The design and implementation of "teaching-reproduction" algorithm for servo motor based on RFC 470 PLC

Yu Xiao1,a, Xiangyu Kong2,b, Yulin Zhang3,c*
1School of Information Science and Engineering University of Jinan Jinan, China
2School of Information Science and Engineering University of Jinan Jinan, China
3School of Information Science and Engineering University of Jinan Jinan, China
a e-mail: 1036840440@qq.com
c*Corresponding Author e-mail: ise_zhangyl@ujn.edu.cn

Abstract—Aiming at the characteristics of more swing and less rotation of the nozzle at the end of teaching-type sand-blasting robots, a motion control algorithm which can reproduce the swing and rotation trajectory of the servo motor is designed. The RFC 470 PLC of PHOENIX Contact is used as the control core to build the hardware experimental platform in this paper. In the process of teaching, the trajectories of the servo motor are saved to the data storage of the RFC 470 PLC by the algorithm and then the trajectories can be reproduced automatically. The experimental results show that the designed algorithm has high accuracy and can meet the needs of practical engineering application.

1. INTRODUCTION
With the proposal of the 'made in China 2025' national strategy, teaching-type robots have developed rapidly and gained new applications in the sand-blasting cleaning industries. Servo motor is the basic motion unit of teaching-type robots, so the study of "teaching-reproduction" technology of servo motor plays an important role in promoting the application of teaching-type robots in the emerging industries. When the teaching-type sand-blasting robots process the workpiece, the motion of the nozzle is characterized by more oscillation and less rotation, while the oscillation and rotation of the nozzle are realized by servo motor[1-3]. In order to meet the needs of the sand-blasting industries, a motion control algorithm which can reproduce the swing and rotation trajectories of the servo motor is designed in this paper. The RFC 470 PLC of PHOENIX company is used as the control core, a hardware experiment platform is built and the algorithm is realized by programming.

2. DESIGN SCHEME
The design scheme is shown in Fig. 1. It mainly includes a PC, a main controller, a motion controller, a servo motor, and a matching servo driver. The PC is connected to the main controller via a USB cable, and sends specific operating instructions to realize the control function of the motor. An algorithm that enables the "teaching-reproducing" function of the servo motor is developed in the main controller. Driver of servo motor are written in motion controller. The servo driver is responsible for driving the servo motor to move according to specific operation instructions.
3. HARDWARE EXPERIMENTAL PLATFORM
The RFC 470 PLC of PHOENIX Company is chosen as the main controller in this paper. The PLC has 8Mbyte program content, 16Mbyte data memory, supports multiple communication protocols and can run stably under harsh working environment. The DHR41B PLC of SEW Company is chosen as the motion controller. The reason for choosing this controller is that it is relatively easy to implement some basic motion functions of the motor, such as jogging, zero position setting, absolute positioning, relative positioning, speed controlling, etc. The developer can directly call the corresponding function block. At the same time, DHR41B PLC can realize the intelligent coordination between multiple servo motors through the fast synchronization system bus, and provides hardware support for the later "teaching-reproducing" function of multiple servo motors. Servo motors and servo controllers use SEW products, which are CMP series and MDX61B series. The communication protocol between the motion controller and the servo driver is the CAN bus protocol. The servo driver and the servo motor are connected through a power cable.

4. DESIGN AND IMPLEMENTATION OF ALGORITHM
4.1. Servo motor driver
The driver of the servo motor is developed in the motion controller. The main function is to realize the three working modes of the motor during the movement: jogging mode, autozero mode, positioning mode[4]. In jogging mode, the motor first reaches the set speed with a certain acceleration, and then moves at a constant speed. The acceleration and the set speed are adjustable. In autozero mode, the current position coordinate of the motor is set to 0. In positioning mode, the motor moves to the set position coordinates at a certain speed, and the speed and set position coordinates can be changed. The above three working modes can be implemented by the function block integrated in DHR41B PLC. The driver is shown in Fig. 2. During the operation of the algorithm designed in this paper, the motion controller DHR41B receives instructions issued by the main controller RFC 470, switches the different motor working modes, and implements the "teaching-reproducing" function.
4.2. Communication between controllers

According to the model of the equipment used in this paper and the servo motor must have a sufficiently high accuracy during the "teaching-reproducing" process, the Profinet protocol is selected as the communication protocol between the main controller and the motion controller[5]. Configure the IP address of the main controller RFC 470 as: 192.168.0.20, and the IP address of the motion controller DHR41B as: 192.168.0.24, and the subnet masks are: 255.255.255.0 to ensure that the devices are in the same network segment. During the running of the algorithm, the main controller RFC 470 sends the motor's working mode, set speed, target position coordinates and other information to the motion controller DHR41B, while receiving the motor working status, instantaneous motion speed, and a series of the current position coordinate information, and save this information in the memory space of the main controller. The communication contents are shown in Table I, and the communication message format of the variable 'C01U1-SET-CMD' is shown in Table II.

