ISA Transactions 68 (2017) 116–126

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

Research article

Development of a decentralized multi-axis synchronous control approach for real-time networks ☆

Xiong Xu [a,*], Guo-Ying Gu [b], Zhenhua Xiong [b], Xinjun Sheng [b], Xiangyang Zhu [b]

[a] School of Mechanical Engineering, Shanghai Institute of Technology, Shanghai 201418, China
[b] State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 12 November 2015
Received in revised form
23 January 2017
Accepted 17 March 2017
Available online 23 March 2017

Keywords:
Networked control system
Multi-axis motion
Decentralized control
Position synchronization
Cross-coupled control
Real-time network

ABSTRACT

The message scheduling and the network-induced delays of real-time networks, together with the different inertias and disturbances in different axes, make the synchronous control of the real-time network-based systems quite challenging. To address this challenge, a decentralized multi-axis synchronous control approach is developed in this paper. Due to the limitations of message scheduling and network bandwidth, error of the position synchronization is firstly defined in the proposed control approach as a subset of preceding-axis pairs. Then, a motion message estimator is designed to reduce the effect of network delays. It is proven that position and synchronization errors asymptotically converge to zero in the proposed controller with the delay compensation. Finally, simulation and experimental results show that the developed control approach can achieve the good position synchronization performance for the multi-axis motion over the real-time network.

1. Introduction

The distributed control systems with a real-time network is becoming promising in modern industrial and commercial applications equipped with a number of actuators and sensors [1], for instance, humanoid robots, printing machines, automobiles, computer numerical control (CNC) machining centers, and so on. Such kind of systems is generally known as the networked motion control systems (NMCSs), where functional modules such as actuators, sensors, and controllers are spatially interconnected and distributed with a communication network [2]. Compared with traditional centralized control systems with directly wiring devices together, NMCSs provide several advantages in terms of remote-control capability, scalability and reliability. Besides, NMCSs reduce the problems of transmit-length limitation and wiring connection, and decrease reconfiguration, installation, and maintenance costs [3].

In a networked motion control system, a large number of feedback and command messages should be periodically communicated among distributed nodes with a high speed [3]. To address such challenges, a lot of efforts have been made in recent years, which can be categorized into the following two parts: networking technologies and control over network [4]. In order to develop the network appropriate for real-time applications, the main concern in networking technologies is to improve the network quality of service (QoS) [5] such as routing control, congestion reduction and communication protocol. On the other hand, control over network focuses on controller design over the network to maintain the control quality of performance (QoP) by compensating for the network-induced delay or packet dropout. At the same time, synchronous errors among axes are important aspects that significantly affect the motion accuracy in a multi-axis motion control system [6]. Different from the traditional centralized control systems, besides the inertias and disturbances factors, the message scheduling and the network-induced delays also influence the synchronous performance in network-based systems.

However, literatures focusing on NMCSs study combined with the synchronous control are few reported. It remains a challenging problem how to design a decentralized multi-axis synchronous controller with considering the network-induced factors and prove the asymptotic stability of the global system. To this end, we present a decentralized multi-axis synchronous control approach in this paper to realize the accurate position tracking and synchronizing motion, dealing with the factors such as the different inertias, different disturbances, the message scheduling and the network-induced delays of the real-time network together. Based on the detailed analysis of the problems of the message scheduling in the real-time network, we choose a subset of preceding-axis pairs as the position synchronization error. And a motion message

☆This research was supported in part by Shanghai Sailing Program under Grant 14YF1414200, the Science and Technology Commission of Shanghai Municipality under Grant 16JC1401000, and State Key Laboratory of Robotics and System (HIT).

* Corresponding author.
E-mail address: xuxiong19850707@163.com (X. Xu).

http://dx.doi.org/10.1016/j.isatra.2017.03.012
0019-0578/

estimator is introduced in the synchronous controller to estimate the current position errors of preceding axes to reduce the effect of network-induced delays. Simulations and experiments are conducted on networked multi-axis motion control systems to verify the effectiveness of the developed approach. The networked simulation system is designed through the Matlab/Simulink and the TrueTime toolbox [7], while the experimental set-up is built based on smart networked motion control nodes, a four-axis table and the real-time Ethernet communication. It should be also noted that compared with our previous developed system [8], the time synchronization scheme is added to the real-time communication cycle and the time synchronization performance is also validated in the test system. Both simulation and experimental results demonstrate that the proposed synchronous control approach achieves good position synchronization performance in the networked system for multi-axis motion control.

The remainder of this paper is organized as follows. In Section 2, we survey existing results on networked motion control systems. and the problem formulation is devoted in Section 3. In Section 4, we design a decentralized synchronous controller for networked motion control systems. In Section 5 and Section 6, simulations and experiments are performed to verify the proposed synchronization control approach. Conclusions are given in the last section.

2. Related work

2.1. Networking technologies

While choosing the communication network for the multi-axis motion control system, the main issues include time-synchronized actions between field devices, time-deterministic communication, reliability and availability. In the literature, a number of real-time control networks, such as Profibus, control area network (CAN), FIP, serial real-time communication specification (SERCOS), have been developed for NMCSs during the past decade [9–12]. As an alternative, due to the advantages of high transmission speed, deterministic communication, and low price, the real-time Ethernet (RTE) has recently been developed for information exchanges in real-time motion control systems [13,14]. Besides, we can adopt a double-ring topology architecture or optical fiber as the transmission medium to improve the reliability of the RTE, which indicates the effect of the packet dropout can be neglected in these real-time networks. This is also the reason why the packet dropout will not be considered in this work.

2.2. Control over network

After using a proper real-time communication protocol to guarantee the network QoS, we also need an advanced controller to guarantee the control QoP. In terms of the controller design, the current research in NMCSs focuses on control system design and control strategies when a traditional feedback control system is closed through a network. This control system can also be denoted as a networked control system (NCS). For example, NCS models with network-induced delay were developed by Zhang and Branicky [15]. A theoretical framework was presented by Yook et al. [16] to develop an optimal network architecture. The modelling and optimal controller design of NCSs with multiple communication delays were investigated by Lian et al. [17]. A controller gain adaptation was proposed in [5] to handle the changes in QoS requirements. Besides, there are many algorithms in networked control systems to compensate the network-induced delays, such as Smith predictor [18], model-based control [19], networked predictive control [20] and communication disturbance observer [21]. But these delay compensation algorithms were complicated and required extensive on-line calculations.

2.3. Traditional multi-axis synchronous control

The reported works were only contributed to the stability analysis and controller development of single closed-loop in network environments. In traditional centralized control systems, different inertias and disturbances in different axes are the main factors which cause synchronous problem. To improve the synchronization performance for centralized systems, many efforts have been devoted. Koren [22] proposed a cross-coupled control (CCC) for biaxial motion systems. Various improved contour controllers were later presented to further reduce synchronous errors of different axes [23–29].

2.4. Multi-axis synchronous control over network

As aforementioned, it remains a challenging and open problem how to design a decentralized multi-axis synchronous controller over a network. Recently, Nuño [30] proposed an adaptive synchronous controller to achieve the networks' synchronization in nonidentical Euler–Lagrange systems. But noted that constant network delays were assumed and delay compensation was not considered in their controller design. Besides, we made some attempts on this issue in [31] and have proposed a time-stamped cross-coupled controller in networked CNC systems. But this controller was based on biaxial motion systems and was verified only by numerical simulation. In order to validate the synchronous controller in applications, a networked multi-agent system based on real-time Ethernet was developed in our previous work [8]. This paper furthers these research and investigates a decentralized multi-axis synchronous control over real-time networks.

