Performance-Oriented Networked Visual Servo Control with Sending Rate Scheduling

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

[1]Institute of Automatic Control Engineering (LSR)
[2]Institute for Advanced Study (IAS)
Technische Universität München
D-80290 München, Germany

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

Abstract— In order to speed up image processing in visual servoing, the distributed computational power across networks and appropriate data transmission mechanisms are of particular interest. In this paper, a high sampling rate of visual feedback is achieved by distributed computation on a cloud image processing platform. For target tracking with a networked visual servo control system, a switching control law considering the varying feedback delay caused by image processing and data transmission is applied to improve the control performance. A sending rate scheduling strategy aiming at saving the network load is proposed based on the tracking error. Experiments on a 7 degree-of-freedom (DoF) manipulator are carried out to validate the proposed approach. The proposed approach shows a similar control performance as a system without sending rate scheduling, however, beneficially with largely reduced network load.

I. INTRODUCTION

Real-time tracking of moving objects in unknown environments by vision systems has gained more and more interests in the domain of robot control. The visual information extracted from video sequences is utilized for vision-based control systems, e.g. target tracking, cooperative action, surveillance, and navigation. Previous efforts in these areas are far too numerous to be listed here exhaustively [1]–[4]. In this paper, we focus on the inherent problem of visual servo control systems: the long image processing delay. The feedback delay due to image processing limits the control performance and may lead to unstable systems [5].

The image processing delay consists primarily of delays caused by feature detection and feature matching, such as the Scale-Invariant Feature Transform (SIFT) [6] or Speed Up Robust Features (SURF) [7] extraction for target tracking. Though in recent years, image processing is accelerated by GPGPU (General-Purpose Computation on Graphics Processing Units) implementation [8]–[10], the about 20 ms time needed for SIFT feature extraction (on graphic card NVIDIA GeForce 8800, capable of detecting about 100 features in an image of size 640×480 pixels) still largely exceeds the typical updating interval of joint servo control (≤ 1 ms).

In order to achieve better control performance, it is desirable to obtain visual feedback at high sampling rate. With recent advances in computation and communication technologies, parallel computation based on networked computational resources has been proven to be an effective and
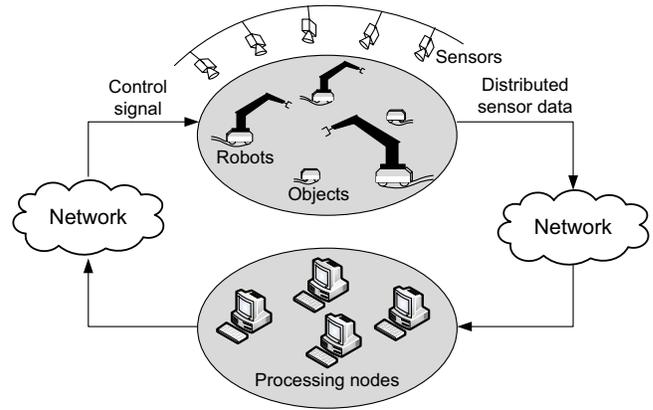
Fig. 1: Scheme of networked visual servo control systems with distributed sensors and distributed computations.

economical platform for high-performance computing [11]–[14]. In the light of the success of parallel computation, *Networked visual servo control (NVSC)* proposed in [15] is considered also in this work to deal with the low-sampling-rate problem of visual servoing, see Fig. 1 for a visualization. In NVSC systems, video grabbing, image processing algorithms, the controller, and actuators can be implemented on different processing nodes across a communication network. The data from sensors can be processed locally or sent to other computational resources connected to the network. The processing results are sent to a controller, where the input signals for the actuators are determined. In our previous work [15], a tracking problem with a NVSC system with a random sampling strategy was investigated and its efficacy was demonstrated in an experiment with two 1 degrees-of-Freedom (DoF) linear motor axis. In this paper, we further extend this approach by considering a sampling strategy dependent on the tracking error and its overall approach is validated in a 3D target tracking with a redundant robot manipulator (7-DoF robot arm).

