Distributed Computation and Data Scheduling for Networked Visual Servo Control Systems

Haiyan Wu[1], Lei Lou[1], Chih-Chung Chen[1], Kolja Kühnlenz[12], Sandra Hirche[1]

[1]Institute of Automatic Control Engineering (LSR)

[2]Institute for Advanced Study (IAS)

Technische Universität München

D-80290 München, Germany

Email: {haiyan.wu, chen, koku, hirche}@tum.de, lou@lsr.ei.tum.de

Abstract— The stability and performance of visual servo control systems strongly depend on the delays caused by image processing. In order to accelerate the visual feedback, the distributed computational power across networks and appropriate data transmission mechanism are of particular interest. In this paper, a novel distributed computation with data scheduling is proposed for networked visual servo control systems (NVSCSs) aiming at improving the control performance. A realtime transport protocol is developed for image data transmission. For a NVSCS which is modeled as a continuous-time system with computation, transmission and holding delays, a switching control law is applied. A probabilistic sampling scheduler is derived such that the control performance and the network load caused by image data transmission are balanced. Experiments on two 1-DoF linear modules equipped with a camera are conducted to validate the proposed approach. A visual servo system without data scheduling is implemented for comparison. The experimental results demonstrate a comparable control performance of the proposed approach with an advantage of reduced network load.

I. INTRODUCTION

By visual servo control, vision sensors are utilized in the closed-loop to provide useful visual information for interpreting general three-dimensional relationships in a scene [1]. However, the control performance of visual servo control systems might be limited by the long feedback latency caused by the image processing [2]. For a position-based visual servoing (PBVS) system [3] considered in this paper, long image processing delay primarily caused by feature extraction and pose estimation algorithms exists in the feedback loop. The image processing delay is random due to the varying number of the extracted image features. Random delay in the control loop poses challenges on the stability analysis and controller design for visual servo control systems.

With the recent developments in communication and computing technologies, parallel computation based on networked computational resources has gained tremendous attention [4], [5]. Particularly for computer vision, the networked distributed computation platform has become an attractive alternative to traditional supercomputers [6], [7]. Thus, in this paper the computational power across the network is considered for visual servo control. By using the existing computational resources, video grabbing, image processing algorithms, the controller and actuators can be implemented on different platforms across a communicational
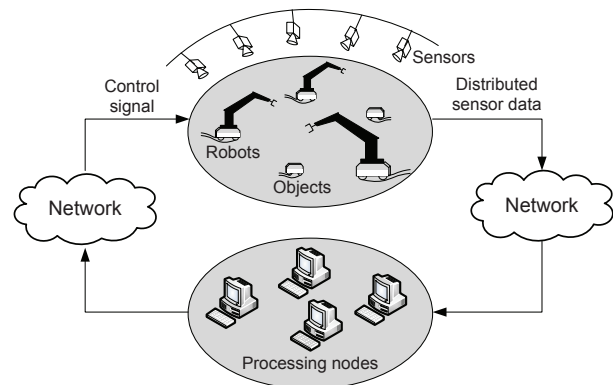

Fig. 1. Scheme of networked visual servo control systems with distributed sensors and distributed computations.

network, see Fig. 1 for a visualization. This kind of setup results in *networked visual servo control systems (NVSCSs)*. However, exchanging the visual data over a communication network suffers an additional non-ideal signal transmission, especially if transmission technology is used. In this paper, a novel framework of NVSCSs with distributed computation is proposed to increase the control performance. A realtime transport protocol is developed for image data transmission. In order to reduce the network load caused by image data transmission, a *data scheduling* considering the network capacity and the control performance constraints is designed.

The remainder of this paper is organized as follows: In Section II, the image processing algorithm is introduced, and a realtime image data transport protocol is developed. A switching control law considering feedback delay is applied in NVSCSs to improve the control performance in Section III. In Section IV, a cost function is defined concerning control performance and network data flows, and a data scheduling is developed to reduce the network load. In Section IV, the experimental validation and performance comparison are presented and discussed.