3. Problem formulation

Generally speaking, the multi-axis motion control system over real-time networks includes a set of smart networked motion control nodes, which are spatially interconnected and distributed by one real-time network. And its typical network architecture is the ring topology due to the low cabling cost. As shown in Fig. 1, the master node manages the whole digital communication and achieves the key functions of a reference position generator, for instance, trajectory interpolation, velocity planning, code interpretation, etc. Simultaneously, position-loop control and cross-coupled control are implemented in the distributed motion control nodes to realize the parallel computing and fast response. There are some applications, like soft-CNC systems [32,33], which integrate position-loop control, velocity-loop control, and even current-loop control in the master node. However, the main difficulty in these applications arises from the tradeoff between the extensive real-time calculations of the centralized multi-axis controller and the constrained computation capability in the master node. This has been the motivation of this work to design a decentralized synchronous controller for real-time network-based systems. Compared to the traditional centralized architecture, the decentralized architecture can improve the response speed and the modularity of the system through combining the merits of the distributed computing and the real-time network. Finally, we can summarize our problem as:

Problem: Consider that a networked multi-axis system with n axes requires position synchronization of all distributed axes. As in [24], this task requirement can be transformed to a relationship that the position errors of all axes must be regulated, namely

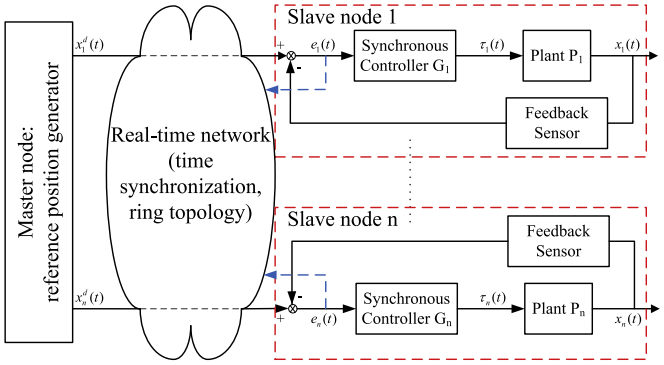


Fig. 1. Control architecture over the networked multi-axis motion control system.

$$e_1 = e_2 = \cdots = e_n, \tag{1}$$

where e_i is the position error of the ith axis and $e_i = x_i^d - x_i$. Define the desired and actual positions of the ith axis as x_i^d and x_i. In the next section, this synchronization control problem will be solved in following three aspects:

1. Design of the decentralized synchronization architecture.
2. Consideration of the message scheduling and network-induced delays in decentralized synchronous controller.
3. Analysis of the asymptotic stability of the entire networked system.

In addition, we focus on the decentralized motion synchronization according to the assumption that all the distributed clocks are synchronized. This can be realized by using a time synchronization protocol, such as network time protocol or IEEE 1588 standard. The description of the realization of the hardware-based time synchronization strategy is detailed in our previous work [34].

4. Decentralized synchronous control

In this section, a decentralized synchronous controller will be designed for networked motion control systems. The goal of the controller is to achieve the motion synchronization for the distributed system with considering the message scheduling and the network-induced delays in real-time networks. The following development consist of the design of the decentralized synchronization architecture, the definition of the position synchronization error, the design process of the decentralized synchronous controller and the stability analysis of this controller.

4.1. Decentralized synchronization architecture

As shown in Fig. 2, we adopt a time-based control architecture to achieve multi-axis motion synchronization based on the high-precision time synchronization. In this distributed control architecture, time-critical system actions depend on a system-wide time scale achieved by the high-precision time synchronization, instead of message receipt times [35,36]. Then, all periodic tasks with a sample period of T_s, including the reference position generator in the master node and the servo controller of each slave node, are therefore scheduled by these synchronized clocks. This means the start of the trajectory planning in the master node and the position sampling in each slave node can be synchronized on the order of tens of nanoseconds [34]. The synchronized sampling position is used to calculate the feedback position error, which is then stored in a buffer to be sent out. At the same time, the

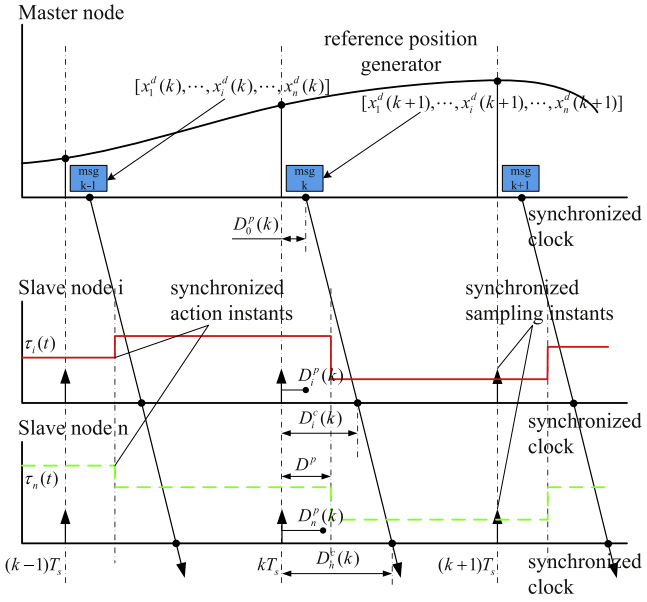


Fig. 2. Timing diagram of the decentralized synchronization strategy.

reference position generator executes the one-step look-ahead interpolation algorithm and sends the position information $[x_1^d(k+1), \ldots, x_i^d(k+1), \ldots, x_n^d(k+1)]$ in the kth frame. This lumped frame is then passed through each slave node (i.e. slave node i), which processes the frame by reading the command position $x_i^d(k+1)$ or some axes' feedback data, and writing the feedback position error $e_i(k)$. The command position is also buffered to be used by the synchronous controller in the $(k+1)$ th servo period. The timing diagram of the entire process is shown in Fig. 2, where two types of time delays, i.e., processing delay and communication delay are defined. Processing delays occur at the reference position generator and the decentralized synchronous controllers are represented as $D_0^p(k)$ and $D_i^p(k)$ respectively. In order to synchronize the actuation among distributed axes, we adopt the deferred actuation strategy by updating the actuation waiting for a specific time slack $D^p - D_i^p(k)$, where $D^p \geq \max\{D_1^p(k), \ldots, D_i^p(k), \ldots, D_n^p(k)\}$ and can be optimized through the method described in [37]. Note that the servo controller achieves the minimal output delay which is just equal to D^p. Besides, communication delay between the master node and the slave nodes are denoted as $D_i^c(k)$. Here, the network-induced delays are assumed to be bounded by the sampling period T_s. Moreover, multi-axis synchronous control with randomly time-varying delays is hardly to solve out. This is also the reason why constant network delays were assumed in the current state of the art [30]. In order to make these network-induced delays time-invariant, a buffer at the slave node is introduced to have the length of one sample period. Thus, these time delays are constant and can be easily compensated in the following proposed synchronous controller.

4.2. Position synchronization errors

Considering a general mechanical system with n axes, the system dynamics can be expressed as

$$H(x)\ddot{x} + C(x, \dot{x})\dot{x} = \tau, \tag{2}$$

where $x = \{x_i\}$ is the $n \times 1$ vector of axis displacements, $\tau = \{\tau_i\}$ is the $n \times 1$ vector of input forces, $H(x) = \{H_i(x_i)\}$ is the positive definite inertia matrix, and $C(x, \dot{x}) = \{C_i(x_i, \dot{x}_i)\}$ denotes the $n \times 1$ vector of centripetal and coriolis forces.

Define a vector of position synchronization errors as

$$\varepsilon = Te, \tag{3}$$

where $e = [e_1, e_2, \ldots, e_n]'$, $\varepsilon = [\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n]'$, the notation "′" means the transpose of a matrix, and T indicates a synchronization transformation matrix. As in [38], a coupled position error is used in this work to link ε and e together. It is defined as

$$E = e + \alpha\varepsilon. \tag{4}$$

In this equation, the α is a control gain matrix with diagonal and positive definite properties. Substituting (3) into (4) yields

$$E = (I + \alpha T)e, \tag{5}$$

where I means a unit matrix. From Eq. (5), $E \to 0$ implies $e \to 0$ when the inverse of $(I + \alpha T)$ exists. Further, $e \to 0$ implies $\varepsilon \to 0$.

Then, we should further analyze the message scheduling problem in the decentralized synchronization architecture of Fig. 2. First, in a ring topology, the first networked motion control node (node 1) and the last networked motion control node (node n) are separated by multi-hop nodes. Hence, there exists a sensorial problem between these two networked motion control nodes. Second, in some real-time networks, such as the EtherCAT network with a ring topology [39], a lumped frame is periodically sent by the master node and passed through all the slave motion control nodes. The basic operating principle is that all slave nodes in the EtherCAT network can read command data (such as reference position) and write feedback data (such as position error) to the EtherCAT telegram as it passes by. It is easy for each slave node to acquire the feedback position errors of its preceding nodes, but it is hard to obtain its subsequent nodes' feedback data. The first problem also exists in a line formation of multiple mobile robots, as mentioned in [40]. They utilize a high-level planner to make the two robots acquire the information between each other. This method may settle the second problem as well. However, the method would increase the network-induced delays in the networked multi-axis motion control system because it needs more network bandwidth to transfer all slave nodes' position errors and each slave node can only receive the feedback position errors in next lumped frame. Besides, with the requirement of real-time control, the feedback and command information is exchanged among distributed nodes within a designed sample period [41]. Although the throughput of the EtherCAT network is really high, the maximum nodes will be halved since the total lumped frames in each period is doubled.

Remark 1. *(Communication constraints)*: Considering the problem of the message scheduling in the networked multi-axis motion control system, the position synchronization error should be defined as a subset of preceding-axis pairs. Therefore, T satisfies the constraint: T is a lower triangular matrix, that means $T = (t_{ij})$ with $t_{ij} = 0$ for $i < j$.