An essential problem of NVSC systems is how to transmit the image data over the network in an efficient way. Considering a single channel image with a resolution of 640×480 pixels which has a size of about 300 KB, the network load will be more than 144 Mbps at a sampling
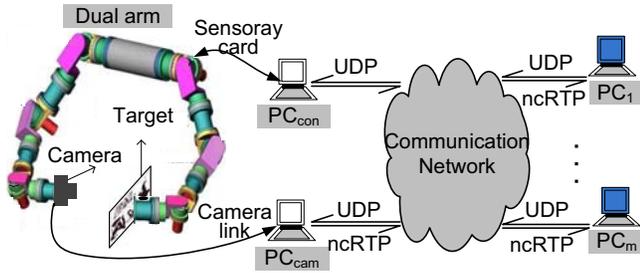
Fig. 2: Illustration of cloud image processing platform. Dual-arm [17]: redundant manipulator, 7-DoF each arm; PC_1, ..., PC_m: workstations equipped with GPUs (NVIDIA GeForce GT260) for image processing; PC_{cam}: image streaming server; PC_{con}: controller.

rate of ≥ 60 Hz. Although there are various existing network protocols in the public domain, they are not suitable for image transmission, e.g. UDP cannot transmit data larger than 64 KB and TCP has a larger absolute value and also larger variance of the transmission delay due to windowing behavior and retransmissions. Therefore, a novel real-time image data transmission protocol *ncRTP (networked control Realtime Transport Protocol)* developed in [16] is utilized in this paper. A *cloud image processing platform* based on ncRTP is established for parallel image processing. Since the high sampling rate achieved by the distributed computation would meanwhile lead to a high traffic load on the network, a *sending rate scheduling* considering the network capacity and the control performance constraints is proposed to reduce the network load.

The remainder of this paper is organized as follows: The cloud image processing platform for distributed computation is introduced in Section II. The 3D tracking problem with NVSC system on a 7-DoF manipulator and the switching controller considering varying feedback delay are discussed in Section III. In Section IV, the sending rate scheduling strategy for reducing the network load on the promise of guaranteed control performance is given. The experimental validation on a 7-DoF dual-arm robot and the performance comparison are presented in Section V.

II. CLOUD IMAGE PROCESSING PLATFORM

In this section, the platform of the NVSC system for target tracking with the redundant manipulator is illustrated. As discussed in Section I, image processing algorithms are in general time-consuming. The computational capacity becomes a bottleneck for visual servo control systems. To improve the control performance limited by the low-sample-rate problem, a high-sample-rate visual feedback is required.

A. GPGPU Cluster for Image Processing

With the development of GPGPU techniques in recent years, algorithms of high computational complexity are able to be implemented on a GPU. To exploit the computation capacity further, a GPGPU cluster is built in a LAN based on ncRTP, see Fig. 2. The platform consists of a robot manipulator, a streaming server, several computation nodes and

a controller. For real-time applications, the ncRTP protocol is installed on these nodes for transmitting large volume image data. In the following more details on the individual components are given.

- *Redundant manipulator*: A dual-arm robot with each arm having 7-DoF is chosen for target tracking, see also Fig. 2 and [17]. A high-speed camera is mounted on the end-effector of the left arm, while a target is mounted on the right arm. The objective for the arm with the camera is to track the motion of the target with the visual feedback provided by the camera.

- *Streaming Server PC_{cam}*: The streaming server is connected to the high speed camera and the sending process is scheduled by the real-time RTAI kernel. To reduce the latency, two real-time tasks run parallel. The first task polls the camera frame buffer periodically, while the other task sends the image data through ncRTP with an online determined interval d_k. The details of sending rate scheduling will be discussed in Section IV-B. The sending jitter is not more than several hundred nanoseconds. Moreover, multi-streaming is also supported by ncRTP.

