A Servo-Based Approach for Adaptive Synchronization of Wheeled Manipulators in the Complicated Tasks

Journal:	*Science Progress*
Manuscript ID	SCI-25-1443
Manuscript Type:	Original Research Article
Date Submitted by the Author:	30-Jun-2025
Complete List of Authors:	Nguyen, Hung; HUTECH University Nguyen, Thanh Phuong; HUTECH University Nguyen, Song Hung; VNUHCM-Ho Chi Minh City University of Technology Ngo, Ha Quang Thinh; VNUHCM-Ho Chi Minh City University of Technology,
Keywords:	Wheeled manipulator, real-time control, embedded system, motion control, synchronization
Abstract:	In the field of wheeled manipulator (WM), the mobile platform plays a crucial role in ensuring precise and efficient manipulation. In previous works, there lacks of in-depth researches to concern the real-time performance of wheeled base in WM. Therefore, a novel concept to synchronize the motion of omnidirectional wheels is proposed by using the servo-based real-time express protocol. In this design, all-wheel drive (AWD) or four-wheel differential drive (FWDD) is empowered by four networked-servo motors. To manipulate them, both hardware design and software program of our motion controller which consists of four modules such as microcontroller unit (MCU), real-time protocol module, physical layer module, and pulse transform module, are presented. To validate the efficiency and feasibility of our method, the real-world platform of wheeled mobile robot and practical hardware of the proposed motion controller are launched. From these results, it could be seen clearly that the proposed approach is proper and effective for the synchronous control in the industrial WM system.

Hung Nguyen [1], Thanh Phuong Nguyen [1], Song Hung Nguyen [2,3] and Ha Quang Thinh Ngo [2,3,*]

[1] HUTECH Institute of Engineering, HUTECH University, Ho Chi Minh City, Vietnam
[2] Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, District 10, HCMC, 700000, Vietnam
[3] Vietnam National University Ho Chi Minh City (VNU-HCM), Linh Trung Ward, Thu Duc District, Ho Chi Minh City, 700000, Vietnam
* Correspondence: nhqthinh@hcmut.edu.vn

A Servo-Based Approach for Adaptive Synchronization of Wheeled Manipulators in the Complicated Tasks

ABSTRACT

In the field of wheeled manipulator (WM), the mobile platform plays a crucial role in ensuring precise and efficient manipulation. In previous works, there lacks of in-depth researches to concern the real-time performance of wheeled base in WM. Therefore, a novel concept to synchronize the motion of omnidirectional wheels is proposed by using the servo-based real-time express protocol. In this design, all-wheel drive (AWD) or four-wheel differential drive (FWDD) is empowered by four networked-servo motors. To manipulate them, both hardware design and software program of our motion controller which consists of four modules such as microcontroller unit (MCU), real-time protocol module, physical layer module, and pulse transform module, are presented. To validate the efficiency and feasibility of our method, the real-world platform of wheeled mobile robot and practical hardware of the proposed motion controller are launched. From these results, it could be seen clearly that the proposed approach is proper and effective for the synchronous control in the industrial WM system.

Keywords: Wheeled manipulator, real-time control, synchronization, embedded system, motion control.

1 INTRODUCTION

The era of Industry 4.0 requires robotic system to be more flexible, reconfigurable, smart mobility and manipulation in the complicated environment such as closing to humans. Early deployments proposed the integrated solutions with customization for specific tasks [1-3]. Wheeled manipulator offers an innovative evolution in the architectures of robotic system which merge the flexibility of mobile platform and the articulated capability of robot arm. WM has become popular in the large numbers of applications, i.e. industry [4, 5], household [6], manufacturing [7, 8], outdoor environment [9-11] or rescue operation [12]. Owing to the merger of mobility and manipulation, handling tasks of WM cover relatively broad categories from inspection [13], pick-and-place [14], transportation [15], assembly [16], and more. Eventually, the convergence of its applications across various tasks in autonomous, semi-autonomous, and full tele-operation of contexts gives rise to a seemingly infinite diversity of intelligent mobile manipulation.

In most of cases, WM requires a cyber-physical or articulated robotic system to achieve smart direct/non-direct interaction with the environment. Owing to the timescales involved, this intelligence must rely on different technologies and frameworks, which can be categorized into timing control (offline, deliberative), methodology (distributed, centralized), and protocol (wireless, cable).

In general, wheeled mobile manipulator could be classified into a grounded-robotic base with one or more mounted industrial manipulators [17-19]. It inherits many regular advantages, for instance extended workspace (conventional, dexterous) [20, 21], reconfigurable transform [22], high disturbance-rejection abilities and robustness [23, 24]. To gain these specifications, wheeled platform serves as the key component since it must be manipulated firstly in most of handling tasks. There is a need to synchronize the motion of robotic wheels by not only programming control but also hardware level.

Table 1. Review of the cutting-edge specifications in related researches

Category	Application	Author(s)	Publication year	Description	Hardware	Software	Communication Protocol	Limitation(s)
	Transportation	Motion planning Bahadir P. et al [17]	2020	A control framework to autonomously handle and reposition pallet jacks in unstructured environments was proposed	A Robotnik SUMMIT-XL STEEL mobile platform, a Franka Emika Panda robotic arm, ASUS Xtion Pro RGB-D camera, HOKUYO-10LX Laser	ROS, Matlab	Wireless	+ Unable to deal with uncertainties + Lack of verifying the feasibility of maneuvers in narrow environment
	Assembly	Li, F. et al [18]	2019	A deep reinforcement learning method without prior knowledge for teaching assembly skills was effectively invented for handling uncertainties	KUKA iiwa robot, air pump and workbench, circuit breaker HYB1-63, NVIDIA GTX1070, digital camera	Tensorflow	Ethernet	+ Low success rate and large training time
	Warehouse management	Nguyen, T. P. et al [20]	2024	The motion controller for a WM was presented to achieve over 90% success in robotic manipulation tasks	DDR mobile base, robotic arm with parallelogram mechanism	SolidWorks, Matlab,Visual C/C++	USB	+ Slow response time, burden computation and unexpected vibration
	Additive manufacturing	Path planning Brahmi M. et al [22]	2020	A novel approach for building accurate thin shell parts using supportless extrusion-based additive manufacturing was developed	Two 6 DOF manipulators, 3 DOF build-platform and 3 DOF extrusion tool	AutoCAD, ABB RobotStudio	Ethernet	+ Unable to build certain asymmetric geometries, such as an inverted Y-shaped hollow tube or a part with a handle
	Disinfection	Nguyen, T. P. et al [23]	2023	Investigators presented a navigation framework for a wheeled robot equipped with advanced sensors for autonomous disinfection	Lidar UTM-30LX, Kinect digital camera, DDR wheeled platform	ROS, Gazebo, Matlab	RS-232	+ Limited flexibility due to lack of manipulator + Difficult to work in unknown environment
	Agriculture	Xiong, Y. et al [25]	2020	An autonomous strawberry-picking robot with an advanced obstacle-separation algorithm and dual-arm system, achieving high success rates in harvesting tests but	RGB-D camera, a single-rail dual-arm manipulator, two grippers, and a mobile platform	ROS, Arduino IDE	CANbus	+ Limited success with fully surrounded targets + Insufficient gripper dexterity, limited adaptability to different crop types
	Household	Robotic Packaging Park, D. et al [26]	2020	This study discussed and evaluated a novel meal-assistance system which enables individuals with motor impairments to eat independently	Willow Garage PR2 robot, two cameras, tablet, feeding tool, force/torque sensor, current sensor, tactile sensor, micro-phone	Web-based GUI, ROS	Web socket, JSON	+ Potential psychological impact + Handling of non-anomalous scenarios
	Treatment	Lin, T. C. et al [27]	2022	Researchers evaluated and enhanced teleoperation interfaces for tele-nursing robots	TRINA robot, Stylus device, four cameras (ELP-USBFHD01M 180◦ fisheye, two Intel RealSense D435, Microsoft Kinect 2)	Vicon Nexus motion capture, NASA-TLX	USB	+ Not fully address all sources of physical fatigue + Depending on different tasks or user conditions
	Public	Yamamoto, T. et al [28]	2019	HSR as a standardized platform to enhance global collaboration in robotics, detailing its development, technical features, and effectiveness was studied	3 DoF omni-directional mobile base, 4 DoF robotic arm, and 1 DoF cylindrical torso lift, head 3D sensor, head stereo camera	ROS	RS-485	+ Potential vibration when it accelerates suddenly + Timing cost owing to sequential actions

Human Support Robot (HSR), Robotic Operating System (ROS), Degree-of-Freedom (DoF), Universal Serial Bus (USB)

2 PREVIOUS STUDIES

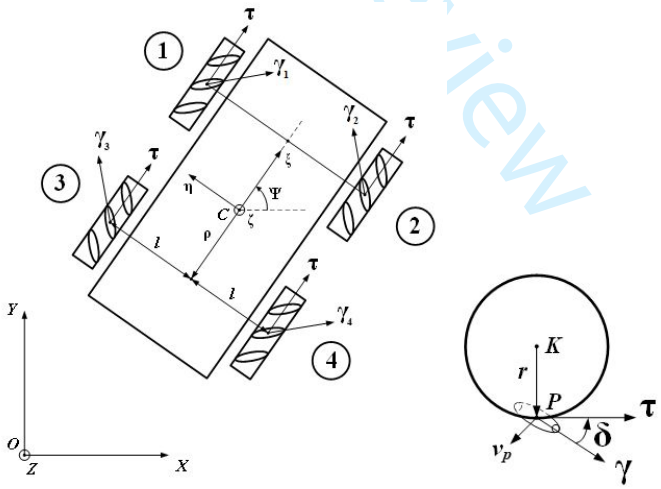
Ethernet technique has played a vital rule in the socio-economic context of humanity. It was evidenced to be quite fundamental and successful in satisfying the communication needs of numerous applications such as video conferencing system [8], traffic [9], vehicular autonomy [10], food safety [11] or industrial Internet of Things (IoT) [12]. From these fields, the technology of web browsing, audio/video streaming or sensing data from IoT devices are energized to develop. However, not all ethernet-based applications match with the requirements of low latency and transmission delay. Especially, in the field of robotics, synchronous control of multi-agent system significantly depends on the real-time communication.

To summarize the state-of-the-art researches in this domains, Table 1 represents the classifications and its applications of related techniques. Basically, motion control, path planning, and robotic interaction are three aspects which attract a lot of attentions from various scholars and practitioners. In each type, content of research, hardware platform, software program, and communication protocol are briefly mentioned.