In this work, we choose the value of T as

$$T = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ -\dfrac{1}{n-2} & \cdots & -\dfrac{1}{n-2} & 1 & 0 \\ -\dfrac{1}{n-1} & -\dfrac{1}{n-1} & \cdots & -\dfrac{1}{n-1} & 1 \end{bmatrix}. \tag{6}$$

That means the position synchronization errors should be rewritten as

$$\begin{cases} \varepsilon_1 = e_1 - e_1 \\ \varepsilon_2 = e_2 - e_1 \\ \varepsilon_3 = e_3 - \dfrac{1}{2}(e_1 + e_2) \\ \varepsilon_4 = e_4 - \dfrac{1}{3}(e_1 + e_2 + e_3) \\ \quad\vdots \\ \varepsilon_n = e_n - \dfrac{1}{n-1}\displaystyle\sum_{i=1}^{n-1} e_i \end{cases} \tag{7}$$

Here, no more network bandwidth is required and all the preceding axes' feedback information is fully utilized. Meanwhile, the computation time is still very small. Compared to [24], T is not a symmetric matrix here, but the inverse of $(I + \alpha T)$ also exists. Moreover, $(I + \alpha T)$ is positive definite as the α is selected to be small enough. Note that the synchronization goal $e_1 = e_2 = \cdots = e_n$ is achieved automatically when all position synchronization errors in (7) are zero.

In the following section, a decentralized synchronous control for position synchronization will be provided in detail to prove that the coupled position error E converges to zero as $t \to \infty$.

4.3. Decentralized synchronous control

In practical applications, a model-free synchronous control is often used because it demands little online computation by avoiding the usage of the system dynamic model. The proportional and derivative (PD)-type model-free synchronous controller is given by

$$\tau = K_H\ddot{x}^d + K_C\dot{x}^d + K_P(I + \alpha T)e + K_D(I + \alpha T)\dot{e} + (I + \alpha T')^{-1}K_e\dot{e} + sign(\dot{E})K_N, \tag{8}$$

where T' means the transpose of the synchronization transformation matrix T, K_H and K_C denote positive feedforward control gains, K_P, K_D, and K_e indicate positive control gains. Here, we give the following assumptions: x^d is bounded up to its second derivative and $H(x)$ and $C(x, \dot{x})$ are bounded so long as their arguments are bounded. Note that the control gains K_H and K_C are used to replace the modeling parameters $H(x)$ and $C(x, \dot{x})$ so that the synchronous controller is independent of the model. The fifth term of the right-hand side of (8) is added because of system stability. The last term in (8) is a sign function to handle the nonlinear effect, which is caused due to the difference of the control gains K_H and K_C and the model parameters $H(x)$ and $C(x, \dot{x})$. The parameter K_N therefore satisfies the following condition:

$$K_N = \Delta_H\left\|\ddot{x}^d\right\| + \Delta_C\left\|\dot{x}^d\right\|, \tag{9}$$

where Δ_H and Δ_C are scalars.

Remark 2. *(Distributed computation constraints):* In order to calculate the τ_i only using the preceding axes' position errors, we should choose K_H, K_C, K_P, and K_D as lower triangular matrices or diagonal matrices. Besides, the $I + \alpha T$ is a lower triangular matrix as well. Thus, the first four terms of the right-hand side in (8) can be calculated based on the preceding axes' information. But how to make the fifth term have the same characteristic. To work out this problem, let $K_M = mI \triangleq (I + \alpha T')^{-1}K_e$, where the notation "$\triangleq$" means "is defined as", K_M is a scalar matrix, and m is a scalar with $m > 0$. Then, the K_e can be calculated as

$$K_e = m(I + \alpha T') = \begin{bmatrix} m & -m\kappa & -\frac{m\kappa}{2} & \cdots & -\frac{m\kappa}{n-1} \\ 0 & m(\kappa+1) & -\frac{m\kappa}{2} & \cdots & -\frac{m\kappa}{n-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & m(\kappa+1) & -\frac{m\kappa}{n-1} \\ 0 & 0 & \cdots & 0 & m(\kappa+1) \end{bmatrix}. \tag{10}$$

In this equation, the κ is a scalar with $\alpha = \kappa I$ and $\kappa > 0$. Thus, the decentralized synchronous controller, ignoring the network-induced delays in the system, is expressed as

$$\tau = K_H\ddot{x}^d + K_C\dot{x}^d + K_PE + K_D\dot{E} + mI\dot{e} + sign(\dot{E})K_N. \tag{11}$$

Remark 3. *(Delay compensation in controller)*: According to Eq. (11), the synchronous controller needs all actual position errors of preceding axes to calculate the position synchronization error at the beginning of each sample period T_s. Because of network-induced delays, the sensor of the ith ($i = 1, \ldots, k-1$) axis measures the position error $e_i(t)$ and sends it to the kth ($k = 2, \ldots, n$) axis as the feedback signal $e_i'(t)$, which have the relation of

$$e_i'(t) = e_i(t - D_{ik}), \tag{12}$$

where D_{ik} is the time delay caused by the transition of the measured signal from the ith axis to kth axis. As aforementioned, these time delays are constant and meet

$$\begin{cases} D_{ik} = T_s \\ D_{kk} = 0 \end{cases} \quad (i = 1, \ldots, k-1; \quad k = 2, \ldots, n). \tag{13}$$

In addition, most motion measurements in real applications are smooth and predictable, and can be described as a Taylor expansion with a suitable order. As in [2], a 3rd-order Taylor message estimator $F(z)$ was proposed by Hsieh to reduce the data-dropout effect, which can be expressed in the z-transform as

$$F(z) = \frac{21}{8}z^{-1} - \frac{19}{8}z^{-2} + \frac{7}{8}z^{-3} - \frac{1}{8}z^{-4}. \tag{14}$$

It was verified that the message estimator with the 3rd-order Taylor expansion is more suitable in real motion systems. We therefore use the same Taylor message estimator here, to obtain the current position errors of preceding axes with the form of

$$\hat{e}_i(t) = \frac{21}{8}e_i(t - T_s) - \frac{19}{8}e_i(t - 2T_s) + \frac{7}{8}e_i(t - 3T_s) - \frac{1}{8}e_i(t - 4T_s), \tag{15}$$

where $e_i(t - T_s)$, $e_i(t - 2T_s)$, $e_i(t - 3T_s)$ and $e_i(t - 4T_s)$ mean the past four sequential position errors.

Let $I + \alpha T = P + Q$, where P is a diagonal matrix, Q is a lower triangular matrix, and all the diagonal elements of the Q are equal to zero. In addition, P approximates to a unit matrix and Q is close to a null matrix under the condition that α is selected small enough. Then, the decentralized synchronous controller with the delay compensation can be written as (16):

$$\begin{aligned} \tau = {} & K_H\ddot{x}^d + K_C\dot{x}^d + K_P\Big(Pe + \frac{21}{8}Qe(t - T_s) - \frac{19}{8}Qe(t - 2T_s) \\ & + \frac{7}{8}Qe(t - 3T_s) - \frac{1}{8}Qe(t - 4T_s)\Big) \\ & + K_D\Big(P\dot{e} + \frac{21}{8}Q\dot{e}(t - T_s) - \frac{19}{8}Q\dot{e}(t - 2T_s) + \frac{7}{8}Q\dot{e}(t - 3T_s) \\ & - \frac{1}{8}Q\dot{e}(t - 4T_s)\Big) + mI\dot{e} + sign\Big(P\dot{e} + \frac{21}{8}Q\dot{e}(t - T_s) \\ & - \frac{19}{8}Q\dot{e}(t - 2T_s) + \frac{7}{8}Q\dot{e}(t - 3T_s) - \frac{1}{8}Q\dot{e}(t - 4T_s)\Big)K_N \end{aligned} \tag{16}$$

4.4. Stability analysis

Theorem 1. *For the decentralized synchronous controller with the delay compensation (16), the system is stable during the motion and guarantees $e \to 0$ and $\varepsilon \to 0$ as time $t \to \infty$ at the final desired position based on the conditions of:*

1) the sample period T_s and the control gain α are selected small enough, and K_e is properly selected to satisfy $\lambda_{\min}\{K_e\} = m \geq c_1\|x\| + c_2\|\dot{x}\|$, where $\lambda_{\min}\{K_e\}$ is the minimum eigenvalue of the gain matrix K_e, c_1 and c_2 are Lipschitz constants;

2) scalars Δ_H and Δ_C are large enough so that $\Delta_H \geq \|K_H - H(x)\|$ and $\Delta_C \geq \|K_C - C(x, \dot{x})\|$.