- *Computation nodes PC_1,... ,PC_m*: Each computation node in this platform is equipped with a NVIDIA GeForce GT260 graphic card. Before image processing, the received image data is assembled to retrieve the complete image frame. Then the relative pose between the camera and the target is obtained by running the pose estimation algorithm. The results of image processing within a size of 64 KB are sent to the controller by UDP.

- *Controller PC_{con}*: The controller receives image processing results from the computation nodes and determines the joint torques to be applied to the manipulator, which drive the left robot arm to track the motion of the target. The tracking error is sent back to the streaming server PC_{cam} for sending rate scheduling.

This platform works in an event-based manner. When an image has been sent by the streaming server, the image processing on the workstation equipped with GPU is triggered immediately. The results are sent to controller directly after the image processing algorithms are returned.

B. Control Performance versus Number of Processing Nodes

With the platform described above, parallel image processing is established. In this section, the benefit of higher sampling rates for good control performance is demonstrated. Therefore a tracking problem with a PD controller on a 1-DoF manipulator is simulated. The manipulator is simulated by a double integrator. A stochastic time-varying feedback delay bounded by [15 ms, 35 ms] is simulated. Sampling rates of $\{40\,\text{Hz}, 80\,\text{Hz}\}$ with PD parameters chosen heuristically are tested for tracking control. If one node processes image with 40 Hz, $m \times 40\,\text{Hz}$ is achieved when m processing nodes are available over the network.

The simulation results are shown in Fig. 3. It is observed that, the control error is reduced with increased sampling
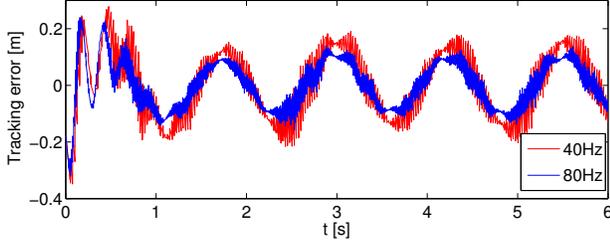
Fig. 3: Simulation results of control performance versus sampling rate of vision signal. Desired trajectory: $\sin(t) + \sin(5t + \frac{\pi}{5}) + 0.7\sin(2t + \frac{\pi}{6})$; delay: [15 ms, 35 ms].

rate, e.g. the mean control error of 80 Hz is 28.57% smaller than that of 40 Hz. Thus, it is reasonable to utilize the cloud image processing platform to obtain a high sampling rate for a performance-oriented visual servo control system.

III. SYSTEM MODELING

Based on the platform introduced above, a 3D tracking problem with networked visual servo control is investigated in this section. A camera-in-hand structure is selected for target detection and position-based visual servoing (PBVS) is applied. The system block diagram is shown in Fig.4. $X(t_k - \tau(t))$ is the current position of the manipulator in Cartesian space sampled at time instant t_k. Through the inverse kinematics model the joint angle q is calculated. A controller is applied to compute the joint torques. The feedback in Cartesian is also utilized for sending rate scheduling. The dynamics of the 7-DoF robot arm is given by

$$M(q)\ddot{q} + C(q,\dot{q}) + g(q) = \Gamma,$$

where $q \in \mathbb{R}^7$ denotes the angle displacement of each joint; $M(q) \in \mathbb{R}^{7\times 7}$ is the inertia matrix of the manipulator; $C(q,\dot{q}) \in \mathbb{R}^{7\times 7}$ represents the centrifugal force and Coriolis force; $g(q) \in \mathbb{R}^7$ is the gravity force of the manipulator; $\Gamma \in \mathbb{R}^7$ is the force variables. For further stability analysis and controller design, the robot arm is linearized by an inner nonlinear compensation loop based on the computed torqued control approach [18].