3 PRELIMINARIES

3.1 SYSTEM MODEL

In Fig. 1a, our robotic platform comprises four omnidirectional wheels or mecanum wheels which are driven independently by four servo motors. This type of wheel is moved by the rotation of rollers on its outer rim. It is considered that the angle δ between axis of each roller and plane of wheel as Fig. 1b has a range from 0^o to 90^o. Then, this angle δ is equal to 45^o in our design. Mecanum wheel is modeled as a thin disk with a radius R, the velocity $\boldsymbol{V}_{\mathrm{P}}$ of the point of contact P of the disk with the supporting plane is orthogonal to the axis of the roller.



(a) (b)

Fig. 1. Modeling of our wheeled platform (a) and its omnidirectional wheel (b).

Table 2. List of the mathematical symbols in our model.

Symbol	Description	Symbol	Description
$m0$	Mass of platform.	ω_i	Angular velocity of the i^{th} robotic wheel
A	Geometry center of Robot.	φ_i	Rotational angle of the i^{th} robotic wheel

C	Center of mass of Robot	m_i	Mass of Robot wheel.
ρ	Distance from A point to the axis of the wheel.	ψ	Angle formed by the longitudinal axis of symmetry of the body with axis OX.
$2l$	Coaxial distance between couples Mecanum wheel.	Axyz	Robot-attached coordinate system Axyz with origin at the geometry center of the cart
OXYZ	Fixed coordinate system.		

Where γ is the unit vector of the axial roller. List of mathematical symbols in our model is described in Table 1. We assume that there is no slipping phenomenon when robotic wheels move [29], we have

$$v_P \cdot \gamma = 0 \quad (1)$$

v_k is the velocity of center point K in wheel, then

$$v_p = v_k + \omega \times r \quad (2)$$

Since r = $\overrightarrow{KP}$, equation (1) can be represented as

$$(v_k - r\omega\tau) \cdot \gamma = 0 \quad (3)$$

3.2 KINEMATIC CONSTRAINTS

Let V_c be the velocity at the mobile robot's center of mass and $R_i = \overrightarrow{CK_i}$, then:

$$V_{Ki} = V_c + \Omega \times R_i (4)$$

V_{Ki} is the velocity of each wheel and Ω is the angular velocity of the mass of the body robot. So, the kinematics of the system can be represented by:

$$(v_A + \Omega \times R_i) \cdot \gamma_i = \frac{r}{\sqrt{2}} \omega_i (5)$$

Introduce a robot-attached coordinate system $Axyz$ with origin at the geometry center of the cart. We point axis Ax along the longitudinal symmetry axis of the cart, axis Ay along the lateral symmetry axis, and axis Az vertically upward. Denoted by V_{Ax} and V_{Ay} the projections of the velocity of the center of mass onto the movable axes Ax and Ay, respectively, and represent expression (5) as follows:

$$v_{Ax}\gamma_{ix} + v_{Ay}\gamma_{iy} + (r_{ix}\gamma_{iy} - r_{iy}\gamma_{ix})\Omega = \frac{r}{\sqrt{2}} \omega_i (6)$$

And we have:

$$\gamma_{1x} = \gamma_{4x} = \frac{1}{\sqrt{2}},$$
$$\gamma_{2x} = \gamma_{3x} = \frac{1}{\sqrt{2}},$$
$$r_{1x} = r_{2x} = \rho,$$
$$r_{3x} = r_{4x} = -\rho,$$
$$\gamma_{1y} = \gamma_{4y} = \frac{1}{\sqrt{2}} (7)$$
$$\gamma_{2y} = \gamma_{3y} = \frac{1}{\sqrt{2}},$$
$$r_{1y} = r_{2y} = l,$$
$$r_{3y} = r_{4y} = -l$$

So (6) become:

$$\begin{cases} v_{Ax} - v_{Ay} - (\rho + l)\Omega = r\omega_1 \\ v_{Ax} + v_{Ay} + (\rho + l)\Omega = r\omega_2 \\ v_{Ax} + v_{Ay} - (\rho + l)\Omega = r\omega_3 \\ v_{Ax} - v_{Ay} + (\rho + l)\Omega = r\omega_4 \end{cases} (8)$$

(8) can be written as a forward kinematic equation:

$$\begin{bmatrix} v_{Ax} \\ v_{Ay} \\ \Omega \end{bmatrix} = \begin{bmatrix} \frac{R}{4} & \frac{R}{4} & \frac{R}{4} & \frac{R}{4} \\ -\frac{R}{4} & \frac{R}{4} & \frac{R}{4} & -\frac{R}{4} \\ -\frac{R}{4(\rho + l)} & \frac{R}{4(\rho + l)} & -\frac{R}{4(\rho + l)} & \frac{R}{4(\rho + l)} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} (9)$$

Transform the velocity of the center from local coordinate $Axyz$ to global coordinate $OXYZ$:

$$\begin{bmatrix} V_{Ax} \\ V_{Ay} \end{bmatrix} = \begin{bmatrix} cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} v_{Ax} \\ v_{Ay} \end{bmatrix} (10)$$

3.3 NETWORK-BASED METHOD

In the previous solutions, most of robotic systems are launched as standalone platform that operates independently without needing support or connection to other systems or devices. Data exchange in such system is regularly serial or chained communication. It causes some disadvantages, for example difficult to share data, higher maintenance cost or limited collaboration. To conquer these challenges, the network-based robotic system which utilizes the distributed servo-driven actuators, becomes one of the most advanced candidates. Presently, there are several alternative industrial networks for our system. However, some of them are not proper for instance EtherCAT-based technique which requires high license cost and logic programming language, has been embedded into the platform of automated guided vehicle [30]. This network does not support to embed the advanced algorithms. In the other effort [31], developers suggested the implementation of DeviceNet protocol in the robotic cell. Nevertheless, their protocols did not play a key role in data exchange among actuators or components.

Inspiring the trendy development in this domain, the network-based design for mobile base should be considered in this research. Our contributions are (a) to introduce a novel concept of motion controller for AWD-enabled mobile platform using the real-time servo-based protocol, (b) present both hardware and software design of the proposed approach, (c) launch the experimental system and achieve several practical results in different test scenarios to verify the effectiveness and feasibility of our approach.

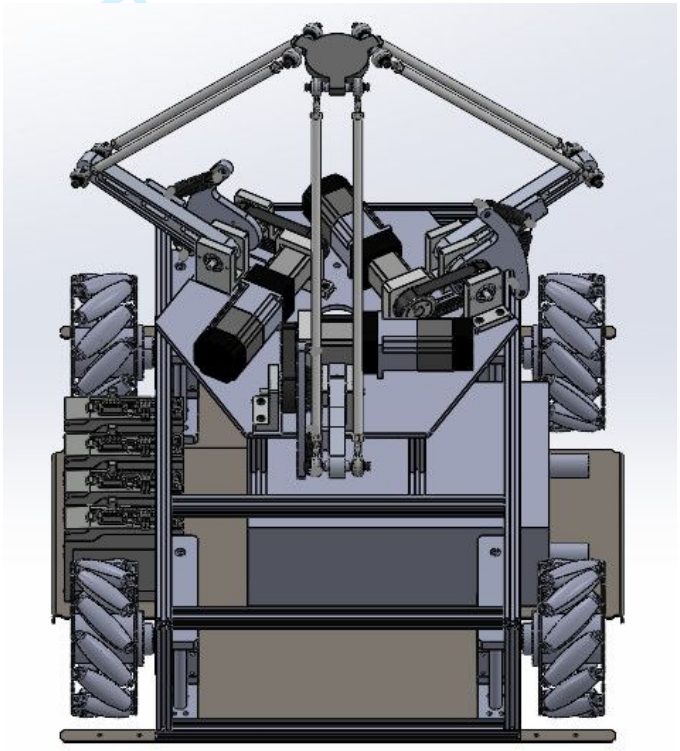
4 THE PROPOSED APPROACH

The content of this section is organized as follows. Firstly, the conceptual design of robotic platform including four omnidirectional wheels, four gearboxes, servo-based driving mechanism, is depicted. Secondly, hardware structure of our motion controller and its functional blocks are illustrated. Some system analysis is conducted to established the theoretical constraints and desired performance.

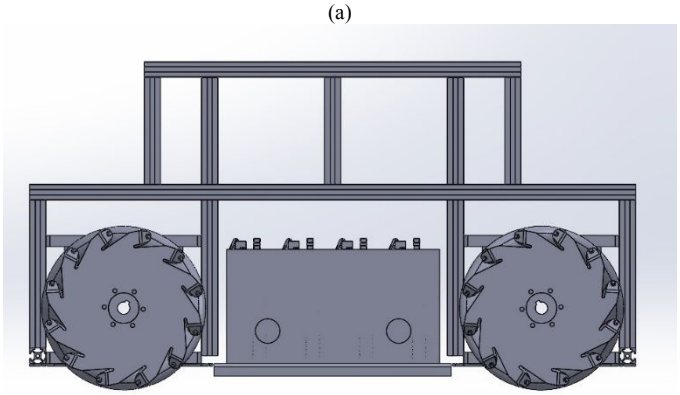
4.1 CONCEPTUAL DESIGN

Generally, our WM as Fig. 2a can be classified into two parts such as mobile base and robot arm which is situated in the upper frame of wheeled platform. The architecture of robotic manipulator is parallel and supports payload manipulation in three-dimensional space. With this design, it allows to distribute the forces on the payload to avoid stress concentrations. Nonetheless, we focus on the synchronous control of mobile platform to guarantee the real-time performance. Hence, omnidirectional wheeled base becomes our target to study.

Consequently, four driving mechanisms are located symmetrically in the wheeled platform. For each one, network-based driver, servo motor, mechanical gearbox and omnidirectional wheel are connected sequentially. Mobile base is made of aluminium, with the upper frame made of steel to bear the load. The wheels consist of a rubber part that contacts the ground, while the wheel frame is made of aluminium. Since these motors are driven by alternative current (AC), it is necessary to deploy an inverter to convert direct current (DC) to AC. Besides, battery packs, servo drivers and other peripheral devices are uniformly distributed to certify the balanced status when robot moves.



(a)



(b)

Fig. 2. 3D design model of our WM (a) and its side view of mobile platform (b).

4.2 HARDWARE IMPLEMENTATION

In Fig. 3, the proposed motion controller contains microcontroller unit (MCU), real-time protocol part, physical layer part, and pulse transform part. This circuit is specified to include two source integrated chips (ICs) with the same ground (GND), one for MNM1221 and KSZ8041, one for PIC24F, in order to facilitate for writing firmware. The PIC24F circuit part is designed based on the endorsed minimum connections circuit of the datasheet along with the development kit for PIC24F. Port D is configured as its data bus, while port A is utilized as address bus along with the XSYNC and XINTRX pins. In addition, this microprocessor comes with a variety of built-in peripherals, high performance in mathematical computation and low power consumption.