Proof. A Lyapunov function candidate is defined as

$$V = \frac{1}{2}\dot{e}'(I + \alpha T')H(x)\dot{e} + \frac{15T_s^2}{2}\dot{e}'(I + \alpha T')K_PQ\dot{e} + \frac{1}{2}E'K_PE. \tag{17}$$

Here, $(I + \alpha T')(H(x) + 15T_s^2K_PQ)$ is positive definite when the control gain α and the sample period T_s are selected small enough. The derivative of V versus time is expressed as

$$\begin{aligned} \dot{V} = {} & \dot{e}'(I + \alpha T')H(x)\ddot{e} + \frac{1}{2}\dot{e}'(I + \alpha T')\dot{H}\dot{e} \\ & + 15T_s^2\dot{e}'(I + \alpha T')K_PQ\ddot{e} + E'K_P\dot{E}. \end{aligned} \tag{18}$$

□

If the sample period T_s is selected to be small enough, $e(t - iT_s) = e - iT_s\dot{e} + \frac{(iT_s)^2}{2}\ddot{e}$, $\dot{e}(t - iT_s) = \dot{e} - iT_s\ddot{e}$, $(i = 1, \ldots, 4)$, which can be obtained by the Taylor expansion. The decentralized synchronous controller with the delay compensation (16) is therefore rewritten as

$$\begin{aligned} \tau = {} & K_H\ddot{x}^d + K_C\dot{x}^d + K_P(Pe + Qe + 15T_s^2Q\ddot{e}) \\ & + K_D(P\dot{e} + Q\dot{e}) + mI\dot{e} + sign(P\dot{e} + Q\dot{e})K_N \\ = {} & K_H\ddot{x}^d + K_C\dot{x}^d + K_PE + 15T_s^2K_PQ\ddot{e} + K_D\dot{E} \\ & + mI\dot{e} + sign(\dot{E})K_N. \end{aligned} \tag{19}$$

Substituting (19) into (2) yields the closed-loop dynamics

$$\begin{aligned} & H(x)\ddot{e} + C(x, \dot{x})\dot{e} + K_PE + 15T_s^2K_PQ\ddot{e} + K_D\dot{E} \\ & \quad + mI\dot{e} + N + sign(\dot{E})K_N = 0, \end{aligned} \tag{20}$$

where $N = (K_H - H(x))\ddot{x}^d + (K_C - C(x, \dot{x}))\dot{x}^d$, which is bounded under the aforementioned assumptions. Multiplying both sides of (20) by $\dot{E}'$ yields

$$\begin{aligned}
&\dot{E}'H(x)\ddot{e} + \dot{E}'C(x,\dot{x})\dot{e} + \dot{E}'K_PE + 15T_s^2\dot{E}'K_PQ\ddot{e} \\
&\quad + \dot{E}'K_D\dot{E} + \dot{E}'mI\dot{e} + \dot{E}'N + \left\|\dot{E}'\right\|K_N \\
&= \dot{e}'(I+\alpha T')H(x)\ddot{e} + \dot{e}'(I+\alpha T')C(x,\dot{x})\dot{e} + \dot{E}'K_PE \\
&\quad + 15T_s^2\dot{e}'(I+\alpha T')K_PQ\ddot{e} + \dot{E}'K_D\dot{E} \\
&\quad + \dot{e}'K_e\dot{e} + \dot{E}'N + \left\|\dot{E}'\right\|K_N = 0.
\end{aligned} \tag{21}$$

Substituting (21) into (18) yields

$$\begin{aligned}
\dot{V} &= \dot{e}'(I+\alpha T')\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\dot{e} - \dot{E}'K_D\dot{E} \\
&\quad - \dot{e}'K_e\dot{e} - \dot{E}'N - \left\|\dot{E}'\right\|K_N \\
&\le -\dot{E}'K_D\dot{E} - \dot{e}'\left(K_e - \alpha T'\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\right)\dot{e} \\
&\quad + \left\|\dot{E}'\right\|\|N\| - \left\|\dot{E}'\right\|K_N \\
&\le -\dot{E}'K_D\dot{E} - \dot{e}'\left(\lambda_{\min}\{K_e\} - \left\|\alpha T'\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\right\|\right)\dot{e} \\
&\quad - \left\|\dot{E}'\right\|(K_N - \|N\|).
\end{aligned} \tag{22}$$

It should be noted that $\dot{e}'\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\dot{e} = 0$ because $\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)$ is skew symmetric. As both $H(x)$ and $C(x,\dot{x})$ satisfy the Lipschitz condition [24], we have $\left\|\alpha T'\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\right\| \le c_1\|x\| + c_2\|\dot{x}\|$, where c_1 and c_2 are Lipschitz constants. If the control gain K_e is properly selected to satisfy $\lambda_{\min}\{K_e\} = m \ge c_1\|x\| + c_2\|\dot{x}\|$, then we have $\dot{e}'\left(\lambda_{\min}\{K_e\} - \left\|\alpha T'\left(\frac{1}{2}\dot{H}(x) - C(x,\dot{x})\right)\right\|\right)\dot{e} \ge 0$.

Under the conditions as in Theorem 1, namely $\Delta_H \ge \|K_H - H(x)\|$ and $\Delta_C \ge \|K_C - C(x,\dot{x})\|$, it follows that

$$\begin{aligned}
K_N - \|N\| &= \Delta_H\left\|\ddot{x}^d\right\| + \Delta_C\left\|\dot{x}^d\right\| - \|N\| \\
&\ge \left\|K_H - H(x)\right\|\left\|\ddot{x}^d\right\| + \left\|K_C - C(x,\dot{x})\right\|\left\|\dot{x}^d\right\| - \|N\| \\
&\ge \left\|(K_H - H(x))\ddot{x}^d + (K_C - C(x,\dot{x}))\dot{x}^d\right\| - \|N\| = 0.
\end{aligned} \tag{23}$$

Therefore, $\dot{V} \le 0$, which indicates the system is stable. Since the Lyapunov function V is lower bounded (by zero), V tends to a constant as $t \to \infty$ and is therefore bounded for $t \in [0, \infty]$. This means that $\dot{e}$ and E are bounded as they appear in V, and hence, $\dot{E}$ and e are bounded according to the definition (5). Furthermore, we can conclude that $\ddot{E}$ and $\ddot{e}$ are bounded as well according to the closed-loop dynamics (20) and the expression (5), which consequently implies the uniform continuity of $\dot{e}$ and $\dot{E}$. From Barbalat's lemma, $\dot{e} \to 0$ and $\dot{E} \to 0$ as time $t \to \infty$. Then, the error dynamics of (20) can be expressed as

$$K_PE + N = 0. \tag{24}$$

Since $\dot{x}^d = 0$ at the final desired position, $N=0$ from its definition. We then obtain $E=0$ from (24), and further, $e=0$ and $\varepsilon = 0$ based on (5) and (3). As a result, it exists an invariant set $\Omega = \{(x,\dot{x})\colon \dot{x} = 0, E = 0, e = 0, \varepsilon = 0\}$. From the LaSalle' theorem, it therefore derives the system is asymptotic stability, i.e., $E \to 0$, $e \to 0$ and $\varepsilon \to 0$ as $t \to \infty$ at the final desired position.

5. Simulations

In this section, simulations is first performed to illustrate the motion accuracy of the proposed algorithm in the networked multi-axis motion control system.

5.1. Simulation platform

As shown in Fig. 3, the simulation model is designed based on Matlab/Simulink and the TrueTime toolbox [7]. In the TrueTime toolbox, computer block and network block are adopted. The computer block is based on the event-driven mechanism and executes user-defined tasks, such as sampling and actuation tasks, the control algorithm, the application layer of the communication and the network interface. According to a chosen network model, such as CSMA/CD (Ethernet), and CSMA/AMP (CAN), messages are sent and received for the network block. And Ethernet with 100 Mbps link speed was set as the transmission media of the simulation model. To simulate a real-time network with a ring topology, the lumped frame-based EtherCAT protocol was programmed into the computer block of each networked node.