II. IMAGE PROCESSING AND TRANSMISSION IN NVSCSS

In this paper, a tracking problem with networked visual servo control is studied, as shown in Fig. 2. A camera-in-hand structure is selected for tracking a moving target with PBVS.
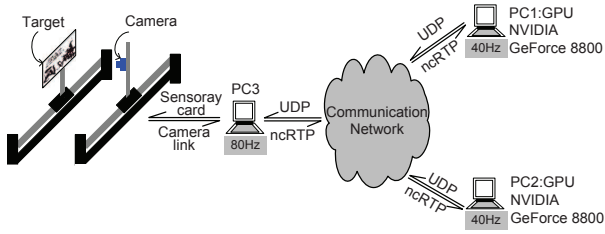
Fig. 2. Experimental setup of a NVSCS comprising of a pair of 1-DoF linear motor modules. PC_1, PC_2: image processing with GPU implementation (40Hz). PC_3: image capturing (80Hz) and controller;

A. Pose Estimation

An essential problem of PBVS is the 3D pose estimation by using image-based observations. To achieve this, image features are firstly extracted from the image sequences. The extracted features are then matched with the ones in the reference image captured at a desired pose. By given the matched feature pairs, the pose of the target with respect to the camera is estimated [8].

In order to increase the accuracy of pose estimation, SIFT feature [9], which is known for its robustness, is applied for feature extraction. Moreover, a GPGPU (General-Purpose computing on Graphics Processing Units) implementation of SIFT feature extraction and feature matching is applied. Exploiting the massive parallel processing capability of GPU, the computational time is reduced. In addition, RANSAC (RANdom Sample Consensus) [10] algorithm is selected for outliers rejection after feature matching.

It has to be mentioned, that image features vary from frame to frame due to different view angles, illumination conditions and noise. This causes a random delay for pose estimation which depends on the number of image features. Besides, although a speedup of the image processing is achieved with the GPU implementation [11], the computation delay is much longer ($\approx$ 20ms for an image of 640×480 pixels with about 100 features detected, on the graphic card NVIDIA GeForce 8800) than the control period ($\approx$ 1ms) of a joint servo system. To achieve better control performance, a distributed computation platform is thus considered to accelerate the visual feedback.

B. Distributed Computation

To establish a flexible and economical parallel image processing platform, the computational sources connected over a common network are utilized. A single channel image with a resolution of 640×480 pixels has a size of 300 KB. With a sampling rate larger than 60 Hz, the network load is more than 144 Mbps. In this paper, the 1000BASE-T Ethernet is used to build the physical network. Other data link/physical layer devices such as EtherCAT, CAN, SERCOS or WLAN either need dedicated hardwares or have limited transmission rates. For example, EtherCAT and RTNET are based on Ethernet and have deterministic delays. However, the overhead for synchronization (EtherCAT: token ring, RTNET: TDMA) will introduce high bandwidth penalty, which means the average transmission delay is much larger. By WLAN,

the theoretical speed of 802.11n is 300 Mbps. However, the maximum real-world speed is only about 100 Mbps.

Although there are various transport protocols in the public domain, they are not suitable for image transmission, e.g. UDP can not transmit data larger than 64 KB and TCP has a large variance of the transmission delay due to retransmissions. Therefore, a realtime image transmission protocol based on Realtime Transport Protocol (RTP) is developed.

RTP for Networked Control

RTP [12] is an application layer protocol for transmitting latency-sensitive data, such as video and audio on the Internet. It is the foundation of many Voice over IP and media streaming systems. RTP uses a separate channel for monitoring and adjusting the data delivery. In this paper, ncRTP (Networked Control RTP) is designed and implemented based on GNU ccRTP [13], which supports UDP as transport layer protocol. The algorithms of ncRTP are described in the following.

1) Fragmentation Mechanism: To support diverse data formats of cameras for NVSCS applications, a transparent data fragmentation/reassembly layer is built in ncRTP. Large frames are truncated as a series of data blocks patched with an extended identification number. For example, an image with a size of 300 KB is segmented into five data blocks with each block smaller than 64 KB. The receiving stack can detect and reassemble the frame according to the identification number. If there is any packet losses, the whole image frame will be dropped, instead of retransmitting it.

2) Synchronization and Timing: The distributed computation system is designed as an event-driven system. Data processing threads are triggered when new frames have been assembled in receiving buffer. Therefore the system works with the same frequency of the grabbing rate of cameras. The Real Time Clock (RTC) of hosts in NVSCS can be synchronized by Network Time Protocol (NTP) up to 0.1 ms. The RTP header is extended according to [12] and the RTC time stamp is appended in the extension. The streaming server is designed on RTAI realtime kernel and the sending process is scheduled periodically within a jitter of 10 μs.

With the ncRTP introduced in this section, a parallel image processing platform is established for NVSCSs. However, transmitting image data over the network will result in a large network load. To guaranteed a desired control performance and to avoid the network congestion, a data scheduling strategy is considered. In the rest of this paper, a switching controller as proposed in [14] is applied to improve the control performance. The properties of the feedback delays are analyzed. Then, a cost function is defined to balance the guaranteed control performance versus the network data flow. Moreover, the data scheduling algorithm for image streaming developed in this paper is introduced.