Consider the linearized manipulator dynamics and controller $u(t) = Kx(t_k)$ with t_k representing the sampling instant, the discrete-time closed-loop system is derived as

$$\dot{x}(t) = Ax(t) + BKx(t_k), \quad t_k \leq t < t_{k+1}, \quad (1)$$

where $x = [q\ \dot{q}]^T \in \mathbb{R}^{14}$ is the state; A and B are constant matrices with appropriate dimensions. $x(t_k)$ is delayed due to image processing and data transmission, see Fig. 5. It arrives at the controller with delay τ_k^{c+x} consisting of image processing delay τ_k^c and the transmission delay τ_k^x. d_k is the sampling interval between k-th and $k+1$-th samplings, and h_k is the interval between two arriving data at the controller side defined as

$$h_k = t_{k+1} + \tau_{k+1}^{c+x} - t_k - \tau_k^{c+x} = d_k + \tau_{k+1}^{c+x} - \tau_k^{c+x},$$

and the input value to the manipulator is hold constant at the last value until new data arrives and a new control can be computed.
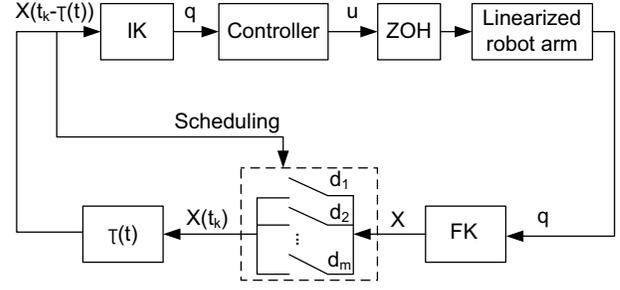
Fig. 4: System diagram of a NVSC system with time-varying feedback delay $\tau(t)$. $X \in \mathbb{R}^6$: position of end-effector in Cartesian space; q: joint angles; IK/FK: inverse/forward kinematics; d_1, d_2, ..., d_m: different sampling interval.

A. System reformulation

To develop a control algorithm such that the closed-loop system (1) is mean exponentially stable (MES), the system is reformulated into a continuous-time system with time-varying delay by means of input-delay approach [19]. Reconsider the time instant t_k as

$$t_k = t - (t - t_k) = t - \tau(t), \quad t \in [t_k + \tau^{c+k}, t_{k+1}^{c+k}).$$

$\tau(t)$ is the overall time-varying delay consisting of image processing, transmission delays τ_k^{c+x} and holding delay $h(t)$ for k-th sampling

$$\tau(t) = \tau_k^{c+x} + h(t), \quad h(t) \in [0, h_k),$$

with $\tau(t)$ bounded by $\underline{\tau} \leq \tau(t) \leq \bar{\tau}$. Substituting $x(t_k) = x(t - \tau(t))$ into the closed-loop system (1), it results in a continuous-time system with time-varying delay

$$\dot{x}(t) = Ax(t) + BKx(t - \tau(t)),$$
$$t \in [t_k + \tau_k^{c+x}, t_{k+1} + \tau_{k+1}^{c+x}), \quad (2)$$

with initial condition $x_0 = x(\theta), \theta \in [-\bar{\tau}, 0]$.

B. Switching controller

A switching control law depending on the current value of the time delay $\tau(t)$ is applied to achieve better control performance than a worst-case design where a single (non-switching) controller is required to stabilize for all possible time delays. The overall closed loop system equation is then represented by

$$\dot{x}(t) = Ax(t) + \sum_{i-1}^{n} \beta_i BK_i x(t - \tau(t)), \quad (3)$$

where K_i indicates a set of n controllers and the indicator function β_i indicates in which of the n subintervals the overall time delay τ currently falls with

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

and s_i are the bounds of the subintervals. According to the feedback delay interval $s_{i-1} \leq \tau < s_i$, the controller switches to a different control parameter set K_i. For the detailed stability analysis and controller design approach please refer to [15]. With the switching control law, the control performance of NVSC systems with varying feedback delay is improved as shown for a 1-DoF case in [15].
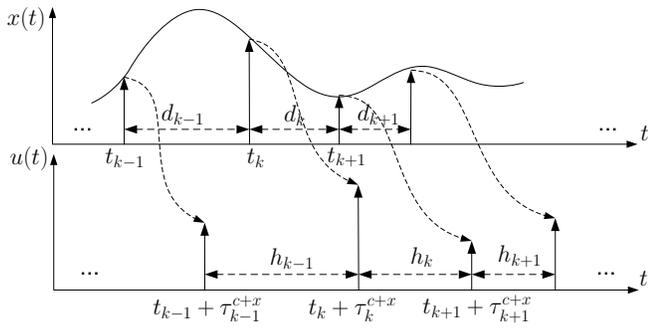
Fig. 5: Timing diagram of a NVSC system with random computation delay τ_k^c, transmission delay τ_k^x and sampling interval d_k.