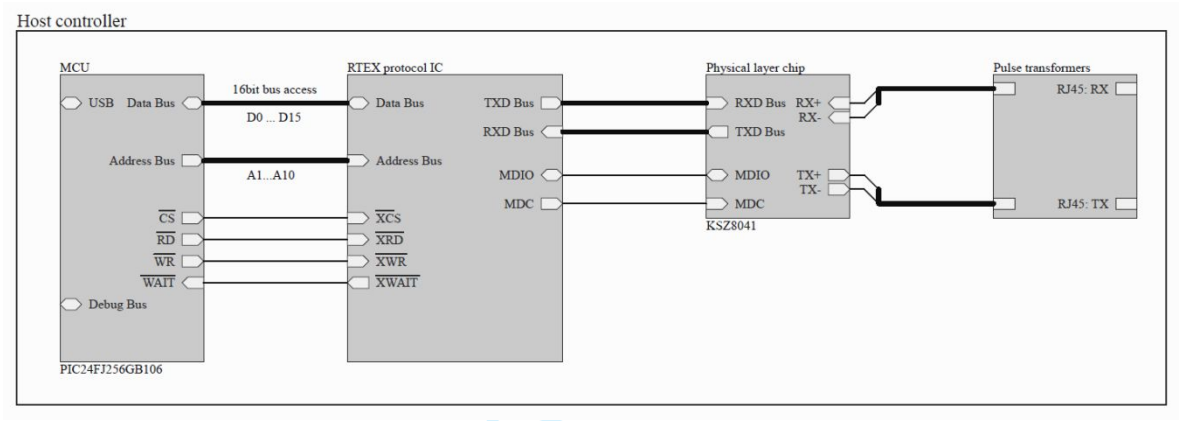


Fig. 3. Overall diagram for connections between MCU, MNM1221 and KSZ8041.

On account of this diagram, the core of motion controller to encrypt and de-encrypt the communication frame is a real-time protocol module which offers a high-speed synchronous motion network. This protocol brings 100 Mbps high speed which is ten times greater than the existing specification, while the system cost could be maintained low. Usually, there are two banks of cache memory for handling data, TX bank to transmit and RX bank to receive. Because of this specification, assignment of two banks is switched alternatively. To avoid any data conflict, one preserved bank is specified to store data. For handling the transmission interface, KSZ8041 delivers a solution for single-chip physical layer transceiver. It provides 10Base-T/100Base-TX physical layer transceiver which supplies various clock modes to transmit and receive data. Furthermore, it utilizes a unique mixed-signal design to extend signaling distance while reducing power consumption. In the last stage, a pulse transformer is an electric component utilized in power systems to isolate circuits that are physically separated or to modify circuit properties like voltage or current ratings. These transformers are ideal for various industrial applications due to their compact design, reliability, and ability to manage high-power pulses effectively.

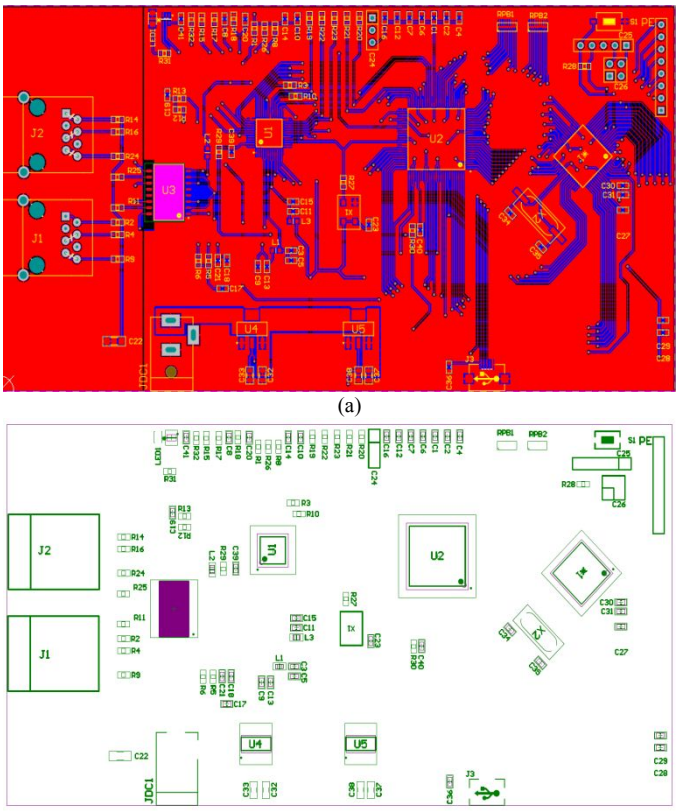


Fig. 4. Design of the proposed control board, (a) PCB layout and (b) location of parts.

To evaluate the real-world performance of our method, a printed circuit board (PCB) with dimensions of 150x80 mm has been developed, as shown in Fig. 4a. This circuit prioritizes providing sufficient space for routing according to anti-interference standards rather than minimizing size. Specifically, junctions are designed with one 90-degree angle and two 135-degree angles to prevent GND overlays, with any space beneath the clock wire left blank. Additionally, the area beneath the pulse transformer is kept clear. The placement of components on this circuit is detailed in Fig. 4b for clarity.

4.3 FIRMWARE IMPLEMENTATION

In general, there are five states to handle this protocol such as initial, ring config, ready, running and reset. Each state has separated functions which characterize its role and working status. In the initial state, our firmware would set related registers and prepare its configurations. Later, it searches for the information about each slave station, and configures the potential operation. After completing this process, the firmware is waiting for ready state. Consequently, it checks the slave information, for instance the sum of slaves, MAC-IDs, and validate them. Then, our network would be driven to running state. Using the TX and RX cache memory, data which consist of command code or motion parameters, is exchanged during the cyclic period. Whenever reset status is activated, it would become initial state. Besides, several constraints among states are established to strengthen its relations.

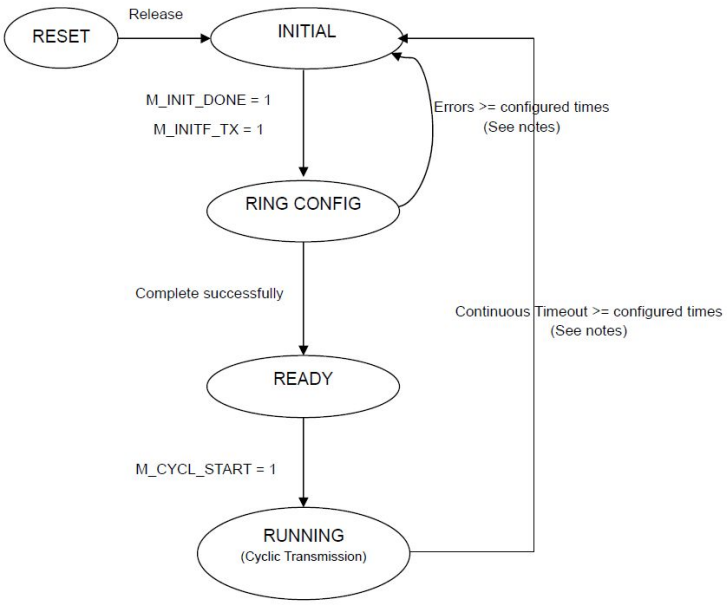


Fig. 5. Flowchart of state machine for our approach.

4.4 THEORETICAL ANALYSIS

To imitate the real-world applications, there are six stations including one master, one reference slave and four servo slaves in our system. Whilst the proposed motion controller is considered as a master station, both reference slave and servo slaves are connected in the ring topology. In reference slave, digital or analog input/output (I/O) signals from sensors are collected. Also, each servo motor is driven as one slave station. According to this network, it is essential to describe the detailed computations of timing transmission. Three types of measurements include transmission between master station M and reference slave S0, between master station M and servo slaves Sn (n=1…4), between reference slave S0 and servo slaves Sn (n=1…4), and among servo slaves Sn (n=1…4).

To clarify the relationship between timing slices, and to specify the notation of the real-time characteristics as Fig. 6, several time sequences are defined as following, t_m: forwarding time just before the master sends the command to TX cache, t_{s0}: execution time when receiving the feedback from reference slave in RX cache, and t_{s1} to t_{s4}: propagation time when servo completes the traveling to target.

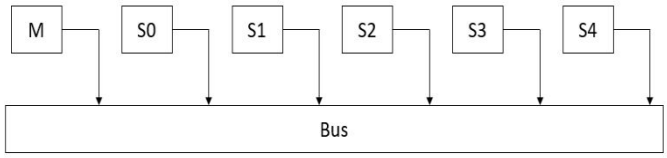


Fig. 6. Illustration of timing relations in the typical network. M: Master, S0: Reference Slave, S1-4: Slave Servo.

Later, some errors of timing transmission in each station could be estimated as

$$d_n^m = t_m - t_{sn} \quad n = 0, 1, 2, 3, 4 \tag{11}$$

$$d_n^{s0} = t_{s0} - t_{sn} \quad n = 1,2,3,4 \tag{12}$$

$$d_n^{s1} = t_{s1} - t_{sn} \quad n = 2,3,4 \tag{13}$$

where,

d_n^m: propagation delay when master start to transmit data until servo reaches to target.

d_n^{s0}: synchronization error between the n^{th} servo and reference slave.

d_n^{s1}: synchronization error between the n^{th} servo and the 1^{st} servo.

According to Fig. 7, if d_n^m, d_n^{s0},d_n^{s1} are negative, then the synchronous signal is later than the reference signal. Otherwise, it is earlier than the reference signal. The synchronization errors in each case of servo slaves are recorded as Table 3. It is considered that with the setting period T_{set}, master station handles the motion command to target during the traveling period T_{run}. Since the level of feedback signal is high for a period T_{set} and low for a period T_{run}, it requires to notice the time of falling edge t_m which command code (0x17) is activated to transmit via TX pin to servo motor.

Table 3. List of the synchronization errors in respect to each station.

	S0	S1	S2	S3	S4
M	$d_0^m = t_m - t_{s0}$	$d_1^m = t_m - t_{s1}$	$d_2^m = t_m - t_{s2}$	$d_3^m = t_m - t_{s3}$	$d_4^m = t_m - t_{s4}$
S0		$d_1^{s0} = t_{s0} - t_{s1}$	$d_2^{s0} = t_{s0} - t_{s2}$	$d_3^{s0} = t_{s0} - t_{s3}$	$d_4^{s0} = t_{s0} - t_{s4}$
S1			$d_2^{s1} = t_{s1} - t_{s2}$	$d_3^{s1} = t_{s1} - t_{s3}$	$d_4^{s1} = t_{s1} - t_{s4}$

The variable $s_0(k)$ contain a bit of the same value as the response signal of master. When the reference slave receives bit $s_0(k)$ from RX pin, it would output a signal for the data logger to record the time at the rising and falling edges. Like master station, it is only noticed about the falling edge of signal.