III. SWITCHING CONTROL FOR NVSCSS

Consider a linearized NVSCS

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad (1)$$

where $x(t) \in \mathbb{R}^n$ is the state and $u(t) \in \mathbb{R}^m$ is the control input; A and B are constant matrices with appropriate dimensions.
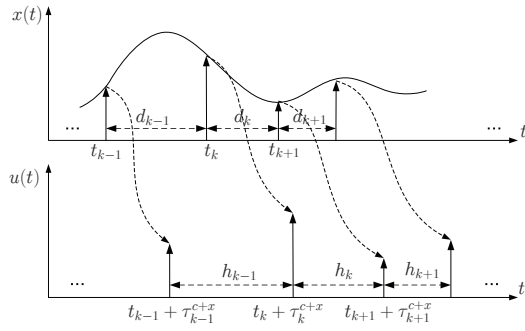
Fig. 3. Timing diagram of a NVSCS with random computation delay τ_k^c, transmission delay τ_k^x and sampling interval d_k.

Under the control law

$$u(t) = Kx(t_k), \quad t \in [t_k, t_{k+1}], \quad \forall K \in \mathbb{N},$$

the closed-loop system is derived as

$$\dot{x}(t) = Ax(t) + BKx(t_k), \quad t \in [t_k, t_{k+1}). \quad (2)$$

For NVSCSs, the packet $x(t_k)$ arrives at the controller with delay τ_k^{c+x} consisting of random computation delay $\tau_k^c > 0$ and transmission delay $\tau_k^x > 0$, see Fig. 3. d_k is the sampling interval of the sensor for k-th sampling. The computation delay, the transmission delay and sampling interval are assumed to be i.i.d. processes. Hence, the closed-loop system in (2) becomes

$$\dot{x}(t) = Ax(t) + BKx(t_k), \quad t \in [t_k + \tau_k^{c+x}, t_{k+1} + \tau_{k+1}^{c+x}), \quad (3)$$

with update interval

$$h_k = d_k + \tau_{k+1}^{c+x} - \tau_k^{c+x}. \quad (4)$$

For further analysis we assume that the computation delays, transmission delays and sampling intervals are modeled by i.i.d process and take values in a finite set

$$\begin{aligned}
\tau_k^c &\in T^c = \{T^{c1}, T^{c2}, \ldots, T^{cp}\}, \quad p \in \mathbb{N}, \\
\tau_k^x &\in T^x = \{T^{x1}, T^{x2}, \ldots, T^{xq}\}, \quad q \in \mathbb{N}, \\
d_k &\in T^d = \{T^{d1}, T^{d2}, \ldots, T^{dl}\}, \quad l \in \mathbb{N}.
\end{aligned} \quad (5)$$

Moreover, throughout this paper we assume that packets do not overtake each other, i.e. packets arrive at the controller according to their sending order.

Reformulate (3) into continuous-time system with time-varying delay by means of input-delay approach [15]

$$\begin{aligned}
\dot{x}(t) &= Ax(t) + BKx(t - \tau(t)), \\
t &\in [t_k + \tau_k^{c+x}, \quad t_{k+1} + \tau_{k+1}^{c+x}), \\
x_0 &= x(\theta), \quad \theta \in [-\bar{\tau}, 0],
\end{aligned} \quad (6)$$

where $\tau(t)$ is time varying delay and assumed be bounded

$$\begin{aligned}
\tau(t) &= \tau_k^c + \tau_k^x + h(t), \quad t \in [t_k + \tau_k^{c+x}, t_{k+1} + \tau_{k+1}^{c+x}], \\
\underline{\tau} &= \min_{k \in \mathbb{N}}\{\tau_k^{c+x}\}, \quad \bar{\tau} = \max_{k \in \mathbb{N}}\{h_k + \tau_k^{c+x}\},
\end{aligned} \quad (7)$$

with $h(t) \in [0, h_k]$ denotes the holding delay, see also [14]. In order to improve the control performance, a switching control mechanism proposed in [14] is applied resulting in

$$\dot{x}(t) = Ax(t) + \sum_{i=1}^{n} \beta_i BK_i x(t - \tau(t)), \quad (8)$$

where β_i is the indicator function

$$\beta_i = \begin{cases} 1, & s_{i-1} \leq \tau(t) < s_i, \quad i = 1, \ldots, n, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

Categorizing $\tau(t)$ into n finite intervals $\tau_i = \{\tau | s_{i-1} \leq \tau < s_i\}$, $i = 1, \ldots, n$ and define

$$\begin{aligned}
\tau_k^a &= \tau_k^c + \tau_k^x + h_k = \tau_{k+1}^c + \tau_{k+1}^x + d_k, \\
\tau_k^a &\in T^a = \{T^c + T^x + T^d\}.
\end{aligned}$$