IV. TRACKING ERROR BASED SENDING RATE SCHEDULING

In addition to the switching control law in Section III, also high sampling rate of visual feedback improves the control performance as demonstrated in Section II. With the cloud image processing platform introduced above, parallel image processing is realized by distributing the images to different processing nodes. However, high network load results from the large volume of transmitted image data. The goal of this section is to balance the control performance versus the network data flow of NVSC systems.

A. Cost Function

The trade-off between the control performance and the network cost is captured in the cost function

$$J = \lim_{T \to \infty} \mathbb{E}\left\{ \int_0^T x^T(t)Rx(t)dt + \frac{1}{T}\sum_{w=1}^m \int_0^T C_w(t)dt \right\}, \quad (4)$$

where $R > 0, R \in \mathbb{R}^{14\times14}$, and $C_w(t) > 0, C_w(t) \in \mathbb{R}$, denotes the network cost of the associated sampling interval d_k. With a smaller sampling interval d_k, a higher network cost factor C_w is assigned. The first term in (4) concerns the control performance cost and is bounded by a positive scalar $\bar{L}_{\text{per}}$ [16]. The second term represents the network cost over the run time T. If the occurrence probabilities of transfer and processing delays are known a priori, the occurrence probability $p_j^d, j = 1,\ldots,m$ of different sampling interval is determined by minimizing the cost function (4). In our previous work [16], the sending rate scheduling is designed as random selection of sending interval d_k at time instant t_k satisfying the predetermined occurrence probability of different sending intervals. However, a random selection strategy is not ideal. In this paper, a more reasonable sending rate scheduling concerning both occurrence probability and tracking error is proposed.

Remark 1: Within the context of this paper, sending rate scheduling means to determine an appropriate sending interval d_k at time instant t_k based on the tracking error and the occurrence probability obtained by minimizing the cost function (4).
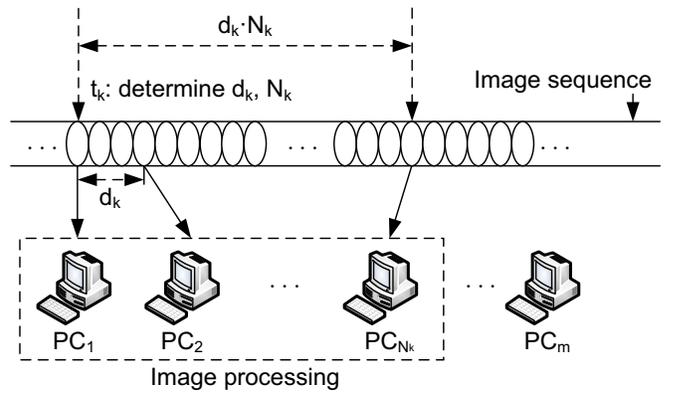
Fig. 6: Sending rate scheduling at the streaming server considering tracking error ΔX and occurrence probability p_j^d, $j = 1,\ldots,m$.