In the other hand, the variable $x_n^d(k)$ comprises value of new position for servo motor. When this variable is received during a period T_{set}, servo driver could reset the output signal of In Position to high level (low active). Then it would wait the command from master in next period T_{run}. Similarly, time sequence $t_{s1\to4}$ is noted at the falling edge of signal.

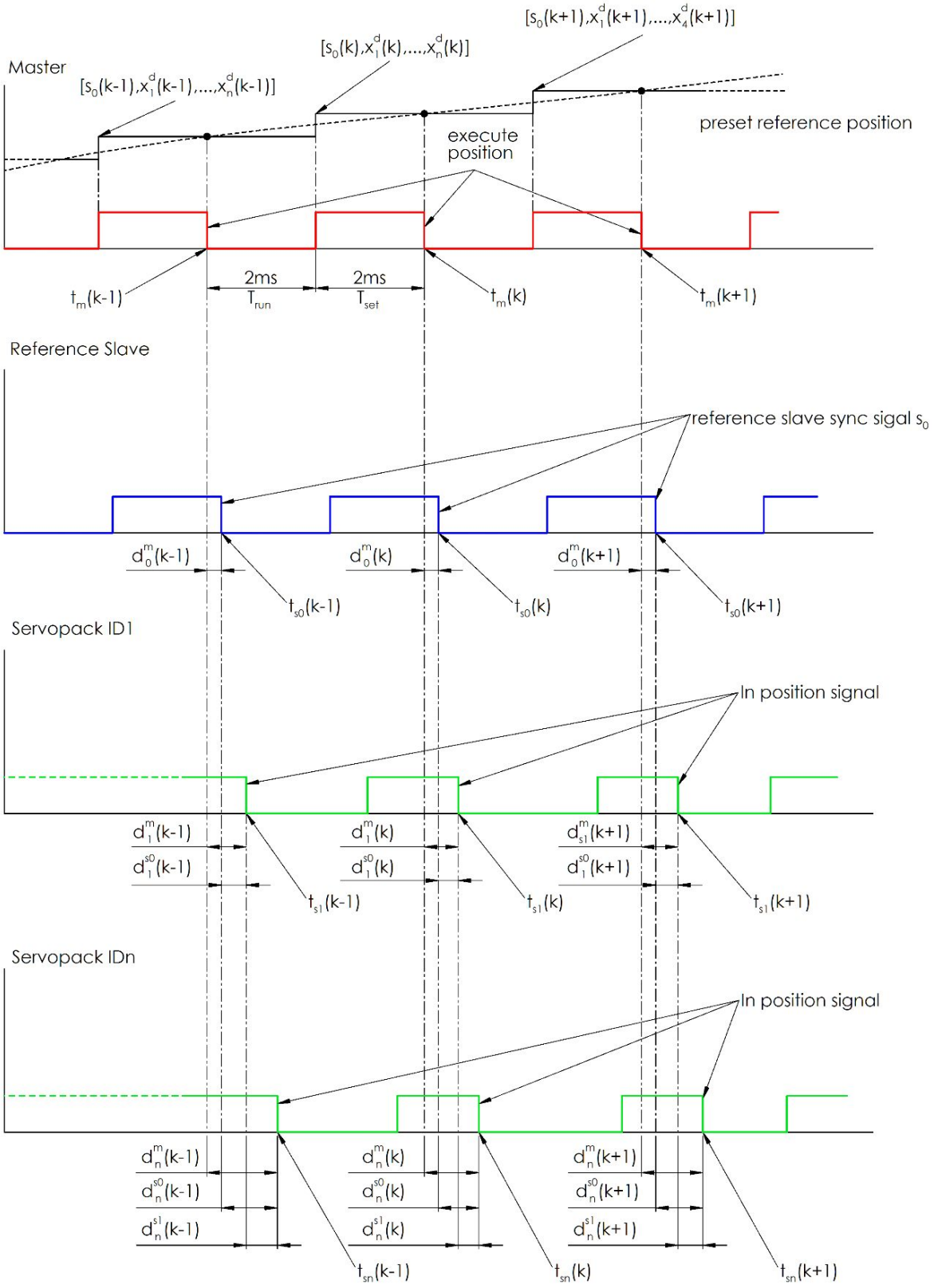


Fig. 7. Illustration of timing relations in the typical network. M: Master, S0: Reference Slave, S1-4: Slave Servo.

Table 4 describes the location of variable s_0,x_n^d in data frame of TX cache. In each station,

there is only one number for identification which could be recognized for the internal communication. In the second byte, the types of motion commands are classified due to the code. Subsequently, the status of servo on/off is written in the third byte. Also, user could verify this status by reading its value. Following, the setting period by master station indicates whether it is active or not. The value of target position is in next four bytes. Then, the reserved bytes for further implementation are from byte 9th to byte 12th. To provide the flexible setup, the system unit in our validation could be set via user-defined. This rate is stored in the command code of encoder setup. Hence, the velocity of each servo is calculated and saved from byte 13th to byte 16th.

Table 4. Description of bit location in data frame.

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
byte0				MAC-IDn				
byte1		T_{set}: command code = 0x10/ T_{run}: command code = 0x17						
byte2	Servo On = 1							
byte3								T_{set}: s_0 = 1/T_{run}: s_0 = 0
byte4	Target Position T_{set}: x_n^d = reference position. T_{run}: Do not care, until it reaches to the setting value in previous period T_{set}							
byte5								
byte6								
byte7								
byte8								
byte9								
byte10								
byte11								
byte12	$v_n^d = V\left(encoder\ unit/second\right)$							

5 EXPERIMENTAL VALIDATIONS

To verify the effectiveness and feasibility of our approach, an experimental platform of mobile base is established as Fig. 8. Four omnidirectional wheels are directly linked by the driving mechanism including gearboxes and servo motors. In this experiment, four sets of motor and driver MINAS A6N Panasonic are chosen to manipulate in the real-time network. Since the protocol of local area network (LAN) is utilized, many wiring cables could be reduced in comparison to the others. In addition, four servo drivers take up a significant amount of workspace and are quite heavy, accordingly they are positioned on the left side for easier management. To balance the load capacity of wheeled base, battery packs are located on the right side with fixed handrail. Instead of spending much time due to the charging battery, an operator can swap another one whenever the level of power is low.

Fig. 8. Experimental mobile platform for the proposed system.

In Fig. 9, the structure of data communication including host computer, one master station, one reference slave, and four servo slaves is launched to validate the high performance of real-time express protocol. The communication protocol between host computer and master is universal serial bus-human interface device (USB-HID) class which allows rapid transmission and low latency. Among slaves in this network, RX or TX port of one slave station can be linked with TX or RX port of the others by using LAN cables. Four sets of servo motor and driver which are labelled as CH2, CH3, CH4 and CH5, are connected to mechanical gearboxes. There is one reference slave CH1 to receive and transmit data feedback from sensing devices. From host computer to master station CH0, user inputs the desired trajectories for servo slaves and values of I/Os (Inputs/Outputs) for reference slave. In our test, data logger is to collect the feedback information from both master and slaves and transmit them to host. The programming languages are Visual C++ for software-based Windows application in host computer and embedded-C for firmware in slaves.

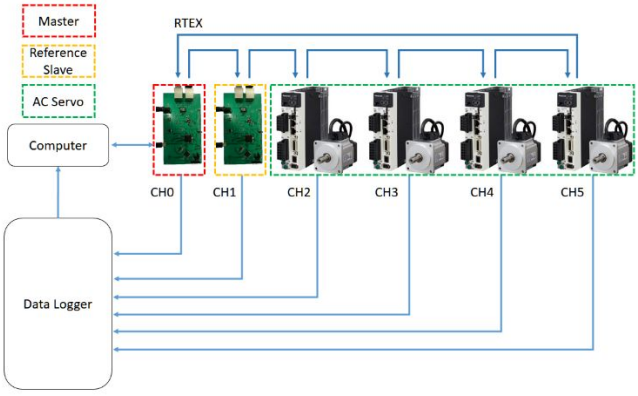


Fig. 9. Overall diagram of hardware connection for the proposed system.

Procedure of validation:

+ Establish the network connection between master and five slaves.

+ Master station receives command from computer and send to five slaves

+ Synchronization test: data logger records the response time of five slaves. From this result, command response delay, synchronization error and jitter could be estimated.

+ Validation to send command for motion profile over network, and get feedback data from this network (velocity, position).

Network setup

ID for AC servo varies from 1 to 4 while ID for reference slave is 0

After the connection is successful and the master station enters the RUNNING state, proceed to send commands to the TX cache as Fig. 10. With 4000 cycles for communication, it includes 2000 cycles for T_{run}, it is expected to receive 2000 feedback data from the master and slaves.