Select s_i as any subset S, $s_i \in S \subset T^a$. Due to the i.i.d. assumption on τ_k^c, τ_k^x and d_k, s_i is also i.i.d [16]. As a result, the occurrence probability p_i of each delay interval satisfies

$$\mathbf{Pr}\{\beta_i = 1\} = p_i, \quad \sum_{i=1}^{n} p_i = 1. \quad (10)$$

Remark 1: In order to derive the occurrence probability p_i from the occurrence probabilities of the delay components, we categorize τ^c, τ^x and d_k into $U \geq 1$, $V \geq 1$ and $W \geq 1$ with s_u^c, s_v^x and s_w^d satisfying

$$\begin{aligned}
s_{u-1}^c &< s_u^c, \quad s_u^c > 0, \quad u = 1, \ldots, U-1, \\
s_{v-1}^x &< s_v^x, \quad s_v^x > 0, \quad v = 1, \ldots, V-1, \\
s_{w-1}^d &< s_w^d, \quad s_w^d > 0, \quad w = 1, \ldots, W-1,
\end{aligned}$$

s_u^c, s_v^x, s_w^d take values in the sets T^c, T^x, T^d respectively. The occurrence probabilities of the delay intervals are

$$\mathbf{Pr}\{s_{u-1}^c \leq \tau_k^c < s_u^c\} = p_u^c, \quad \sum_{u=1}^{U} p_u^c = 1,$$

$$\mathbf{Pr}\{s_{v-1}^x \leq \tau_k^x < s_v^x\} = p_v^x, \quad \sum_{v=1}^{V} p_v^x = 1, \quad (11)$$

$$\mathbf{Pr}\{s_{w-1}^d \leq d_k < s_w^d\} = p_w^d, \quad \sum_{w=1}^{W} p_w^d = 1.$$

The delay intervals and associated occurrence probabilities of indicator functions in (9) (10) become

$$\begin{aligned}
s_i &= \sum_{u=1}^{U} \sum_{v=1}^{V} \sum_{w=1}^{W} s_u^c + s_v^x + s_w^d, \\
p_i &= \sum_{u=1}^{U} \sum_{v=1}^{V} \sum_{w=1}^{W} p_u^c p_v^x p_w^d.
\end{aligned} \quad (12)$$

The switching controller discussed above aims at improving the control performance of NVSCSs. The stability analysis and control design algorithm are given in [14]. In the next section, the delay intervals s_i and p_i are further analyzed to derive a data scheduling mechanism for NVSCSs with guaranteed control performance.

IV. GUARANTEED PERFORMANCE WITH DATA SCHEDULING

A. Performance Cost and Network Usage

In this paper, the control performance is considered together with the network load due to the image data transmission. A cost function towards the trade-off between control performance and network usage is formulated as

$$J = \lim_{T \to \infty} \mathbb{E}\left\{ \int_0^T x^T(t) Rx(t)dt + \frac{1}{T} \sum_{w=1}^{W} \int_0^T C_w(t)dt \right\}, \quad (13)$$

where $R > 0\,(R \in \mathbb{R}^{n \times n})$, $C_w(t) > 0\,(C_w(t) \in \mathbb{R})$. The first term in (13) concerns the control performance cost, whereas the second term represents the network usage over the runtime T. Within the context of this paper, data scheduling means to determine appropriate probability distributions of sampling intervals p_w^d in (11), such that the cost function (13) is minimized and the stability constrain in [14] is preserved.

The first term in (13) is bounded by a positive scalar $\bar{L}_{per}(p_1, \ldots, p_n)$

$$\lim_{T \to \infty} \mathbb{E}\left\{ \int_0^T x^T(t) R x(t) dt \right\} \leq \bar{L}_{per}(p_1, \ldots, p_n) \qquad (14)$$

Proof: See Appendix VIII. ∎

Note that the guaranteed cost in (14) is a function of $p_1, \ldots, p_n$. Assume the probability distributions of transmission and computation delays, $p_1^c, \ldots, p_U^c$ and $p_1^x, \ldots, p_V^x$ defined in (11), are constant and known. According to (12), the guaranteed cost (14) is determined by the probability distributions of sampling intervals. The formula in (14) can be rewritten as

$$\lim_{T \to \infty} \mathbb{E}\left\{ \int_0^T x^T(t) R x(t) dt \right\} \leq \bar{L}_{per}(p_1^d, \ldots, p_W^d) \qquad (15)$$