In the simulation, four axes are used and each axis is actuated by a DC motor and a pulse width modulation (PWM) drive. It should be noted that the four axes are independent. The mathematical model of each axis between the PWM input and the position output can be represented as

$$P(s) = \frac{k}{s(\tau s + 1)}, \tag{25}$$

where the time constants τ (s) for each axis are 0.055 (*Axis 1*),

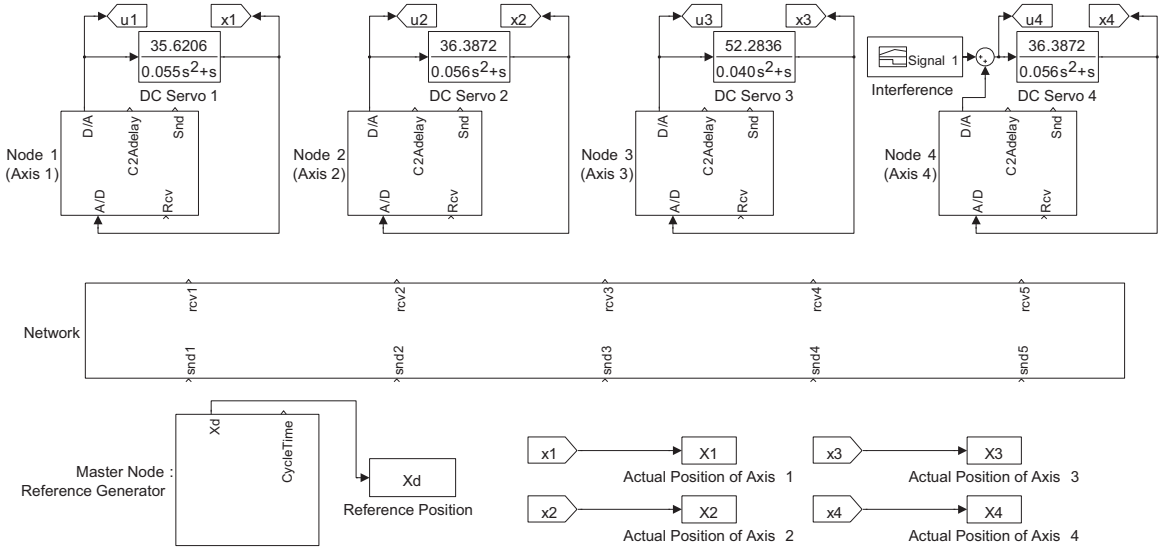


Fig. 3. Simulation model of the networked multi-axis motion control system.

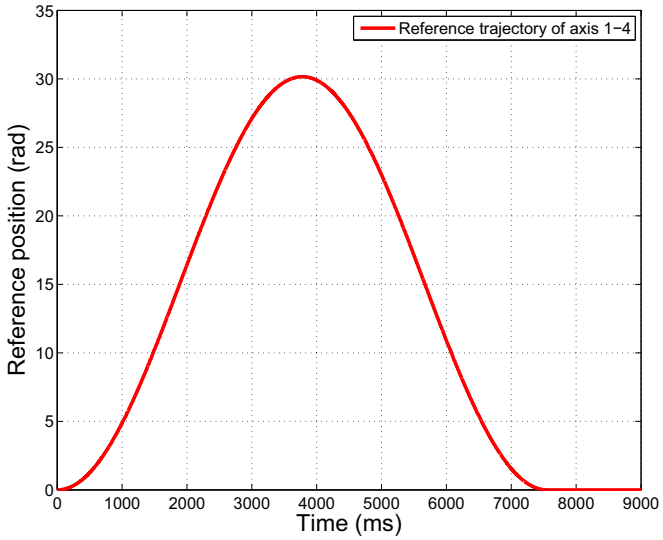


Fig. 4. Reference trajectory in the simulation and the experiment.

0.056 (*Axis 2*), and 0.040 (*Axis 3*), and the overall gains k ($(rad/s)/PWM$) are 35.6206 (*Axis 1*), 36.3872 (*Axis 2*), and 52.2836 (*Axis 3*), respectively [42]. The fourth axis has the same model with the second axis, but adding an input disturbance to test how the proposed algorithm responds to an unexpected disturbances in the motion.

5.2. Simulation results

Simulations are conducted involving a cosine-function motion command to evaluate the performance of the developed algorithm, which can be seen in Fig. 4. The input disturbance in the fourth axis was simulated by adding a positive control input with a magnitude of 0.05 PWM during the time period between 2 and 4 s.

The performance is also compared between the following two controllers: (1) an uncoupled controller with only a standard PD position loop controller, (2) the proposed PD-type synchronous controller for position synchronization in a networked multi-axis motion control system. The control gains in the PD controller are $K_P = diag\{0.6221\}$ (PWM/rad), $K_D = diag\{0.6221\}$ (PWM·ms/rad). For a fair comparative study, K_P and K_D of the synchronous controller are set as the same ones in the PD controller. And the other control gains in the proposed synchronous controller are $\alpha = diag\{1.3\}$, $K_H = diag\{0.001544\}$ $\left(PWM\cdot ms^2/rad\right)$, $K_C = diag\{0.02807\}$ (PWM·ms/rad), $m = 258.6268$ (PWM·ms/rad), $K_N = 10$ (PWM). In simulations, the trial-and-error method was used to obtain all the control gains.

Figs. 5 and 6 show the simulation results. In each figure, the horizontal axis denotes the simulation time and the vertical axis represents the position error or the position synchronization error. The results show that the proposed synchronous controller achieves better synchronization performance in the trajectory tracking, regardless of model errors or unexpected disturbances among distributed axes by comparing with the standard PD control system.

6. Experiments

We further conducted experiments on the networked motion control system to validate the proposed synchronization approach.

6.1. Hardware design of the experimental platform

As shown in Fig. 7, five networked motion control nodes and a four-axis table from Googol Tech Ltd. are used in the established platform. Sensing and actuation, communication capabilities over real-time Ethernet and data processing to accomplish motion control tasks are the key features of the developed networked motion control node. To achieve concurrent and modular designs, both a digital signal processor (DSP) and a field-programmable gate array (FPGA) are adopted in the hardware of each node. The hardware structure of the designed node as shown in Fig. 8 includes the following: an Ethernet physical layer (PHY) module, an FPGA module, a DSP module, and a digital-to-analog converter (DAC) module. In the Ethernet PHY module, a commercial chip is used to implement the physical layer of the real-time communication, and media independent interface (MII) is provided for the media access controller (MAC) in the FPGA module. Both the DSP and FPGA modules are detailed as follows:

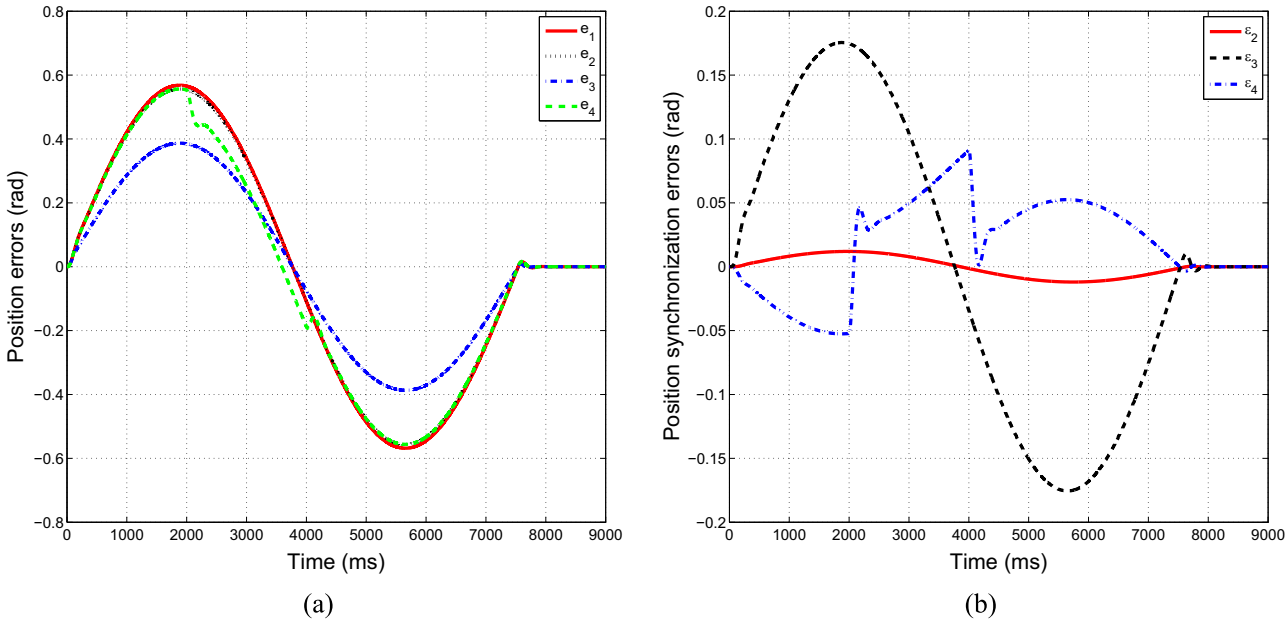


Fig. 5. Results of the standard PD control in the simulation. (a) Position errors. (b) Position synchronization errors.