B. Policy of sending rate scheduling

As shown in Fig. 4, at time instant t_k a sampling interval d_k should be determined for the streaming server in the NVSC system. If there are m processing nodes available over the network, d_k takes the value in a finite set

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

where $\bar{\tau}^c = \max(\tau_k^c)$. The corresponding numbers of necessary processing nodes are

$$N_k = \frac{\bar{\tau}^c}{d_k} \in \left\{ m, \frac{m}{2},\ldots,1 \right\}, \quad N_k \in \mathbb{N}.$$

To determine d_k at time instant t_k, the tracking error ΔX in Cartesian space is considered. With m sending intervals available, $|\Delta X|$ is accordingly categorized into m intervals

$$r_{j-1} \le |\Delta X| < r_j, \quad 0 < r_{j-1} < r_j, \quad j = 1,\ldots,m, \quad (5)$$

where r_j are the bounds of the intervals. Generally speaking, a smaller sending interval should be assigned when the tracking error becomes larger. However, as mentioned in Remark 1 the occurrence probability of different sending interval obtained by minimizing the cost function should also be considered in sending rate scheduling. Therefore, at time instant t_k if tracking error falls into interval $r_{j-1} \le |\Delta X| < r_j$, the sending interval d_k is selected as

$$d_k = \begin{cases} \frac{\bar{\tau}^c}{j}, & \text{if} \quad p_j^d(t_k) < p_j^d, \\ \frac{\bar{\tau}^c}{j'}, & \text{otherwise}, \end{cases} \quad (6)$$

where $j' \neq j$, and $p_j^d(t_k)$ is the occurrence probability of sending interval $\frac{\bar{\tau}^c}{j}$ till instant t_k. Eq. (6) means that, d_k is assigned as $\frac{\bar{\tau}^c}{j}$ when the occurrence probability of sending interval $\frac{\bar{\tau}^c}{j}$ has not reached p_j^d till instant t_k. Otherwise, a smaller sending interval $\frac{\bar{\tau}^c}{j'}, j' > j$ is assigned considering a performance oriented control system, or if no $\frac{\bar{\tau}^c}{j'}, j' > j$ is available due to reached occurrence probability, a larger sending interval $\frac{\bar{\tau}^c}{j'}, j' < j$ is selected. Again, the occurrence probability is considered in choosing $\frac{\bar{\tau}^c}{j'}$.

The details of the image sending rate scheduling at the streaming server are illustrated in Fig. 6 as well as in Algorithm 1: 1) First, at the instant t_k, the sampling interval d_k and the number of necessary processing nodes N_k are determined according to tracking error ΔX and Eq. (6); 2) Then, for the time interval $[t_k, t_k + N_k \cdot d_k)$, the original image sequence is sub-sampled with sampling interval d_k and sent to processing nodes available over the network; 3) After sending the images, steps 1) and 2) are repeated from the instant $t_k + N_k \cdot d_k$ and so on.

Algorithm 1 Sending rate scheduling at streaming server

Require: ΔX at t_k, $p_j^d(t_k)$ and p_j^d, $j = 1,\ldots,m$
Task: determine d_k and the processing node for I_k (image sampled at time instant t_k)
   while $0 < t \le T$ **do**
      determine d_k and N_k according to (6) after sorting $|\Delta X|$ with (5).
      for t = $[t_k, t_k + N_k \times d_k)$ **do**
         $t_k = t$;
         send I_k to PC, (PC $\in \{PC_1,\ldots,PC_m\}$ represents the processing node available at t_k)
         $t = t_k + d_k$;
      end for
   end while

With the algorithm introduced above, the sending rate scheduling is realized at the streaming server in NVSC systems. As a result the network load caused by image transmission is effectively reduced. It has to be mentioned that in the current work, the sending interval is determined based on the tracking error. In the future, the target motion, network congestion and packet loss problem will be considered for sending rate scheduling as well.

Remark 2: There is no specific constraints in this work on selecting thresholds r_i for categorizing tracking error. In the experiments, r_i is heuristically selected.

V. EXPERIMENTS

In order to validate the proposed approach, experiments for a 3D tracking problem with a 7-DoF manipulator [17] are conducted. The end-effector of the manipulator is equipped with a high-speed camera (Mikrotron MC1363), while the target is mounted on the other arm of the end effector, as shown in Fig.7. The manipulator is connected to a PC running real-time RTAI/Linux. The control loop is implemented in MATLAB/SIMULINK blocksets. Standalone real-time is generated directly from the SIMULINK models. The image processing runs parallel on PC_1 and PC_2.