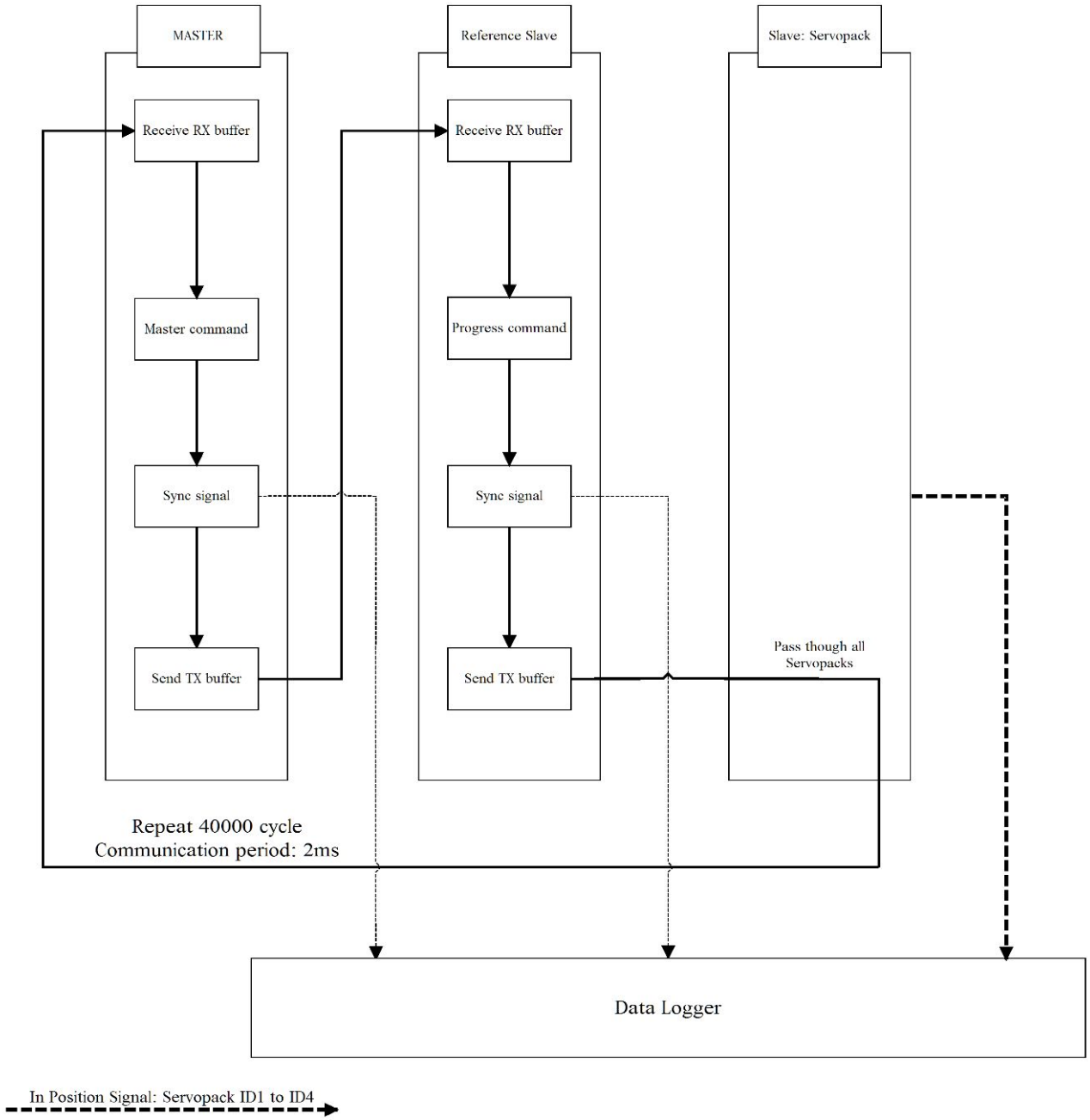


Fig. 10. Block diagram for data collection process in our experiment.

Output setup to synchronize data

Master

Whenever a new period of network begins (every 2 ms), it toggles the voltage level of output pin at the time before sending data in TX cache. The value to toggle would be registered by bit EX-OUT1.

Reference slave

At the beginning of each network cycle, it scans into byte 3 of RX cache to look for the value of bit EX-OUT1. (For the TX cache, the 3-bit byte EX-OUT1 of the master). This output manner would be changed during our experiments to achieve the desired value.

It toggles the voltage level of one output pin with the received EX-OUT1 bit value. Then, write it to the EX-OUT1 bit value.

AC servo

Using Panaterm software to set SO1 value corresponding to INP (In Position). Other settings such as In Position Range (parameter code: Pr4.31) set to 0 to output signal only when servo stops in position, Pr4.32 defaults to 0 active low signal, INP hold time (parameter code: PR4.33) default set it to 0 so that the signal only resets when a new position is received. The same settings are for all four servos.

Timing deviation from master station to slave stations

Before sending data to TX cache, master station would output in order to obtain the time required to process the signal of each slave as Table 5. Thereby, the firmware of master could be adjusted in the appropriate manner. Besides, a generalized delay from the start of sending the motion command to the completion of movement (InPosition) in the AC servo slaves.

Table 5. Estimation of timing errors from master station to slave station.

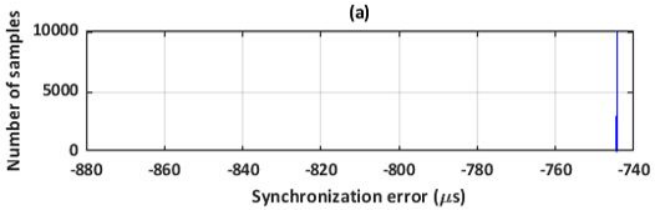
	S0	S1	S2	S3	S4
M	$d_0^m = t_m - t_{s0}$	$d_1^m = t_m - t_{s1}$	$d_2^m = t_m - t_{s2}$	$d_3^m = t_m - t_{s3}$	$d_4^m = t_m - t_{s4}$

In Fig. 11a, it is documented that this chart is distributed in the range from -750 (μs) to -744 (μs) and peaked at -744 (μs). In Fig. 11b, Fig. 11c and Fig. 11d, the AC servo slaves distribute the synchronization errors varying from -870 (μs) to -865 (μs) and maximized at -867 (μs). There are two reasons, (a) the firmware of reference slave releases the feedback signal as soon as it read the fourth byte in RX cache (byte3 contains bit s_0), (b) AC servo slaves travel a distance $x_n^d(k)$ then output the InPosition signal.

In the reference slave, to ensure the output signal closed to the time when servo slaves begin to run, its firmware must be modified so that when receiving 16 bytes, RX cache would proceed to output. In Fig. 12, the results when taking new sample t_{s0} are demonstrated, hence d_{s0}^m could be computed. These values fall in the range from -855 (μs) to -853 (μs), and peaked at -854 (μs) where it closes to the distribution area of timing error in AC servo slaves.

Timing deviation between reference slave and servo slaves

Taking the time to output of reference slave such that the timing errors of servo slaves could be calculated as Table 6, our tests are categorized into two cases, before and after adjusting the firmware of reference slave. These values are meaningful when it is essential to develop a slave device that can be combined with AC servo slave or other industrial devices that support our protocol. Thus, reference slave could be considered as an I/O slave in the network.



(a)

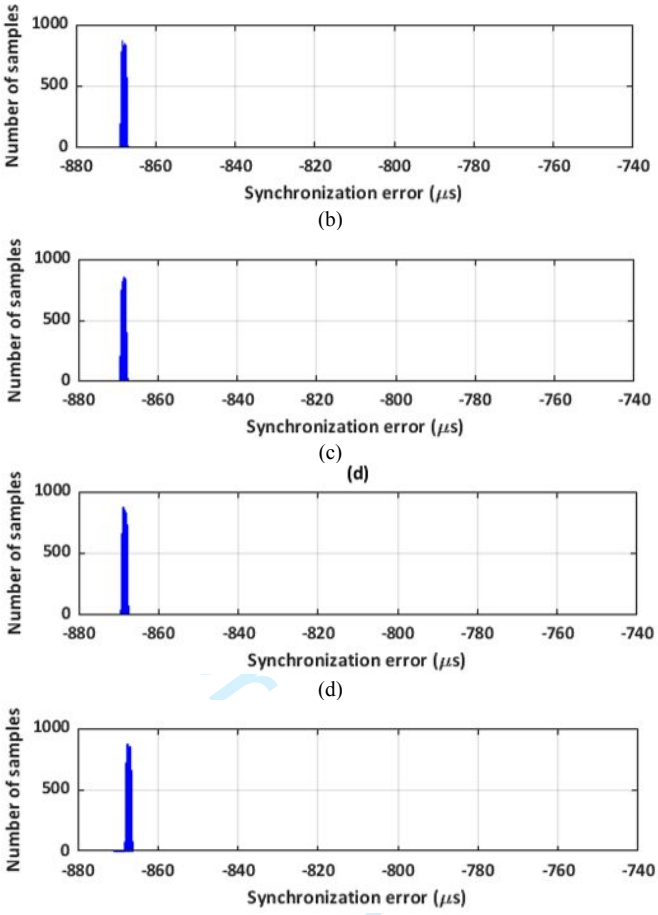
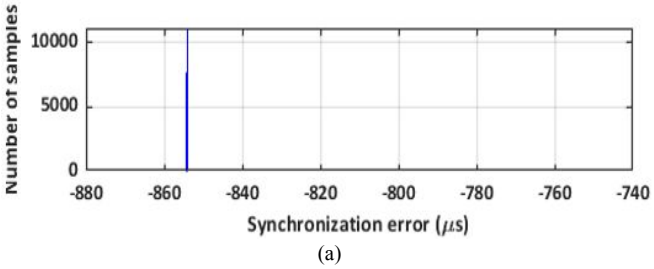


Fig. 11. Distribution diagram of timing errors from master station to slave stations, (a) master to reference slave, (b) master to servo ID 1, (c) master to servo ID 2, (d) master to servo ID 3 and (e) master to servo ID 4.

Table 6. Estimation of timing errors from reference slave to servo slaves.

	S1	S2	S3	S4
S0	$d_1^{s0} = t_{s0} - t_{s1}$	$d_2^{s0} = t_{s0} - t_{s2}$	$d_3^{s0} = t_{s0} - t_{s3}$	$d_4^{s0} = t_{s0} - t_{s4}$

In Fig. 13, it indicates that the differences of feedback signals between reference slave and servo slaves are up to -120 (μs). As mentioned before, it is caused by firmware of reference slave and timing transmission of servo slaves, which is largely due to the location of output feedback in reference slave. After adjusting the firmware, these differences are significantly improved from greater than -120 (μs) to less than -16 (μs) as Fig. 14. Above and beyond, to distinguish these differences, timing errors from reference slave to servo slaves are illustrated as Fig. 15. It is noticeably known that most of values of timing errors would decrease 9 times approximately after adjusting the firmware of reference slave.