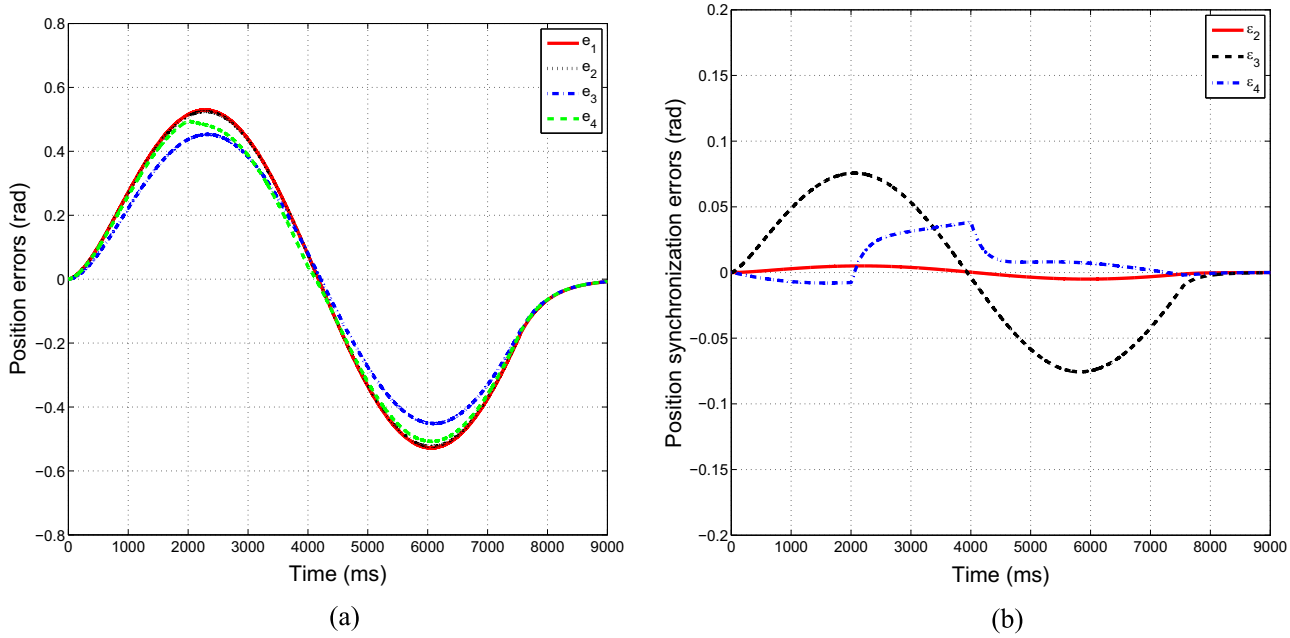


Fig. 6. Results of the synchronous control in the simulation. (a) Position errors. (b) Position synchronization errors.

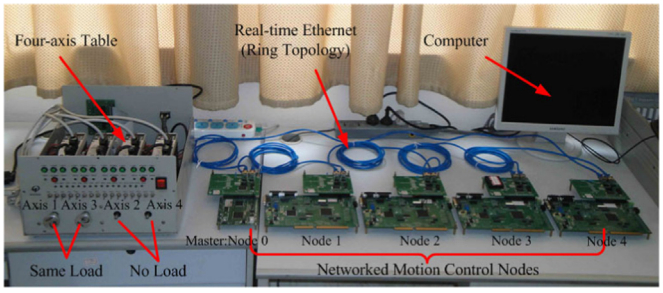


Fig. 7. Experimental set-up, with five networked motion control nodes and a four-axis table.

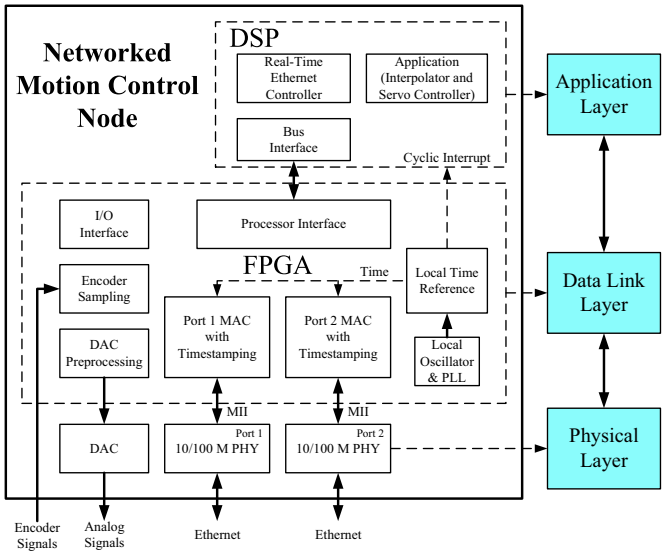


Fig. 8. Hardware structure of the networked motion control node.

6.1.1. *DSP module*

Three main blocks are included in this module. The first one is the real-time Ethernet controller for realizing the network configuration and the application layer of the communication. It is dedicated to real-time communication by implementing one existing RTE communication profile specification of the standard IEC61784-2 [43], e.g., the EtherCAT protocol used in this work. The second block has several application functions, for instance, a sampling filter, an interpolator, and a servo controller. The velocity planning and real-time interpolation are executed in the interpolator. It therefore generates the reference positions for each axis. And the servo controller adopts the decentralized synchronous controller. However, not all application functions are included in a networked motion control node. It depends on the type of the node. As discussed in [16], the interpolator is usually running on the master node, and the slave motion control node consists of the sampling filter and the position synchronous controller under the condition of the decentralized architecture. The third block is the bus interface with the FPGA module using the address mapping technology.

6.1.2. *FPGA module*

The FPGA module provides the following functions and characteristics, such as DAC preprocessing circuit, encoder sampling, input/output (I/O) interface and media access controller (MAC) with timestamping. The DAC preprocessing circuit is designed for the pretreatment of the DAC module, which is used to convert digital signals to analog signals and amplify the analog signals for servo drivers. The encoder sampling is composed of a phase discriminator, a quadruple-frequency circuit and a position counter. The phase discriminator is adopted to judge the rotary direction of the motors. The quadruple-frequency circuit is developed to enhance the orientation accuracy with four times the electrical drive signal. In addition, the position counter is used to count the quadruple encoder signal based on the direction signal. Ethernet ports of the networked motion control node are controlled by "MAC with Timestamping" blocks. To realize the time synchronization, the timestamping approach is used in this work. And a switch is realized in every connected motion control node to support the ring topology in the system. The detailed description of the "MAC with Timestamping" block can be found in [8].

There are two important benefits in the developed experimental platform. First, this platform can easily build multi-axis

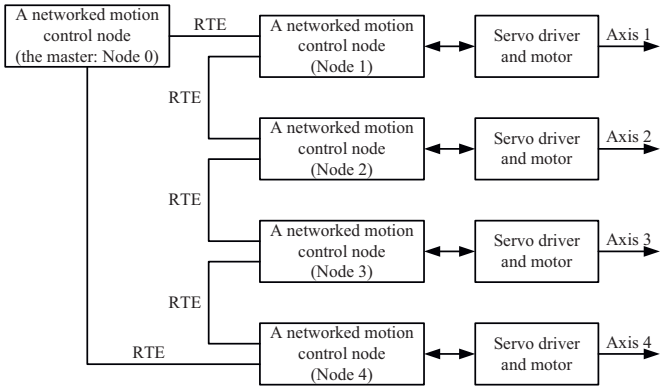


Fig. 9. Architecture of the four-axis networked motion control system.

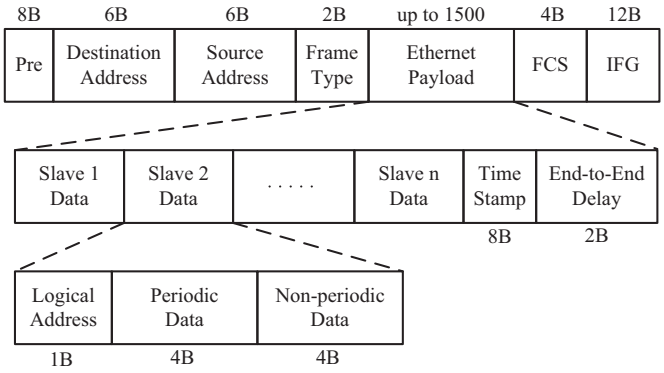


Fig. 10. Frame format of the real-time Ethernet.

motion control systems over different RTE solutions with reduced development cost, only by configuring the related protocol stacks of the application layer and the data link layer on the same hardware implementation. Second, compared with those systems which use directly the existing RTE slave controllers, we can further study the time synchronization methods of the RTE in the designed system due to its openness. As aforementioned, the time synchronization is needed in the presented synchronous control.