The second term in (13) can be estimated as

$$\lim_{T \to \infty} \mathbb{E}\left\{ \frac{1}{T} \sum_{w=1}^W \int_0^T C_w(t) dt \right\} = \sum_{w=1}^W p_w^d C_w, \qquad (16)$$

where C_w denotes the network usage of the associated interval $s_{w-1}^d \leq d_k < s_w^d$ defined by

$$C_w(t) = \begin{cases} C_w, & s_{w-1}^d \leq d_k < s_w^d, \quad w = 1, \ldots W, \\ 0, & \text{otherwise.} \end{cases}$$

Substitute (15) and (16) into (13), the stochastic cost function becomes a deterministic function

$$J \leq \bar{L}_{per}(p_1^d, \ldots, p_W^d) + \sum_{w=1}^W p_w^d C_w. \qquad (17)$$

Therefore, an optimal network utilization results in minimizing (17) by the choice of an appropriate set p_w^d, $w = 1, \ldots, W$. Observe that any change in p_w^d results in variations of p_i, see (12). And the stability condition in [14] depends on p_i and needs to be considered during the optimization. The details of the data scheduling problem are formulated as follows.

Proposition 1: An optimal probabilistic sampling is given by

$$\min_{p_1^d, \ldots, p_W^d} J(p_1^d, \ldots, p_W^d),$$
$$\text{s.t.(LMI in Theorem 1 in [14]))} \qquad (18)$$

where $p_1^d, \ldots, p_W^d$ satisfying $\sum_{w=1}^W p_w^d = 1$ is the set of admissible probability distribution of sampling rates.

Hence, the probability distributions of different sampling rates are determined by minimizing (18). With determined $p_1^d, \ldots, p_W^d$, a data scheduler at the sender is implemented.

B. Data Scheduling Algorithm

As mentioned in Section II-B, each processing node in NVSCSs works at an event-driven architecture and accomplishes the image processing with the delay τ_k^c. Define $d = \max(\tau_k^c)$, the maximum sampling interval of the camera is $\frac{d}{m}$,
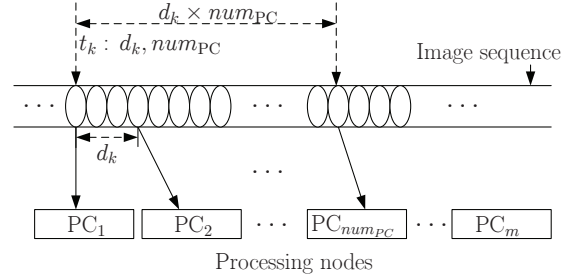

Fig. 4. Data scheduler for image streaming in NVSCS. d_k and num_{PC} is selected at time instant t_k. d_k is used as the sub-sampling interval for the time interval $[t_k, t_k + d_k \times num_{\text{PC}})$. $PC_1, \ldots, PC_m$ are available processing nodes in a NVSCS and works at event-driven architecture.

if m processing nodes are available over the network. Thus, the finite set of d_k values defined in (5) becomes

$$d_k \in T^d = \left\{ \frac{d}{m}, \frac{2d}{m}, \ldots, d \right\}, \quad m \in \mathbb{N}.$$

With selected d_k, the associated number of processing nodes is

$$num_{\text{PC}} = \frac{d}{d_k} \in \left\{ m, \frac{m}{2}, \ldots, 1 \right\}, \quad num_{\text{PC}} \in \mathbb{N}.$$

According to Remark 1, $s_w^d, \ldots, s_W^d$ take also values from $T^d \in \{ \frac{d}{m}, \ldots, d \}$. With $p_1^d, \ldots, p_W^d$ determined by (18), the algorithm of data scheduler realized at the sender is described in Algorithm 1.

Algorithm 1 Image Scheduler at ncRTP sender

Require: $s_w^d = T^d \in \{ \frac{d}{m}, \ldots, d \}$, p_w^d, $w = 1, \ldots, W$
Task: determine the receiver for Img_{t_k} (image
    sampled at time instant t_k)
  while $0 < t \leq T$ **do**
    determine d_k and num_{PC} by the random selection of s_w^d,
    with pre-defined p_w^d, $w = 1, \ldots, W$.
    for $num = 1 : 1 : num_{\text{PC}}$ **do**
      $t_k = t$;
      send Img_{t_k} to PC_{num}, ($PC_{num} \in \{PC_1, \ldots, PC_m\}$ rep-
      resents the available processing node at t_k)
      $t = t_k + d_k$;
    end for
  end while

The objective of Algorithm 1 is to determine the processing node for the image Img_{t_k} captured at t_k. Firstly, at the instant t_k, the sampling interval d_k and the number of processing nodes num_{PC} is determined by randomly selecting $d_k = s_w^d \in \{ \frac{d}{m}, \ldots, d \}$. The probability distribution of s_w^d, p_w^d, is pre-defined by optimizing the problem in Proposition 1. Then, the images captured at the instants $t_k, \ldots, t_k + d_k num_{\text{PC}}$ are sent to $PC_1, \ldots, PC_{num_{\text{PC}}}$ respectively, see also fig. 4.