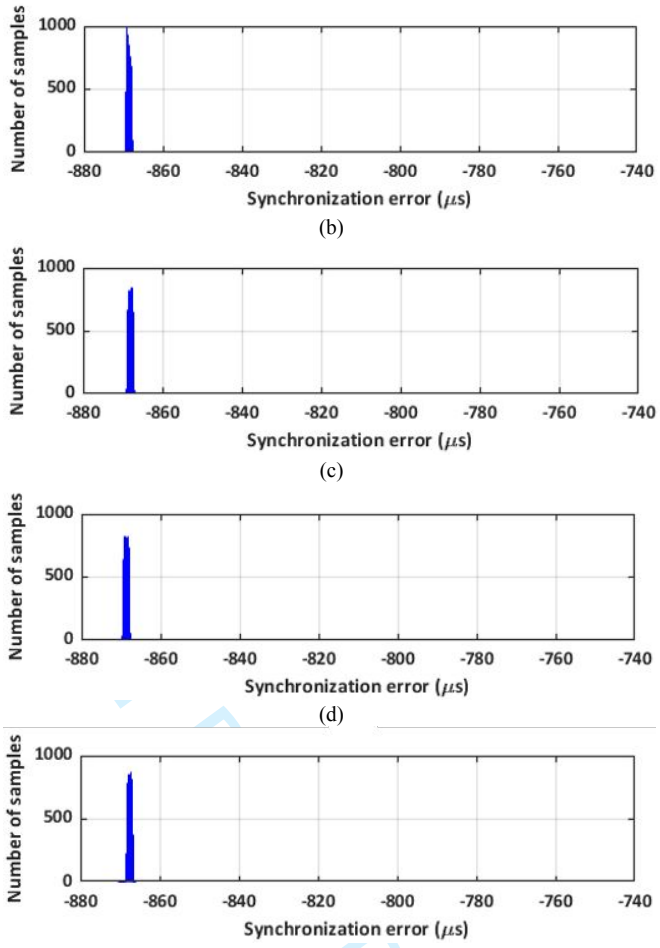
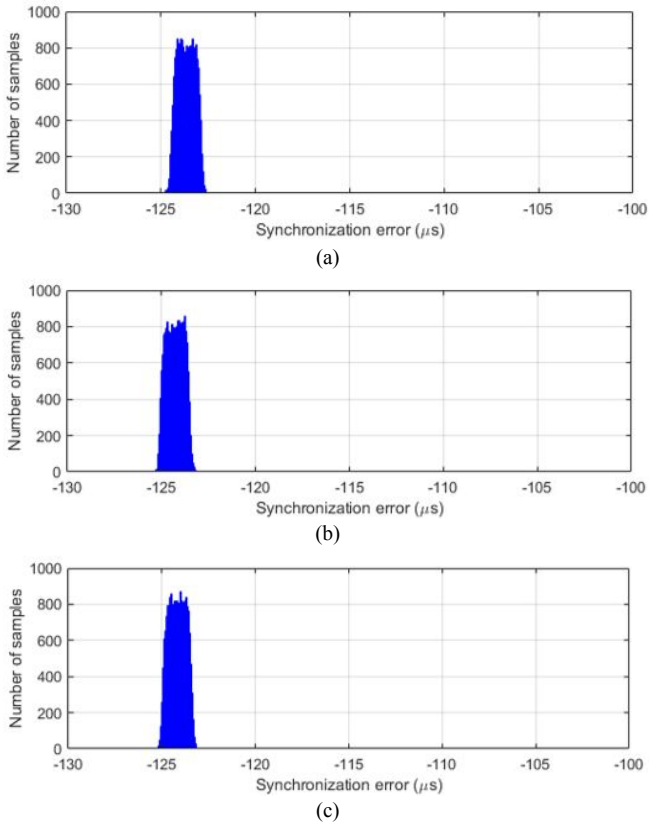
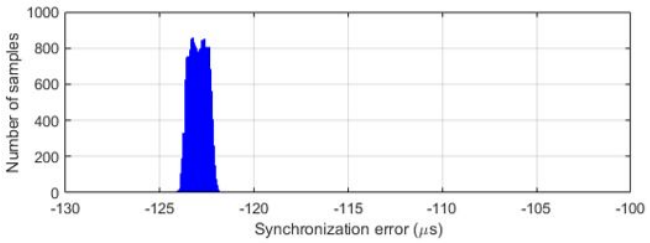


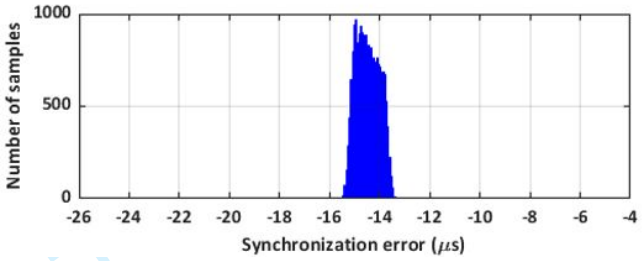
Fig. 12. Distribution diagram of timing errors from master station to slave stations after adjusting the firmware of reference slave, (a) master to reference slave, (b) master to servo ID 1, (c) master to servo ID 2, (d) master to servo ID 3 and (e) master to servo ID 4.



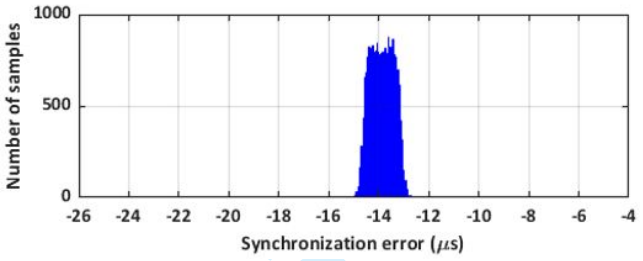


(d)

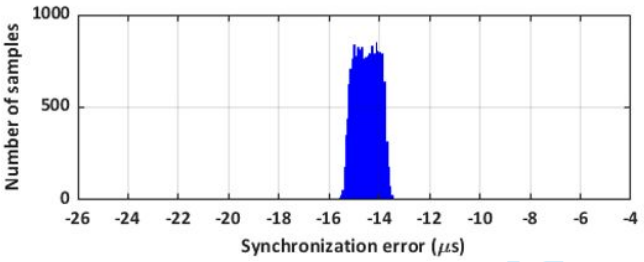
Fig. 13. Distribution diagram of timing errors from reference slave to servo slaves, (a) reference slave to servo ID 1, (b) reference slave to servo ID 2, (c) reference slave to servo ID 3, (d) reference slave to servo ID 4.



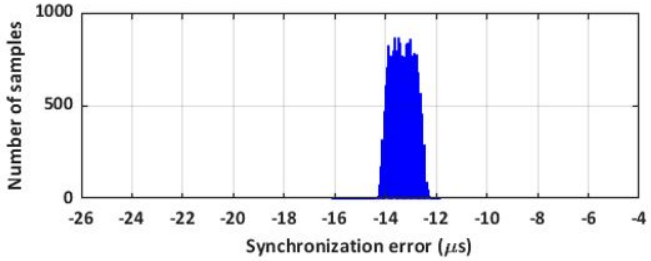
(a)



(b)



(c)



(d)

Fig. 14. Distribution diagram of timing errors from reference slave to servo slaves after adjusting the firmware of reference slave, (a) reference slave to servo ID 1, (b) reference slave to servo ID 2, (c) reference slave to servo ID 3, (d) reference slave to servo ID 4.

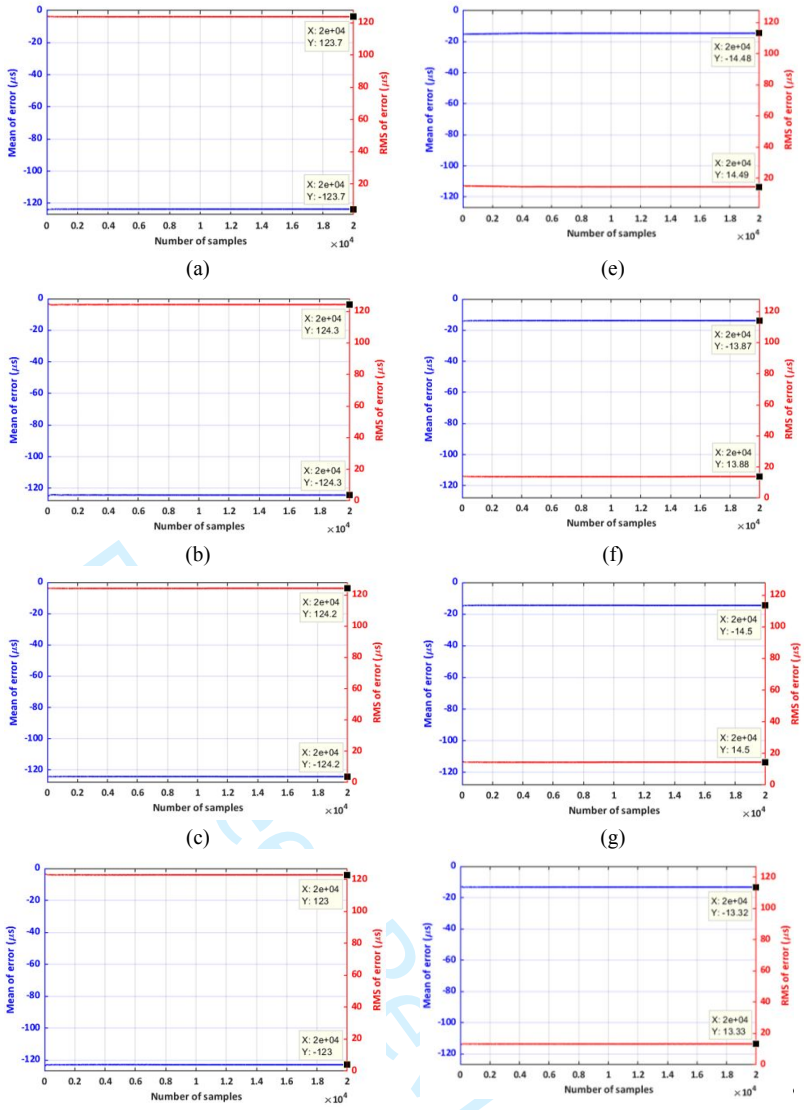


Fig. 15. Diagram for average value and RMS (Root-Mean-Square) value of timing errors from reference slave to servo slaves, (a)-(d) before adjusting the firmware of reference slave, (e)-(h) after adjusting the firmware of reference slave, (a) and (e) from reference slave to servo ID 1, (b) and (f) from reference slave to servo ID 2, (c) and (g) from reference slave to servo ID 3, (d) and (h) from reference slave to servo ID 4.

Timing deviation among servo slaves

When considering the synchronization of four AC servo slaves, it takes the servo slave ID1 as the origin from which to compute the timing errors among them as Table 7. This value is equivalent to the amplitude of jitter during the collaboration of the AC servo slaves. If this value is smaller, the ability to coordinate among servo slaves would be higher. As a result, it helps to control the servo actuators more accurately and safely. This value does not take into account the delay in sending commands from master station to the servo slaves or reference slave since reference slave is not involved in motion control.

