control at the switches was enabled, on the other hand, the end-to-end delay was within 180 μs with 20-μs jitters.

4.3 Active Camera Control

We also evaluated performance of visual target tracking. A metronome was set at about 300 mm in front of the active camera, as shown in Figure 2, and set to swing at approximately 1.6 Hz where the peak-to-peak amplitude was about 140 mm. The focal length of the lens was 6 mm.

The tracking performance was evaluated for the cases with and without the switch priority control. The root mean squared errors of the target position in the image space, measured from the goal position, that is, the center of the image, were 10.26 pixels and 10.98 pixels respectively for the both cases.

While it is difficult to find significant difference in these tracking performances, clear difference was found in the input voltages to the motor. Figures 4 (a) and (b) show time the input voltages to the pan axis of the active camera platform for the cases with and without the priority control, respectively. The root mean squared absolute input voltages for the both cases were 1.48 V and 1.77 V, respectively.

Even when the priority control is disabled, tracking of the metronome with such a relatively moderate speed was almost successful thanks to the powerful actuators employed in the active camera platform, but it was shown to be at the cost of extra load to the motors.

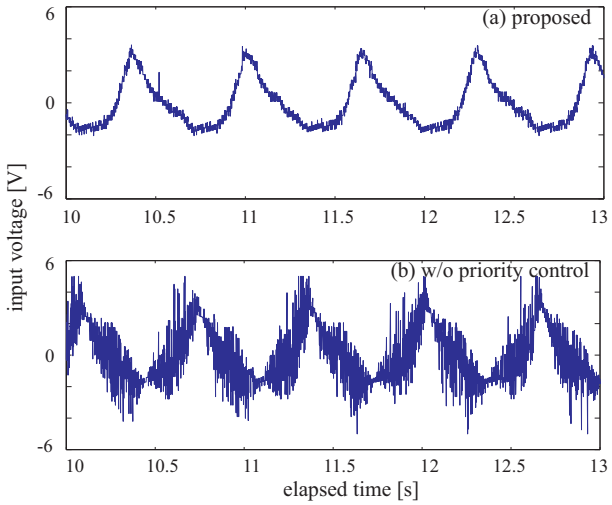

Figure 4: Input voltages to the pan axis. (a) Proposed system. (b) Without priority control.

5 Conclusion

In this paper, a 1,000-fps visual tracking system over a real-time network based on the standard Ethernet and UDP/IP technologies has been demonstrated. By isolating network protocol processing from the host PC and employing consumer-use off-the-shelf Gigabit Ethernet switches with the IEEE 802.1Q/p QoS mechanism, it has been shown that real-time delivery of periodic sensory information with a period of 1 ms is possible with sufficiently low latency even under a heavy network load.

References

[1] Y. Imai, A. Namiki, K. Hashimoto and M. Ishikawa, "Dynamic catching using a high-speed multifingered hand and a high-speed vision system," in *2004 IEEE Intl. Conf. on Robotics and Automation*, pp. 1849–1854, 2004.

[2] R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M. M. A. Sanchez and J. Marescaux, "Beating heart tracking in robotic surgery using 500 Hz visual servoing, model predictive control and an adaptive observer," in *2004 IEEE Intl. Conf. on Robotics and Automation*, pp. 274–279, 2004.

[3] S. Kagami, S. Saito, T. Komuro and M. Ishikawa, "A networked high-speed vision system for 1,000-fps visual feature communication," in *First ACM/IEEE Intl. Conf. on Distributed Smart Cameras*, pp. 95–100, 2007.

[4] "PROFINET," `http://www.profibus.com/pn/`.

[5] "EtherCAT Technology Group," `http://www.ethercat.org/`.

[6] Y. Uchimura and T. Yakoh, "Bilateral robot system on the real-time network structure," *IEEE Trans. Industrial Electronics*, vol. 51, no. 5, pp. 940–946, 2004.

[7] "Hilscher GmbH," `http://www.hilscher.com/`.

[8] T. Komuro, S. Kagami and M. Ishikawa, "A dynamically reconfigurable SIMD processor for a vision chip," *IEEE J. Solid-state Circuits*, vol. 39, no. 1, pp. 265–268, 2004.

[9] Y. Nakabo, M. Ishikawa, H. Toyoda and S. Mizuno, "1ms column parallel vision system and its application of high speed target tracking," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 650–655, 2000.