With the algorithm introduced above, a data scheduling is realized for NVSCSs with guaranteed control performance. Instead of transmitting every captured image, only selective images (sub-sampling of the original image sequence) are sent over the network. Thus, the network load is reduced.

Remark 2: In the current stage, a data scheduling with desired probability distribution is developed with random selection of the s_w^d. In the future, a more appropriate policy for image streaming rather than random selection will

be considered, e.g. switching to higher sending rate with increased control error and vice versa.

V. EXPERIMENT

In this section, the proposed NVSCS with data scheduling is validated in experiments. As shown in Fig. 2, the considered NVSCS consists of two commercial linear motors from Copley Control Corp, a camera and three PCs connected over network. A target is mounted on a reference linear motor module assigned with a trajectory signal which has a amplitude of $20\,\text{cm}$ and a frequency of $\pi/3\,\text{rad/s}$. The two linear motor modules are connected to the host PC_3 running RTAI Linux via a Sensoray S626 I/O card. To enable a high-speed vision feedback in the control loop, the images captured by the camera with a resolution of 640×480 pixels are processed by distributed processing nodes PC_1, PC_2 (X86-64 AMD Phenom II $\times 4$ 810 processor) over a communication network. The image data is transmitted over the network based on the ncRTP introduced in Section II.

In the experiments, the camera works at a framerate of $80\,\text{Hz}$. As mentioned before, each image frame with a resolution of 640×480 pixels has a data size of 300KB. A sampling rate of 80 Hz means 192 Mb/s data flow over the network. In order to reduce the network load, Algorithm 1 for data scheduling introduced in the previous section is applied. Since only two processing nodes are available in the experiments, the data scheduling algorithm is simplified as following:

- PC_1: Images with frame number $\{1,3,...\}$ are sent to PC_1 periodically, which leads to a sampling rate 40 Hz.
- PC_2: The subset of images with frame number $\{2,4,6,...\}$ are sent to PC_2 randomly, which leads to a sampling rate $\leq 40\,\text{Hz}$. In the experiment, the images sent to PC_2 are randomly selected with desired probability distribution.
- PC_3: For the controller, it works at bounded sampling rate $[40\,\text{Hz}, 80\,\text{Hz}]$.

The system parameters are obtained through standard least square identification of the response to square pulse input

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.959 & -1169.9 \end{bmatrix}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}u(t). \quad (19)$$

For simplicity, consider a single interval for the combination of the transmission and the computation delays $\tau_k^c + \tau_k^x$ with the occurrence probability

$$P\{\tau_k^c + \tau_k^x \leq s_1^c + s_1^x\} = p_1^c p_1^x = 100\%,$$
$$s_1^c + s_1^x = \bar{\tau}^{c+x} = 35 \text{ ms},$$

where $\bar{\tau}^{c+x}$ is determined in the experiments. Select $s_1^d = \frac{1}{80}$ ms, $s_2^d = \frac{1}{40}$ ms, and according to (12) in Remark 1, the delay intervals and the probability distributions become

$$s_1 = s_1^c + s_1^x + s_1^d = 47.5\,\text{ms}, \quad p_1 = p_1^c p_1^x p_1^d = p_1^d,$$
$$s_2 = s_1^c + s_1^x + s_2^d = 60\,\text{ms}, \quad p_2 = p_1^c p_1^x p_2^d = p_2^d.$$

The probability distributions, p_1^d and p_2^d, are designed such that the optimal network utilization is achievable. The control module is equipped with a set of delay-dependent PD
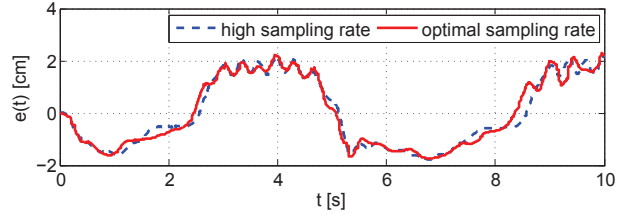

Fig. 5. The mean control error evolution of the proposed approach with data scheduling $[p_1^d \; p_2^d] = [50\% \; 50\%]$ (solid line) and the standard approach without data scheduling $[p_1^d \; p_2^d] = [100\% \; 0\%]$ (dash line).

controllers. Combine the switching PD controller with (19)

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.959 & -1169.9 \end{bmatrix}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}$$
$$+ \sum_{i=1}^{2} \beta_i K_i \begin{bmatrix} x(t-s_i) \\ \dot{x}(t-s_i) \end{bmatrix}, \quad (20)$$

where $K_i = \begin{bmatrix} 0 & 0 \\ -K_{Pi} & -K_{Di} \end{bmatrix}$.