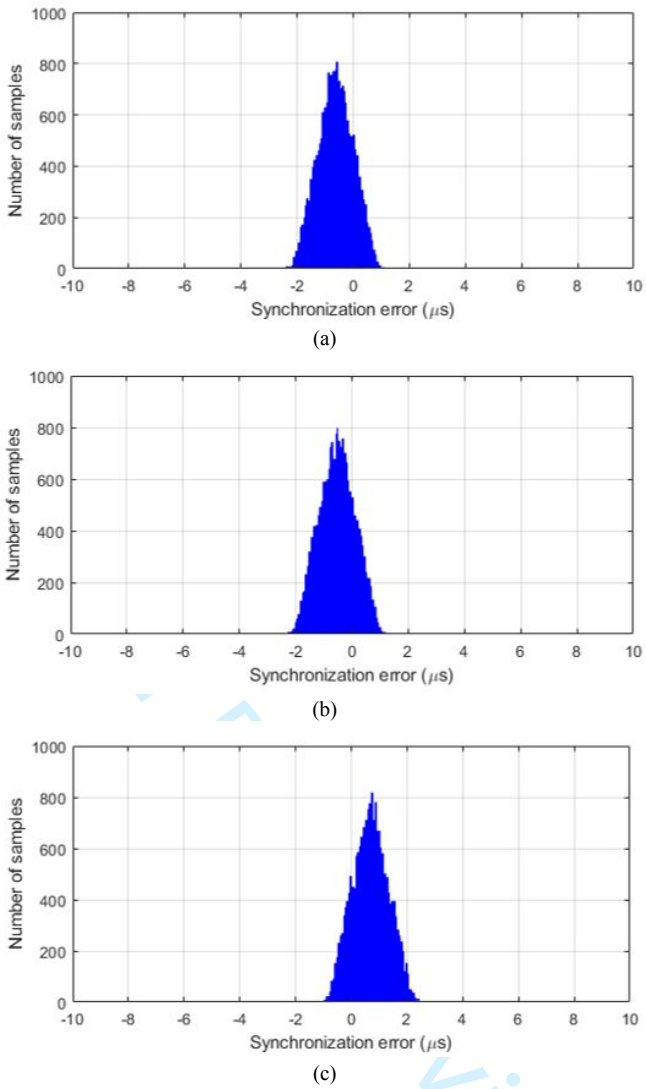


Fig. 16. Distribution diagram of timing errors among servo slaves, (a) servo slave ID 1 and servo slave ID 2, (b) servo slave ID 1 and servo slave ID 3, (c) servo slave ID 1 and servo slave ID 4.

Table 7. Estimation of timing errors among servo slaves.

	S2	S3	S4
S1	$d_2^{s1} = t_{s1} - t_{s2}$	$d_3^{s1} = t_{s1} - t_{s3}$	$d_4^{s1} = t_{s1} - t_{s4}$

In Fig. 16a, Fig. 16b and Fig. 16c, their widths of distribution diagrams are about 4 (μs). With vertices falling into -1 (μs) at Fig. 16a and Fig. 16b, +1 (μs) at Fig. 16c. Although it sometimes falls out of its range, manufacturer requires that it falls within ±1 (μs). Nonetheless, it could be seen clearly that more half of the samples are within the safe range.

Profile position control

As in the above experiments, the frame transmitted in TX cache is still the same, but the position and velocity values would be large enough for the servo motor to run through the full speed profile as Table 7. In this validation, the k^{th} reference value could be sent manually in the network. In detail, k_1 would be sent after the state machine is running and the status of servo slave is on. Later, k_2 is sent after completing k_1 successfully. And k_3 is sent after servo slave reaches to target position, k_4 is sent right after successful sending k_3.

Table 7. List of reference parameters, position [unit], velocity [unit/second].

	T_{set}	T_{run}	T_{set}	T_{run}
Position	$x_n^d(k_1) = 10^7$	$x_n^d(k_2) = 10^7$	$x_n^d(k_3) = 0$	$x_n^d(k_4) = 0$
Velocity	$v_n^d(k_1) = 10^6$	$v_n^d(k_2) = 10^6$	$v_n^d(k_3) = 10^6$	$v_n^d(k_4) = 10^6$

There is no constraint between k_1 and k_2, k_3 and k_4, after sending the reference value at k_1 and k_1. Then, the servo slave would just wait for the running command to operate. Acceleration is assigned by Panaterm software to some registers such Pr8.1 and Pr8.4. Like position and velocity, the unit of acceleration would be unit/s^2. In this test, the symmetric profile is chosen so that the value of acceleration equals to that of deceleration.

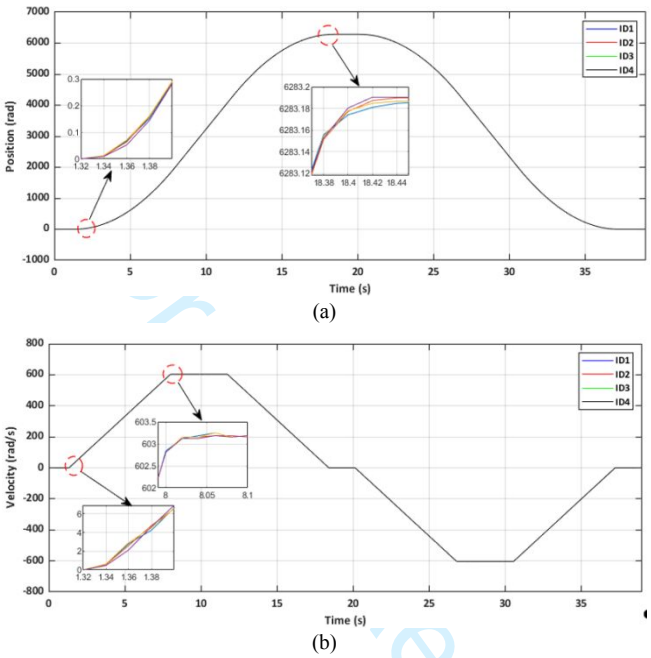


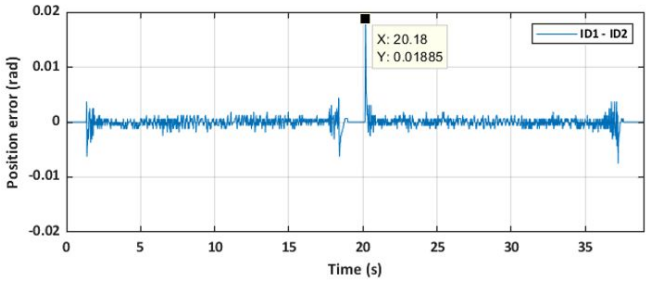
Fig. 17. Experimental result of motion profile in our approach including position (a) and velocity (b).

The command unit is estimated based on three registers, Pr0.8, Pr0.9 and Pr0.10. If we use Pr0.8 and Pr0.8 ≠ 0, then Pr0.9 and Pr0.10 will be ignored. The meaning of register Pr0.8 is the number of command units per revolution of the servo motor, or command unit per revolution. In our experiment, Pr0.8 is set to 10000. For example, sending a command to run in absolute position $x_0^d = 10000000$ will stop the servo motor at +360,000° or 2000π (rad). It is noted that the position and velocity are returned by the servo drive in the command unit. So, we have

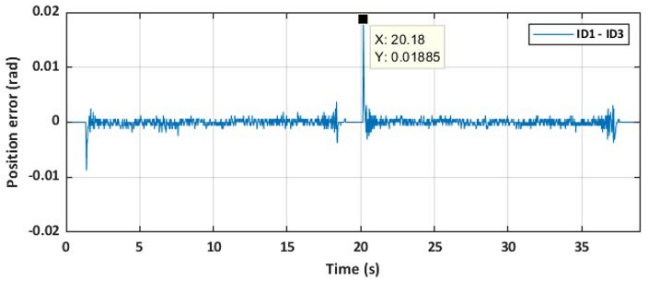
$$x(rad) = x(command\ unit)\frac{2\pi}{10000} \quad (14)$$

$$v\left(^{rad}/_{s}\right) = v\left(^{command\ unit}/_{s}\right)\frac{2\pi}{10000} \quad (15)$$

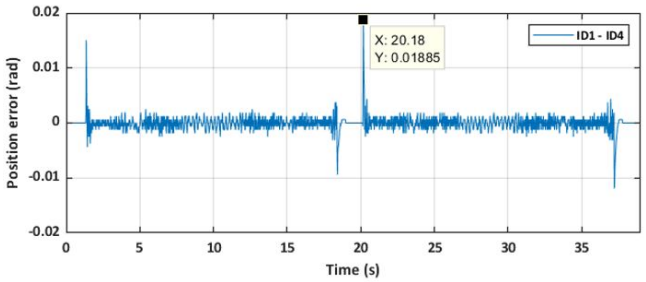
In Fig. 17, it could be observed evidently that the position and velocity of four servo slaves match with the user-defined value which are sent through SRTX network. The following trials to clarify the synchronization of four servo slaves are to obtain the data of both position and velocity in servo slave ID 1 as origin. Thereby, the timing errors is computed due to the performance of other three servo motors.



(a)



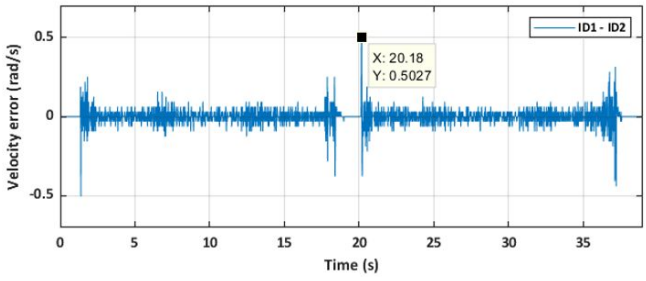
(b)



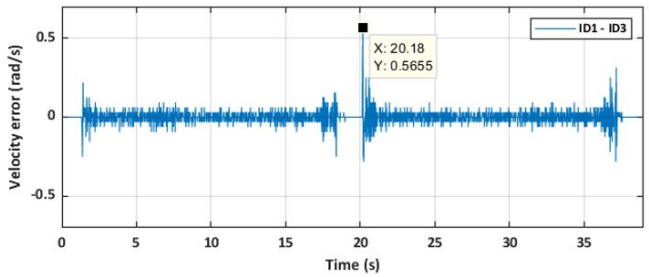
(c)

Fig. 18. Experimental result of position error from servo slave ID 1 to the other servo slaves, (a) servo slave ID 1 and servo slave ID 2, (b) servo slave ID 1 and servo slave ID 3, and (c) servo slave ID 1 and servo slave ID 4.

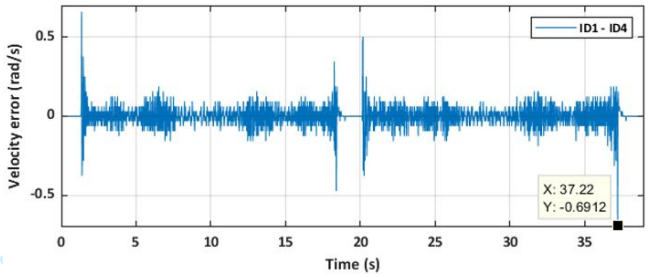
Fig. 18 shows the greatest increase in timing error at each start of a new run. At that moment, when the servo motor finishes running, it also causes the wrong position command instantly but lower than when starting to run. Similar to the position error, the velocity error of three servo motors as Fig. 19 compared with the performance of servo slave ID1 also increased the most at the beginning and end of the run. However, the error value does not necessarily increase sharply at the start of the run, as Fig. 19c shows the largest error at the end and at the position $x_n^d = 0\ (rad)$.



(a)



(b)



(c)

Fig. 19. Experimental result of velocity error from servo slave ID 1 to the other servo slaves, (a) servo slave ID 1 and servo slave ID 2, (b) servo slave ID 1 and servo slave ID 3, and (c) servo slave ID 1 and servo slave ID 4.