A. Image Processing

On the processing nodes, image processing algorithms for the 3D pose estimation are implemented. First, the image features are extracted from the image. To increase the accuracy of the pose estimation and speed up visual feedback, a GPGPU implementation of SURF [7] is applied by exploiting its massive parallel processing capability. After
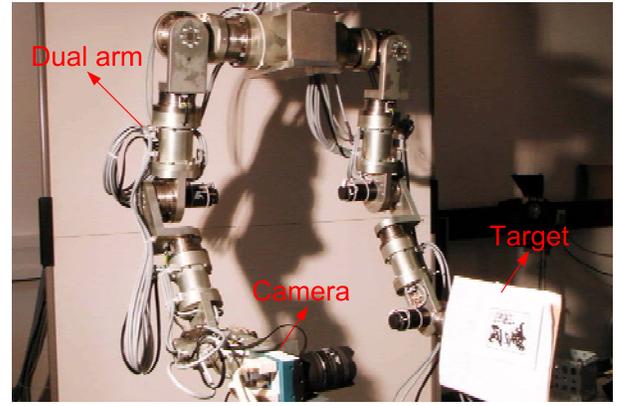
Fig. 7: Experimental setup with 7-DoF robotic arm, high-speed camera and moving target.

feature matching [20], RANSAC (RANdom Sample Consensus) [21] is selected for outliers rejection. Finally, the relative pose between the camera and the target is obtained by calculating the extrinsic parameters of the camera based on the matched feature pairs.

B. Sending Rate Scheduling

In the experiments, the camera runs at a fixed sampling rate of $80\,\mathrm{Hz}@640 \times 480$ pixels. Two sending rates $\{40\,\mathrm{Hz},\, 80\,\mathrm{Hz}\}$ are selected for the sending rate scheduling. The sending rate scheduler described in Section IV-B is realized as follows: the ncRTP steaming server PC_{cam} sends the images either at 80 Hz to PC_1 and PC_2 if $|\Delta X| > r1$, or at 40 Hz to only one PC if $|\Delta X| \le r1$. With this approach, the sampling rate of the feedback is bounded by [40 Hz, 80 Hz].

C. Experimental System Model

After linearization of the manipulator equations through computed torque control the system can be represented by seven decoupled subsystems. For each joint, the summation of the transmission and the computation delays is categorized into two intervals with $s_1 = 45$ ms. By optimizing the cost function (4), the occurrence probabilities of the two sampling intervals are obtained: $p_{40\mathrm{Hz}}^d = 31\%$, $p_{80\mathrm{Hz}}^d = 69\%$. The associated stabilizing state-feedback gains for each joint with $\tau(t) \le s_1$ are $K_{p1} = [120, 120, 60, 60, 30, 30, 30]$, $K_{d1} = [2, 2, 2, 2, 1, 1, 1, 1]$. The gains for $\tau(t) > s_1$ are $K_{p2} = [96, 96, 48, 48, 24, 24, 24]$, $K_{d2} = [2, 2, 2, 2, 1, 1, 1, 1]$. The tracking error threshold $r_1 = 0.01\,\mathrm{m}$ is heuristically selected.

To compare the control performance, a standard networked visual servo control system without sending rate scheduling (high network load), e.g. both PC_1 and PC_2 run at 40 Hz, is implemented.

D. Experimental Results

A sinusoidal function, which has an amplitude of 0.15 m and a frequency of 1 rad/s, serves as the reference trajectory for the target moving along one axis at the Cartesian space. The tracking error ΔX between the motion of left arm and that of right arm is discussed for performance evaluation. The approach proposed with sending rate scheduling based
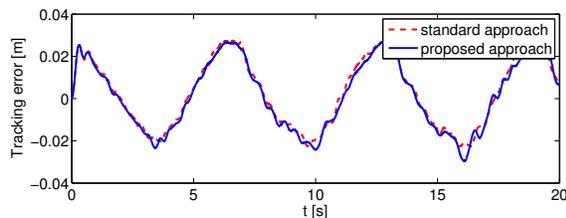
Fig. 8: The control error evolution of the proposed approach with data scheduling (solid line) and the standard approach without data scheduling (dash line).

on tracking error and the standard approach without sending rate scheduling are tested on the 7-DoF robot arm.