4.3. The teaching process

The operator teaches the servo motor through the PC. After the rotation command is issued, the servo motor rotates at a constant speed after reaching a set speed with a certain acceleration. When it stops rotating, the servo motor decelerates to a stationary state with a certain acceleration. Acceleration, set speed, and direction of rotation can be set in the PC. After the swing command is issued, the servo motor starts swinging with the current position coordinate as the swing center point, the swing speed is 20 °/s, the initial swing amplitude is 30 °, the swing amplitude is adjustable and can be increased or decreased by 10 ° each time. After the stop swing command is issued, the servo motor stops from swinging after moving from the current position to the swing center point.

During the teaching process, the algorithm saves different data according to the different motion states of the motor. 'The TP Pulse Timer' and 'The TON On-delay Timer' (triggered by 200ms) in the main controller are used to save the motor trajectories in the data store of the main controller[6]. Before the teaching process starts, the motor is adjusted to an appropriate position, and then the current position coordinate is set to 0 by the autozero mode of the motor as the starting point of the teaching process. After the teaching process starts, when the motor is rotating, the algorithm saves the current position coordinates and set speeds of the motor every 200ms, and saves the current position coordinates in the first array and the set speeds in the second array. When the motor is in the swing state, the algorithm saves the position coordinates of the swing center point in the first array, and saves the time points when the swing started and the time points when the swing stopped to the third array. If the swing amplitude is changed during the swing, the time points of changing the swing amplitude and the changed swing amplitude are saved to the fourth array. If the swing amplitude is not changed, no data is saved in the fourth array.

4.4. Reproduction process

The design idea of realizing the reproduction process is shown in Fig. 3. The variable 'SamePoints_flag' is a Boolean variable with an initial value of 'true', and the variable 'UnlikePoints_flag' is a Boolean variable with an initial value of 'false'. After the reproduction process starts, the timer is started, the trigger time is 200ms, and the variable 't' is automatically incremented after each trigger timing interruption.

The algorithm analyzes the motor position coordinates in the first array to get the first value of 'Special_point'. The idea is as follows: if the motor is stationary or in a swing state, the

| Variable name          | Input/Output | Address          | Meaning                              |
|------------------------|--------------|------------------|--------------------------------------|
| C01U1_Status           | Output       | Fieldbus.OUT[1]  | Current state of servo motor         |
| C01U1_Pos              | Output       | Fieldbus.OUT[2]  | Current position of servo motor      |
| C01U1_Time             | Output       | Fieldbus.OUT[3]  | Running time of servo motor          |
| C01U1_ActualVel        | Output       | Fieldbus.OUT[4]  | Current speed of servo motor         |
The input and output are based on the motion controller DHR41B.

### TABLE 2. COMMUNICATION CONTENT OF THE VARIABLE 'C01U1-SET-CMD'.

| Address | Variable name       | Meaning                                      |
|---------|---------------------|----------------------------------------------|
| Bit0    | C01U1_BreakOpen     | Brake flag                                   |
| Bit1    | C01U1_HomingStart   | Start flag of auto mode                     |
| Bit2    | C01U1_JogForward    | Forward rotation flag of servo motor         |
| Bit3    | C01U1_JogReverse    | Reverse rotation flag of servo motor         |
| Bit4    | C01U1_PosStart      | Start flag of positioning mode               |
| Bit5    | False               | False                                        |
| Bit6    | False               | False                                        |
| Bit7    | False               | False                                        |
| Bit8    | C01U1_ErrClr        | Fault reset                                  |
| Bit9    | False               | False                                        |
| Bit10   | False               | False                                        |
| Bit11   | False               | False                                        |
| Bit12   | False               | False                                        |
| Bit13   | False               | False                                        |
| Bit14   | C01U1-Mode1         | High flag of servo motor working mode        |
| Bit15   | C01U1-Mode0         | Low flag of servo motor working mode         |

Figure 3. The design idea of reproduction process. position coordinates saved to the first array are the same as the previous position coordinates in the array. Then find the last identical position coordinate, and assign the value of the time point of the
position coordinate to the variable 'Special_point', which means that the reproduction process starts to this moment, and the motor is in a stationary state or a swing state.