To visualize the superior performance of our protocol, the other communicative networks such as CANopen [32] and EtherCAT [33] are suggested in this comparison. Table 8 portrays a list of competitive values in latency and synchronization error between CANopen, EtherCAT and the proposed approach. While EtherCAT is designed for ultra-low latency and highly precise synchronization, CANopen tends to have higher latency and larger synchronization errors under similar conditions. In fact, our protocol is somewhat better than those values of EtherCAT network, and it is significantly faster than CANopen. As well, the sub-microsecond synchronization error of this protocol reinforces its suitability for applications requiring tight coordination, all while offering advantages in cost and compatibility. Our method benefits from a lower implementation cost due to simplified hardware requirements and easier integration into existing systems. It also supports a wide range of embedded platforms, making it a versatile choice for various industrial fields.

Table 8. Competitive list in latency and synchronization error of related protocols.

Protocol / Item	CANopen [32]	EtherCAT [33]	Our approach
Latency	1–10 ms	20–100 us	10–50 us
Synchronization error	10us	< 1 us	< 0.5 us

6 CONCLUSIONS

In this paper, the application of synchronous control for a mobile platform in a wheeled manipulator was successfully validated. The study focused on the integration of synchronization mechanisms to achieve coordinated motion between the mobile platform and its manipulator. Both hardware and software implementations were designed to support the synchronous control system, demonstrating practical feasibility without requiring significant modifications to the underlying protocols or specialized designs. To emulate real-world operational conditions, a multi-station experimental setup was developed. The synchronization performance was analyzed at multiple levels, including between the mobile platform and the

manipulator, among various servo components, and across the overall system. Synchronization errors in position and velocity profiles were measured and evaluated. The results indicate that the proposed approach is effective in ensuring coordinated motion, contributing to enhanced operational precision and reliability. However, the synchronization accuracy is influenced by system factors such as communication jitter and environmental disturbances. To mitigate these effects, users are advised to adopt robust communication channels, avoid exposure to strong electromagnetic fields, and ensure stable system timing. Future research will focus on advanced jitter management techniques and optimizing system timing to further enhance synchronization performance, ensuring seamless integration of mobile platforms and manipulators in dynamic industrial environments.

CONFLICT OF INTEREST

None declare.

ACKNOWLEDGEMENTS

We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

REFERENCES

[1] Thakar, Shantanu, Srivatsan Srinivasan, Sarah Al-Hussaini, Prahar M. Bhatt, Pradeep Rajendran, Yeo Jung Yoon, Neel Dhanaraj et al. "A survey of wheeled mobile manipulation: A decision-making perspective." *Journal of Mechanisms and Robotics* 15, no. 2 (2023): 020801.

[2] Sinnemann, J., Boshoff, M., Dyrska, R., Leonow, S., Mönnigmann, M., & Kuhlenkötter, B. (2022). Systematic literature review of applications and usage potentials for the combination of unmanned aerial vehicles and mobile robot manipulators in production systems. *Production Engineering*, *16*(5), 579-596.

[3] Yamamoto, T., Terada, K., Ochiai, A., Saito, F., Asahara, Y., & Murase, K. (2019). Development of human support robot as the research platform of a domestic mobile manipulator. *ROBOMECH journal*, *6*(1), 1-15.

[4] Ghodsian, N., Benfriha, K., Olabi, A., Gopinath, V., Arnou, A., & Charrier, Q. (2022). Toward designing an integration architecture for a mobile manipulator in production systems: Industry 4.0. *Procedia CIRP*, *109*, 443-448.

[5] He, Z., Zhang, X., Jones, S., Hauert, S., Zhang, D., & Lepora, N. F. (2023). TacMMs: Tactile mobile manipulators for warehouse automation. *IEEE Robotics and Automation Letters*, *8*(8), 4729-4736.

[6] Choi, S., & Park, S. (2021). Development of smart mobile manipulator controlled by a single windows PC equipped with real-time control software. *International Journal of Precision Engineering and Manufacturing*, *22*(10), 1707-1717.

[7] Koch, P. J., van Amstel, M. K., Dębska, P., Thormann, M. A., Tetzlaff, A. J., Bøgh, S., & Chrysostomou, D. (2017). A skill-based robot co-worker for industrial maintenance tasks. *Procedia Manufacturing*, *11*, 83-90.

[8] Bejczy, B., Bozyil, R., Vaičekauskas, E., Petersen, S. B. K., Bøgh, S., Hjorth, S. S., & Hansen, E. B. (2020). Mixed reality interface for improving mobile manipulator teleoperation in contamination critical applications. *Procedia Manufacturing*, *51*, 620-626.

[9] Rocha, F., Garcia, G., Pereira, R. F., Faria, H. D., Silva, T. H., Andrade, R. H., ... & Lizarralde, F. (2021). Rosi: A robotic system for harsh outdoor industrial inspection-system design and applications. *Journal of Intelligent & Robotic Systems*, *103*, 1-22.

[10] Štibinger, P., Broughton, G., Majer, F., Rozsypálek, Z., Wang, A., Jindal, K., ... & Saska, M. (2021). Mobile manipulator for autonomous localization, grasping and precise placement of construction material in a semi-structured environment. *IEEE Robotics and Automation Letters*, *6*(2), 2595-2602.

[11] Song, J., & Sharf, I. (2022). Stability-constrained mobile manipulation planning on rough terrain. *Robotica*, *40*(11), 4090-4119.

[12] Sun, Z., Yang, H., Ma, Y., Wang, X., Mo, Y., Li, H., & Jiang, Z. (2021). BIT-DMR: A humanoid dual-arm mobile robot for complex rescue operations. *IEEE Robotics and Automation Letters*, *7*(2), 802-809.

[13] Naazare, M., Rosas, F. G., & Schulz, D. (2022). Online next-best-view planner for 3D-exploration and inspection with a mobile manipulator robot. *IEEE Robotics and Automation Letters*, *7*(2), 3779-3786.

[14] Wang, F., Olvera, J. R. G., & Cheng, G. (2021). Optimal order pick-and-place of objects in cluttered scene by a mobile manipulator. *IEEE Robotics and Automation Letters*, *6*(4), 6402-6409.

[15] Li, Y., He, H., Chen, Y., & Wang, H. (2023). A cloud-based eco-driving solution for autonomous hybrid electric bus rapid transit in cooperative vehicle-infrastructure systems: A dynamic programming approach. *Green Energy and Intelligent Transportation*, *2*(6), 100122.

[16] Xie, D. J., Zeng, L. D., Xu, Z., Guo, S., Cui, G. H., & Song, T. (2023). Base position planning of mobile manipulators for assembly tasks in construction environments. Advances in Manufacturing, 11(1), 93-110.

[17] Balatti, P., Fusaro, F., Villa, N., Lamon, E., & Ajoudani, A. (2020). A collaborative robotic approach to autonomous pallet jack transportation and positioning. *IEEE Access*, *8*, 142191-142204.

[18] Li, F., Jiang, Q., Quan, W., Song, R., & Li, Y. (2019, July). Manipulation skill acquisition for robotic assembly using deep reinforcement learning. In *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)* (pp. 13-18). IEEE.

[19] Li, C., Li, B., Wang, R., & Zhang, X. (2021). A survey on visual servoing for wheeled mobile robots. *International Journal of Intelligent Robotics and Applications*, *5*(2), 203-218.
[20] Nguyen, T. P., Nguyen, H., & Ngo, H. Q. T. (2024). Development of the robotic motion controller for a wheeled manipulator. Measurement and Control, 00202940241260178.
[21] Racz, S. G., Crenganiș, M., Breaz, R. E., Maroșan, A., Bârsan, A., Gîrjob, C. E., ... & Tera, M. (2022). Mobile Robots—AHP-Based Actuation Solution Selection and Comparison between Mecanum Wheel Drive and Differential Drive with Regard to Dynamic Loads. *Machines*, *10*(10), 886.
[22] Bhatt, P. M., Malhan, R. K., Rajendran, P., & Gupta, S. K. (2020). Building free-form thin shell parts using supportless extrusion-based additive manufacturing. *Additive Manufacturing*, *32*, 101003.
[23] Nguyen, T. P., Nguyen, H., & Ngo, H. Q. T. (2023). Visual application of navigation framework in cyber-physical system for mobile robot to prevent disease. *International Journal of Advanced Robotic Systems*, *20*(2), 17298806231162202.
[24] Hu, S., Ren, X., Zheng, D., & Chen, Q. (2024). Neural network-based robust adaptive synchronization and tracking control for multi-motor driving servo systems. *IEEE Transactions on Transportation Electrification*.
[25] Xiong, Y., Ge, Y., Grimstad, L., & From, P. J. (2020). An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation. *Journal of Field Robotics*, *37*(2), 202-224.
[26] Park, D., Hoshi, Y., Mahajan, H. P., Kim, H. K., Erickson, Z., Rogers, W. A., & Kemp, C. C. (2020). Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned. *Robotics and Autonomous Systems*, *124*, 103344.
[27] Lin, T. C., Krishnan, A. U., & Li, Z. (2022). Intuitive, efficient and ergonomic tele-nursing robot interfaces: Design evaluation and evolution. *ACM Transactions on Human-Robot Interaction (THRI)*, *11*(3), 1-41.
[28] Yamamoto, T., Terada, K., Ochiai, A., Saito, F., Asahara, Y., & Murase, K. (2019). Development of human support robot as the research platform of a domestic mobile manipulator. *ROBOMECH journal*, *6*(1), 1-15.
[29] Song, L., Li, J., Wei, Z., Yang, K., Hashemi, E., & Wang, H. (2023). Longitudinal and lateral control methods from single vehicle to autonomous platoon. *Green Energy and Intelligent Transportation*, *2*(2), 100066.
[30] Nguyen, H., Nguyen, T. P., & Ngo, H. Q. T. (2024). Using EtherCAT technology to launch online automated guided vehicle manipulation with unity-based platform for smart warehouse management. IET Control Theory & Applications, 18(2), 229-243.
[31] Cabral, J. V. A., Álvares, A. J., & de Carvalho, G. C. (2024). Digital Twin Implementation for an Additive Manufacturing Robotic Cell based on the ISO 23247 Standard. *IEEE Latin America Transactions*, *22*(8), 651-658.
[32] Chen, J., Zhang, H., Pan, F., Du, M., & Ji, C. (2022). Control system of a motor-driven precision no-tillage maize planter based on the CANopen protocol. *Agriculture*, *12*(7), 932.
[33] Nguyen, H., Nguyen, T. P., & Ngo, H. Q. T. (2024). Using EtherCAT technology to launch online automated guided vehicle manipulation with unity-based platform for smart warehouse management. *IET Control Theory & Applications*, *18*(2), 229-243.