Set the parameters $C_1 = 2$, $C_2 = 1$ and $R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. The optimization problem in Proposition 1 is numerically solved by the optimization `fmincon` as well as `Yalmip toolbox` in Matlab. With the initial condition $[x(0) \; \dot{x}(0)]^T = [0 \; 0]$, $\theta \in [-s_2, 0]$, the cost function in (13) is optimized $[p_1^d \; p_2^d] = [50\% \; 50\%]$ for $J = 1.67$. The associated stabilizing state-feedback gains are

$$K_1 = \begin{bmatrix} 0 & 0 \\ -900 & -15 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 0 & 0 \\ -600 & -5 \end{bmatrix}.$$

For comparing the control performance, a standard networked visual servo control system without data scheduling (high network load), e.g. $[p_1^d \; p_2^d] = [100\% \; 0\%]$, is implemented

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.959 & -1169.9 \end{bmatrix}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}$$
$$+ \begin{bmatrix} 0 & 0 \\ -900 & -15 \end{bmatrix}\begin{bmatrix} x(t-s_2) \\ \dot{x}(t-s_2) \end{bmatrix}.$$

The control error is defined by $e(t) = x_r(t) - x_c(t)$, $x_r(t)$ and $x_c(t)$ denote the position measurements of the reference module and the controlled module respectively. A comparable control performance is achieved, as shown in Fig. 5. The proposed approach with data scheduling has a maximum tracking error $|e|_{\max} = 2.29\,\text{cm}$ and a mean tracking error $|e|_{\text{mean}} = 1.22\,\text{cm}$, similar to the maximal tracking error of high sampling rate design approach (+4.57%), see Table I. However, the network usage (data flow) is 25% less than the one with high sampling rate design approach. It has to be mentioned, that the resulted tracking error is relative large under stability constraint. Besides, the PD controller without considering the signal dynamics leads also to large tracking error. And the control performance is further deteriorated by non-ideal friction compensation and the lack of delay compensation strategy.

The experimental results demonstrate a comparable control performance of the proposed approach with reduced network load compared with the standard approach.

sampling rate	$|e|_{max}$ [cm]	$|e|_{mean}$ [cm]	Network usage [unit]
optimal	2.29	1.22	1.5
high	2.19	1.19	2

VI. CONCLUSIONS

This paper presents a design approach for networked visual servo control systems with distributed computation and data scheduling. A realtime transport protocol ncRTP is developed for image data transmission. High-sample-rate visual feedback is realized with distributed computation. To achieve better control performance, a switching control mechanism is applied. Furthermore, a cost function is proposed to balance the control performance and the network load caused by image data transmission. A data scheduling algorithm is designed for image streaming in NVSCSs. The proposed approach is validated by experiments on two linear motor modules. The results demonstrate comparable control performance of the proposed approach with less network cost compared with the conventional counterpart. In the current stage, the approach is limited to visual servo control, which can be approximated as linear systems. The future work is concerned with extending the approach to more general nonlinear systems. Developing an appropriate policy for sampling rate adaptation is also part of the future work.

VII. ACKNOWLEDGMENTS

VIII. APPENDIX

Consider the Lyapunov-Krasovskii candidate in [14]

$$V(z(t)) = V_0(z(t)) + \sum_{i=1}^{n} V_i(z(t)),$$

where

$$V_0(z(t)) = z^T(t)EPz(t),$$

$$V_i(z(t)) = \int_{-s_i}^{0} \int_{t+\theta}^{t} z^T(s)\bar{A}_i^T Q_i \bar{A}_i z(s)\, ds\, d\theta.$$