In the third array, the time point when the swing starts and the time point when the swing stops are assigned to the variables 'OscStart_point' and 'OscEnd_point', respectively. When the value of 't' is equal to the value of 'OscStart_point', the motor starts to swing. When the value of 't' is equal to the value of 'OscEnd_point', the motor stops swinging. The time point of changing the swing amplitude is obtained in the fourth array, and the time point is assigned to the variable 'Change_point'. When the value of 't' is equal to the value of 'Change_point', the changed swing amplitude is sent to the motor driver, then the reproduction processes of the motor swing state are completed. When the value of 't' is equal to the value of the 'Special_point' variable, the algorithm reverses the state of the variable 'SamePoints_flag' and the variable 'UnlikePoints_flag' and looks for the next value of the variable 'Special_point'. The idea is as follows: if the motor is in a rotating state, the position coordinates stored in the first array are different from the previous position coordinates in this array, then the algorithm finds the last different position coordinate and assigns the value of the time point at this position coordinate to the variable 'Special_point', which means that the motor is in a rotating state. At this time, this position coordinate is sent to the motor driver as the end position coordinates of the motor rotation, and the speed of the corresponding time point in the second array is obtained as the rotation speed, so that the reproduction processes of the motor rotation state is completed.

5. EXAMPLE VERIFICATION
The algorithm designed in this paper is run on a hardware experimental platform to test the performance of the algorithm. Through the motion controller DHR41B development environment 'MOVITOOLS MotionStudio', the motion trajectories of the motor are tracked, and the experimental results are obtained. The experimental teaching process is shown in Fig. 4. After the teaching process is finished, the algorithm designed in this paper is run to reproduce the motor trajectory saved during the teaching process. The experimental result is shown in Fig. 5. The red curve is the motion trajectory of the motor during the teaching process, and the grey curve is the motion trajectory of the motor during the reproduction process.

![Figure 4. The teaching process of experiment.](image1)

![Figure 5. Comparison of trajectories.](image2)
It can be seen from Fig. 5 that the motor trajectory of the reproduction process and the motor trajectory of the teaching process are basically consistent, and the error is small, which proves that the algorithm performs well.

6. CONCLUSION
A "teaching-reproduction" algorithm of the servo motor is designed and implemented in this paper. The motor driver is run in the motion controller DHR41B, the trajectory of the motor during the teaching process is saved to the main controller RFC 470, the "teaching-reproduction" algorithm is run, and the teaching process of the servo motor is completed through PC. Profinet communication protocol is used as the communication protocol between controllers and the communication can be completed successfully after agreeing on the message format. The results of multiple experiments show that the motion trajectory of the teaching process and the reproduction process are highly consistent in time and space and the error is small, which proves that the algorithm has good performance.

ACKNOWLEDGMENT
The author would like to thank Mr. Zhang and Mr. Kong for their guidance on thesis topic selection and all the anonymous reviewers for their constructive comments and suggestions which are helpful to improve this paper.

REFERENCES
[1] Kang, X. J., Zhao, B. J., Long, B., “Design of Sand-Blasting lifting equipment,” Advanced Materials Research. vol. 1006, pp. 211-216, 2014, in press.
[2] Yang, D. Y., Yin, Z. C., Li, J. Y., Li, Y., “Research on process of automatic Sand-blasting robot,” Modern Coatings and Painting. vol. 19, pp. 62–66, 2016, in press.
[3] Jin, H. G., Wu, Y. F., “Teaching reproducible arc welding robot——discussion on principle and function,” Machinery Industry Automation. vol. 3, pp.31-37, 1981, in press.
[4] Jie, Y. H., “Design of embedded industrial robot based on PLC servo control,” Proceedings of 2016 5th International Conference on Environment, Materials, Chemistry and Power Electronics. vol. 1, pp. 25-29, 2016, in press.
[5] Dias, A. L., Sestito, G. S., Brandao, D., “Performance analysis of Profibus DP and Profinet in a motion control application,” Journal of Control, Automation and Electrical Systems. vol. 28, pp.86-93, 2017, in press.
[6] Isaías, G., Antonio, C., Godoy, M. C., “Fuzzy controller based on PLC S7-1200 - application to a servomotor,” Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics. vol. 1, pp. 156-163, 2014, in press.