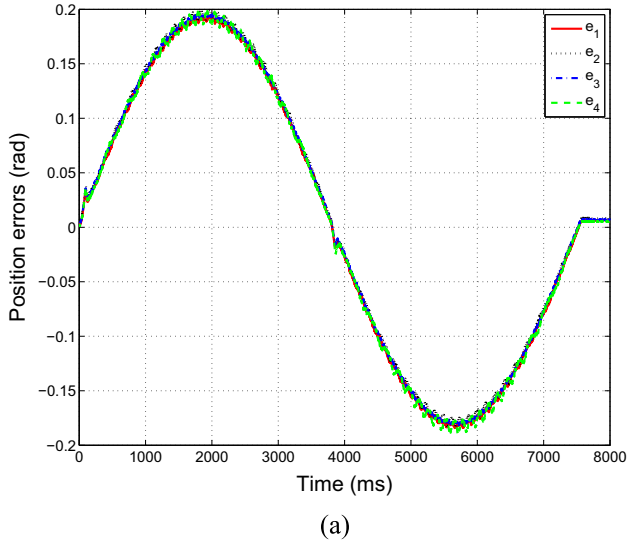
6.2. Implementation of the real-time network

The test system is schematically illustrated in Fig. 9. Standard unshielded twisted pair (UTP) cables with the length up to 3 m were adopted to connect two adjacent nodes with the data rate 100 Mbps of Ethernet. Among them, one networked motion control node was selected as the master. The remainder four nodes controlled four motors respectively. The master node performed trajectory generation, and transmitted the position commands to the slave nodes through the real-time Ethernet every T_s. In the experiment, the sample period $T_s = 1$ ms was set such that the networked multi-axis motion control system was stable.

In the real-time communication network with the EtherCAT protocol, each slave node is allocated a logic address during the network configuration phase. As shown in Fig. 10, the lumped frame was periodically sent by the master node. The overhead of each frame is $12B$, with $8B$ preamble and $4B$ CRC. And the data payload is $48B$. In addition, the time synchronization performance is measured to be only 95.5 ns after four hops [34], which also indicates high-precision synchronous sampling and actuation can be achieved in the experiment.

6.3. Position synchronization performance

Furthermore, the cosine-function desired trajectory was selected to evaluate the synchronous performance of the developed algorithm in the experiment as shown in Fig. 4. The maximum velocity of the trajectory planning was chosen as 753.98 rad/min. Also, without loss of generality, we let the first axis and the third axis have the same load, while make the other two axes with no load. The control gains for the standard PD controller are $K_P = diag\{1.4228\}$ (V/rad), $K_D = diag\{2.8457\}$ $(V{\cdot}\text{ms/rad})$. In order to the compare the performance of the proposed synchronous controller and the standard PD controller, K_P and K_D of the synchronous controller are same as the control gains of the standard PD controller. And the other control gains for the synchronous controller are $\alpha = diag\{0.6\}$, $K_H = diag\{11.0135\}$ $(V{\cdot}\text{ms}^2/\text{rad})$, $K_C = diag\{4.0616\}$ $(V{\cdot}\text{ms/rad})$, $m = 0.02866$ $(V{\cdot}\text{ms/rad})$, $K_N = 0$ (V). All the control gains were also chosen by the trial-and-error method. It should be also noted that the α should be sufficiently small to guarantee the defined Lyapunov function positive for stability proof. However, in applications, the value of α is often tuned by the trial and error method to make the tradeoff of the synchronization performance and the system stability



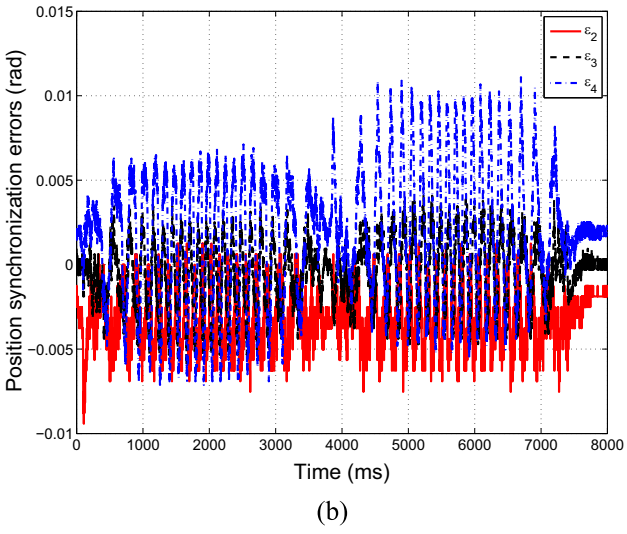


Fig. 11. Results of the standard PD control in the experiment. (a) Position errors. (b) Position synchronization errors.

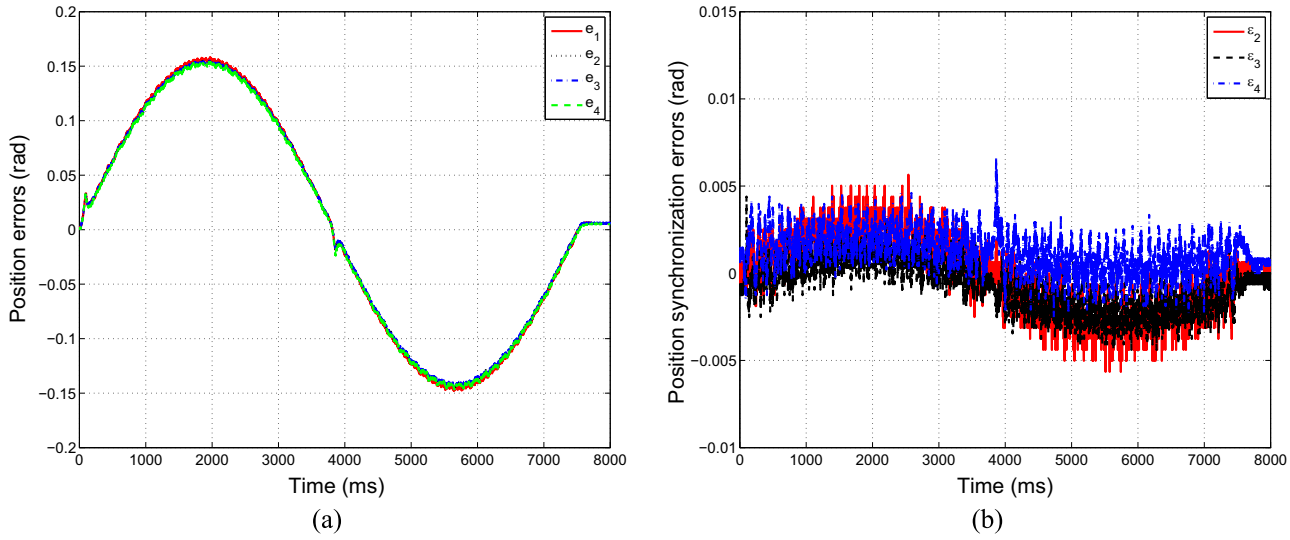


Fig. 12. Results of the synchronous control in the experiment. (a) Position errors. (b) Position synchronization errors.

Table 1
RSMEs of the two controllers.

	Track-RSME/rad	Sync-RSME/rad
Standard PD control	0.25715	0.0059015
Synchronous control	0.20427	0.0032035
RPE	**20.564%**	**45.717%**

simultaneously, which has been discussed and tuned a moderate value with $\alpha = diag\{0.5\}$ in [24]. That is the reason why the values of α were not sufficiently small with respect to 1 in simulations and experiments of this work. In addition, the reasons for using the same K_P and K_D parameters lie in the fact that: (i) in the experimental setup, the hardware of all the slave nodes is equipped equally; (ii) during the system configuration phase in some practical applications, there are many networked nodes and their models are usually considered as the same with minor difference. The field engineer often uses the same control parameters for all distributed nodes to reduce the installation time and costs. Therefore, in this prototype, every PD controller for four slaves is implemented with the same K_P and K_D parameters in simulations and experiments. Figs. 11 and 12 show the experimental results, respectively. In these figures, the horizontal axis denotes the experimental time and the vertical axis represents the value of the position error or the position synchronization error.

As in [44], the root-square mean error (RSME) is used to evaluate the performances of the two controllers. Hence, the tracking RSME (Track-RSME) and the synchronization RSME (Sync-RSME) of the multi-axis motion system are defined as

$$Track - RSME = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(e_1^2(k) + e_2^2(k) + e_3^2(k) + e_4^2(k)\right)}, \tag{26}$$

$$Sync - RSME = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\varepsilon_2^2(k) + \varepsilon_3^2(k) + \varepsilon_4^2(k)\right)}, \tag{27}$$

where $e_i(k)$ and $\varepsilon_i(k)$ represent the position error and the position synchronization error at the kth sampling point of the ith axis, and N is the number of the sample point. The Track-RSME and the Sync-RSME results are shown in Table 1, in which the RPE indicates the reduced percentage of error. The experimental results shown in Table 1 demonstrate that the Track-RSME decreases 20.564% and the Sync-RSME decreases 45.717% using the proposed synchronous control, compared with the standard PD control.