Due to the fact $x(t) = [I\ 0]z(t)$, consider Dynkin's formula and eq. (17) in [14]

$$\mathscr{L}V(z(t)) \leq z^T(t)[\bar{A}^T P + P^T \bar{A} + \sum_{i=1}^{n} s_i \bar{A}_i^T Q_i \bar{A}_i$$

$$+ \sum_{i=1}^{n} s_i P^T Q_i^{-1} P]z(t) = z^T(t)\Theta z(t),$$

where

$$\bar{A} = \begin{bmatrix} 0 & I \\ A + \sum_{i=1}^{n} \beta BK_i & -I \end{bmatrix}, \quad \bar{A}_i = \begin{bmatrix} 0 & 0 \\ 0 & \beta_i BK_i \end{bmatrix},$$

$$z^T(t) = [x^T(t)\ \dot{x}^T(t)], \quad E = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \quad P = \begin{bmatrix} P_1 & 0 \\ P_2 & P_3 \end{bmatrix},$$

and $Q_i > 0$ is symmetric matrices, $P_1 > 0$, P_2 and P_3 are real matrices. Define the performance cost function as

$$L_{per} = \mathbb{E}\left\{ \int_0^T x^T(t)Rx(t)dt \right\},$$

it becomes

$$L_{per} = \mathbb{E}\left\{ \int_0^T z^T(t)\begin{bmatrix} I \\ 0 \end{bmatrix} R\begin{bmatrix} I & 0 \end{bmatrix} z(t)dt \right\}$$

$$= \mathbb{E}\left\{ \int_0^T [z^T(t)\begin{bmatrix} I \\ 0 \end{bmatrix} R\begin{bmatrix} I & 0 \end{bmatrix} z(t) + \mathscr{L}V(z(t))]dt \right\}$$

$$- \mathbb{E}\left\{ \int_0^T \mathscr{L}V(z(t))dt \right\}$$

$$\leq \mathbb{E}\left\{ \int_0^T z^T(t)\bar{\Theta}z(t)dt + V(z(0)) \right\},$$

where $\bar{\Theta} = \Theta + \begin{bmatrix} I & 0 \end{bmatrix}^T R\begin{bmatrix} I & 0 \end{bmatrix}$. By the requirement $\bar{\Theta} < 0$, it is clear that

$$L_{per} = \mathbb{E}\left\{ \int_0^T x^T(t)Rx(t)dt \right\} \leq V(z(0)) = \bar{L}_{per}(p_1,\ldots,p_n),$$

and

$$\lim_{T \to \infty} \mathbb{E}\left\{ \int_0^T x^T(t)Rx(t)dt \right\} \leq \bar{L}_{per}(p_1,\ldots,p_n)$$

$$= V_0(z(0)) + \sum_{i=1}^{n} p_i V_i(z(0)).$$

REFERENCES

[1] L. Weiss, A. Sanderson, and C. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal of robotics and automation*, vol. 3, no. 5, pp. 404–417, 1987.

[2] P. Corke and M. Good, "Dynamic effects in visual closed-loop systems," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 671–683, 1996.

[3] S. Hutchison, G. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

[4] R. Buyya *et al.*, "High Performance Cluster Computing: Architectures and Systems, Volume 1," *Prentice Hall PTR*, vol. 82, pp. 327–350, 1999.

[5] V. Sunderam *et al.*, "PVM: A framework for parallel distributed computing," *Concurrency Practice and Experience*, vol. 2, no. 4, pp. 315–339, 1990.

[6] C. Lee and M. Hamdi, "Parallel image processing applications on a network of workstations," *Parallel Computing*, vol. 21, no. 1, pp. 137–160, 1995.

[7] X. Li, B. Veeravalli, and C. Ko, "Distributed image processing on a network of workstations," *Int Journal of Computers and Applications*, vol. 25, no. 2, pp. 136–145, 2003.

[8] E. Marchand and F. Chaumette, "Virtual visual servoing: A framework for realtime augmented reality," in *Computer Graphics Forum*, vol. 21, pp. 289–298, Wiley, 2002.

[9] D. G. Lowe, "Distinctive image fearures from scale-invariant keypoints," in *Int Journal of Computer Vision*, 2004.

[10] M. A. Fischer and R. C. Bolles, "Random sampled consensus: A paradigm for modeling fitting with applications to image analysis and automated cartography," in *Communications of the ACM*, vol. 2, pp. 381–395, 1981.

[11] *http://www.csc.kth.se/ celle/*.

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RFC3550: RTP: A Transport Protocol for Real-Time Applications," *RFC Editor United States*, 2003.

[13] *http://www.gnu.org/software/ccrtp/*.

[14] H. Wu, C. Chen, J. Feng, K. Kühnlenz, and S. Hirche, "A switching control law for a networked visual servo control system," in *Proc IEEE Int Conf Rob and Automation (ICRA), Anchorage, Alaska*, 2010.

[15] K. Åström and B. Wittenmark, *Adaptive Control*. Addison-Wesley, 1955.

[16] R. M. Gray and L. D. Davisson, *An Introduction to Statistical Signal Processing*. Cambridge, UK: Cambridge University Press, 2004.