As shown in Fig. 8, a comparable control performance is achieved with and without sending rate scheduling. The proposed approach with sending rate scheduling has a mean tracking error $|\Delta X|_{\text{mean}} = 1.67\,\text{cm}$, which are similar to the mean tracking error of the high sampling rate design approach ($|\Delta X|_{\text{mean}} = 1.57\,\text{cm}$), see Table I. However, the network load (data flow) is 20.31% less than that of the high sampling rate design approach.

TABLE I: Control performance and network load.

	$|\Delta X|_{\text{mean}}$ [cm]	$|\Delta X|_{\text{yar}}$ [cm^2]	Network load [Mbps]
with scheduling	1.67	0.09	149.43
high sampling rate	1.57	0.18	187.50

The experimental results demonstrate a similar control performance of the proposed approach with lower network load than that of the standard approach.

VI. CONCLUSION

This paper presents a novel analysis and design approach for networked visual servo control systems with distributed computation and sending rate scheduling. Based on a real-time transport protocol ncRTP for image data transmission, a GPGPU cluster is built for parallel image processing. With the cloud image processing platform, high-sampling-rate visual feedback is achieved. A switching control law is applied to consider the varying feedback delay caused by image processing and data transmission. Moreover, a sending rate scheduling at the streaming server in NVSC systems is designed based on the tracking error to relieve the network burden caused by large volume image data transmission. The proposed approach is validated by experiments on a 7-DoF manipulator. The results demonstrate comparable control performance of the proposed approach with lower network cost than the conventional counterpart.

The future work is concerned with optimizing the sending rate scheduling by considering not only tracking error, but also motion dynamics of the target, network congestion and packet loss.

VII. ACKNOWLEDGMENTS

REFERENCES

[1] K. Hashimoto, *Visual servoing: real-time control of robot manipulators based on visual sensory feedback*. World scientific, 1993.

[2] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm Computing Surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[3] O. Javed and M. Shah, "Tracking and object classification for automated surveillance," *Computer Vision, ECCV 2002*, pp. 439–443, 2006.

[4] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334–352, 2004.

[5] 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.

[6] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer Vision–ECCV 2006*, pp. 404–417, 2006.

[8] V. Vineet and P. Narayanan, "CUDA cuts: Fast graph cuts on the GPU," *Proc. of IEEE computer Society Conference Vision and Pattern Recognition*, pp. 1–8, 2008.

[9] S. Sinha, J. Frahm, M. Pollefeys, and Y. Genc, "GPU-based video feature tracking and matching," in *EDGE, Workshop on Edge Computing Using New Commodity Architectures*, vol. 278, Citeseer, 2006.

[10] T. Xu, T. Pototschnig, K. Kühnlenz, and M. Buss, "A high-speed multi-GPU implementation of bottom-up attention using CUDA," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pp. 1120–1126, Institute of Electrical and Electronics Engineers, 2009.

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

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

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

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

[15] H. Wu, C. Chen, J. Feng, K. Kühnlenz, and S. Hirche, "A switching control law for a networked visual servo control system," in *Proc. of IEEE International Conference on Robotics and Automation, Anchorage, Alaska*, 2010.

[16] H. Wu, L. Lou, C.-C. Chen, K. Kühnlenz, and S. Hirche, "Distributed Computation and Data Scheduling for Networked Visual Servo Control Systems," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[17] B. Stanczyk, *Development and Control of an Anthropomorphic Telerobotic System*. PhD thesis, Technische Universität München, 2006.

[18] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer-Verlag New York Inc, 2008.

[19] K. Åström and B. Wittenmark, *Adaptive Control*. Addison-Wesley, second ed., 1995.

[20] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pp. 331–340, INSTICC Press, 2009.

[21] 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.