7. Conclusions

In this paper, we propose a decentralized synchronous controller for each motion control node to achieve the position synchronization of real-time network-based systems. Simulations and experiments are conducted to demonstrate the effectiveness of the proposed synchronization control approach in the real-time network. Several distinct features of the work are summarized as follows.

(1) We extend the position synchronization approach to the networked multi-axis motion control system. A decentralized synchronous controller, which takes the message scheduling of the network into account, is designed to synchronize its motion with other nodes. Moreover, in order to reduce the effect of network-induced delays, a motion message estimator is utilized in the synchronous controller to estimate the current position errors of preceding axes. It is proven that the decentralized synchronous controller with the delay compensation is asymptotically stable.

(2) A networked simulation system is developed through the Matlab/Simulink and the TrueTime toolbox, and an experimental platform is built based on smart networked motion control nodes and the real-time Ethernet communication. Then, simulations and experiments are conducted to verify the effectiveness of the proposed synchronization control approach. Both simulation and experimental results demonstrate that the decentralized synchronous controller achieves good position synchronization performance in the networked multi-axis motion control system.

References

[1] Bibinagar N, Kim W. Switched Ethernet-based real-time networked control system with multiple-client-server architecture. IEEE/ASME Trans Mechatron 2013;18(1):104–12.
[2] Hsieh CC, Hsu PL. Analysis and applications of the motion message estimator for network control systems. Asian J Control 2008;10(1):45–54.
[3] Lian F, Yook J, Tilbury D, Moyne J. Network architecture and communication modules for guaranteeing acceptable control and communication performance for networked multi-agent systems. IEEE Trans Ind Inf 2006;2(1):12–24.
[4] Gupta R, Chow M. Networked control system: overview and research trends.

IEEE Trans Ind Electron 2010;57(7):2527–35.
[5] Chow MY, Tipsuwan Y. Gain adaptation of networked DC motor controllers based on QoS variations. IEEE Trans Ind Electron 2003;50:5.
[6] Jeong S, You S. Precise position synchronous control of multi-axis servo system. Mechatronics 2008;18(3):129–40.
[7] Cervin A, Henriksson D, Ohlin M. TRUETIME 2.0 beta-reference manual. Department of Automatic Control, Lund University, 2009.
[8] Xu X, Xiong Z, Wu J, Zhu X. Development of a networked multi-agent system based on real-time Ethernet. Intell Robot Appl 2011:356–65.
[9] Barranco M, Proenza J, Almeida L. Quantitative comparison of the error-containment capabilities of a bus and a star topology in CAN networks. IEEE Trans Ind Electron 2011;58(3):802–13.
[10] Kjellsson J, Vallestad A, Steigmann R, Dzung D. Integration of a wireless I/O interface for PROFIBUS and PROFINET for factory automation. IEEE Trans Ind Electron 2009;56(10):4279–87.
[11] Song K, Choi G. Fieldbus based distributed servo control using LonWorks/IP gateway/web servers. Mechatronics 2010;20(3):415–23.
[12] Cena G, Demartini C, Valenzano A. On the performances of two popular fieldbuses. In: Proceedings of IEEE international workshop on factory communication systems. IEEE; 2002. p. 177–86.
[13] Felser M. Real-time ethernet-industry prospective. Proc IEEE 2005;93(6):1118–29.
[14] Jasperneite J, Imtiaz J, Schumacher M, Weber K. A proposal for a generic real-time ethernet system. IEEE Trans Ind Inform 2009;5(2):75–85.
[15] Zhang W, Branicky MS, Phillips SM. Stability of networked control systems. IEEE Control Syst Mag 2001;21(1):84–99.
[16] Yook J, Tilbury D, Soparkar N. A design methodology for distributed control systems to optimize performance in the presence of time delays. Int J Control 2001;74(1):58–76.
[17] Lian FL, Moyne J, Tilbury D. Modelling and optimal controller design of networked control systems with multiple delays. Int J Control 2003;76(6):591–606.
[18] Lai CL, Hsu PL. Design the remote control system with the time-delay estimator and the adaptive Smith predictor. IEEE Trans Ind Inform 2010;6(1):73–80.
[19] Montestruque L, Antsaklis P. On the model-based control of networked systems∗ 1. Automatica 2003;39(10):1837–43.
[20] Hu W, Liu G, Rees D. Event-driven networked predictive control. IEEE Trans Ind Electron 2007;54(3):1603–13.
[21] Natori K, Tsuji T, Ohnishi K, Hace A, Jezernik K. Time-delay compensation by communication disturbance observer for bilateral teleoperation under time-varying delay. IEEE Trans Ind Electron 2010;57(3):1050–62.
[22] Koren Y. Cross-coupled biaxial computer control for manufacturing systems. J Dyn Syst Meas Control 1980;102:265–72.
[23] Shih YT, Chen CS, Lee AC. A novel cross-coupling control design for bi-axis motion. Int J Mach Tools Manuf 2002;42(14):1539–48.
[24] Sun D, Shao X, Feng G. A model-free cross-coupled control for position synchronization of multi-axis motions: theory and experiments. IEEE Trans Control Syst Technol 2007;15(2):306–14.
[25] Sun D, Tong M. A synchronization approach for the minimization of contouring errors of CNC machine tools. IEEE Trans Autom Sci Eng 2009;6(4):720–9.
[26] Yang J, Li Z. A novel contour error estimation for position loop-based cross-coupled control. IEEE/ASME Trans Mechatron 2011;16(4):643–55.
[27] Yao B, Hu C, Wang Q. An orthogonal global task coordinate frame for contouring control of biaxial systems. IEEE/ASME Trans Mechatron 2012;17(4):622–34.
[28] Karimi M, Jahanpour J, Ilbeigi S. A novel scheme for flexible nurbs-based c2 ph spline curve contour following task using neural network. Int J Precis Eng Manuf 2014;15(12):2659–72.
[29] Li C, Yao B, Zhu X, Wang Q. Adaptive robust synchronous control with dynamic thrust allocation of dual drive gantry stage. In: Proceedings of the IEEE/ASME international conference on advanced intelligent mechatronics; 2014. p. 316–21.
[30] Nuño E, Ortega R, Basañez L, Hill D. Synchronization of networks of nonidentical Euler–Lagrange systems with uncertain parameters and communication delays. IEEE Trans Autom Control 2011;56(4):935–41.
[31] Xu X, Sheng X, Xiong Z, Zhu X. Time-stamped cross-coupled control in networked CNC systems. In: Proceedings of the IEEE international conference on robotics and automation; 2011. p. 4378–83.
[32] Albert M, Open-architecture. CNC close servo loop in software. Mod Mach Shop 1997;7.
[33] Shi D, Wang B, Wu H. A novel design of distributed servo system based on rearranging functions between host controller and servo driver. In: Proceedings of the IEEE international conference on automation and logistics. IEEE; 2007. p. 480–84.
[34] Xu X, Xiong Z, Wu J, Zhu X. High-precision time synchronization in real-time Ethernet-based CNC systems. Int J Adv Manuf Technol 2013;65(5–8):1157–70.
[35] Eidson J. Measurement, control, and communication using IEEE 1588.Inc, New York: Springer-Verlag; 2006.
[36] Harris K. An application of IEEE 1588 to industrial automation. In: Proceedings of the IEEE international symposium on precision clock synchronization for measurement, control and communication. IEEE; 2008. p. 71–6.
[37] Kim K, Sung M, Jin H. Design and implementation of a delay-guaranteed motor drive for precision motion control. IEEE Trans Ind Inform 2012;8(2):351–65.
[38] Sun D, Ge S. Synchronization and control of multiagent systems. CRC Press; 2010.
[39] Prytz G. A performance analysis of EtherCAT and PROFINET IRT. In: Proceedings of the IEEE international conference on emerging technologies and factory automation. IEEE; 2008. p. 408–15.
[40] Sun D, Wang C, Shang W, Feng G. A synchronization approach to trajectory tracking of multiple mobile robots while maintaining time-varying formations. IEEE Trans Robot 2009;25(5):1074–86.
[41] Gu G, Zhu L, Xiong Z, Ding H. Design of a distributed multi-axis motion control system using the IEEE-1394 bus. IEEE Trans Ind Electron 2010;57(12):4209–18.
[42] Lian FL, Moyne J, Tilbury D. Control performance study of a networked machining cell. In: Proceedings of the American control conference. 2000; vol. 4. p. 2337–41.
[43] IEC 61784-2 Ed. 1, Industrial communication networks-profiles-part 2: additional fieldbus profiles for real-time networks based on ISO/IEC 8802–3. IEC; 2007.
[44] Shang W, Cong S, Zhang Y, Liang Y. Active joint synchronization control for a 2-DOF redundantly actuated parallel manipulator. IEEE Trans Control Syst Technol 2009;17(2):416–23.