THE DESCENTRALISED CONTROL OF AN ARTICULATED ARM ROBOT

D. PUIU[1] **C. SULIMAN**[1] **F. MOLDOVEANU**[1]

Abstract: *Due to the quick evolution of manufacturing processes, the demand for more flexible automation systems is on the rise. To answer these requirements, distributed motion control architecture based on intelligent drives tends more and more to replace the traditional solutions. This paper presents a local intelligent drive that controls the position of an articulated arm robot. The driver receives the spatial coordinates of the gripper, via a CAN network, and synchronise the motors to move the arm to the desired position.*

Key words: *distributed intelligence, motion control, local intelligent drive, articulated arm robot.*

1. Introduction

Manufacturing companies in the beginning of the 21st Century face frequent, unpredictable market changes. These changes include frequent, rapid introduction of new products, changes in process technology, changes in product demand and mix. To stay competitive, manufacturing companies must possess a new type of manufacturing system that is responsive to all these market changes [12]. This call for reliable and mobile production units that can be easily rearrange in order to coop with unpredictable and frequent market changes [8].

This trend has driven automation users to require flexible automation systems that can be easily modified or upgraded in order to sustain a long term competitive position. In response to these demands, distributed motion control architecture tends more and more to replace the traditional centralized control architecture based around a single host controller such as a computerized numerical control or a computer motion control board [10]. Rather than including all the control tasks in the central controller, the distributed control architecture is based on fieldbusses communication and Digital Signal Processing technology so that decentralized control tasks can reside in the intelligent servo drives [2], [5].

A distributed control solution not only implies the spreading of intelligent modules on the system, it needs a set of instruments and specific mechanisms which facilities the efficient exploitation of the distributed resources of the system [8]. These instruments must allow the engineer to design a distributed control system without tacking care at the distributed resources and at the problems generated by the communication protocols. In this way he can concentrate only at the specific control problems of the automation [6].

Because of the fieldbusses communication facilities and the digital drives technology, the complete control system can be easily re-parameter set for a new product without any hardware modifications [2], [5]. The process can also be easily modified by

---

[1] Dept. of Automation, *Transilvania* University of Braşov.

adding or removing control components without major modifications in the control system [3].

The local intelligent drive presented in this paper controls the position of an arm robot with four degrees of freedom plus a gripper. The local drive also has CAN fieldbus communication facilities, so it can be used in any flexible automation system without major modifications. This is possible, because the master who coordinates the whole system will only have to know the communication protocol of the local drive. After that it will only have to send the position in space of the gripper and the local intelligent driver will manage all the tasks in order to move the gripper to the required position.

2. Control Architectures

There are two general control architectures: centralised and distributed control. For the centralised model, the control systems are based around a central programmable logic controller (PLC) consisting of a central processing unit card and a series of input output boards all connected directly in a chassis [11]. In this case the PLC has to execute all the tasks, and if the process that the PLC has to control is complex, than the PLC's program will be a very complex one and very hard to debug it.

The alternative scheme is to use distributed control architecture, where the overall system control is partitioned into individual programs on each nod [9].

The industrial distributed systems are based on OSI standard for network protocols [5]. It resulted a hierarchic architecture which has its components grouped on four layers as is presented in Figure 1.

Each component from the system receives commands from the devices from the superior layer and sends command to the devices from the inferior layers [5], [7], [9].

The highest layer is the management which ensures a graphic user interface; the downloading and uploading of programs and parameters; the monitoring of the stocks and analysis of the working times [5], [7].

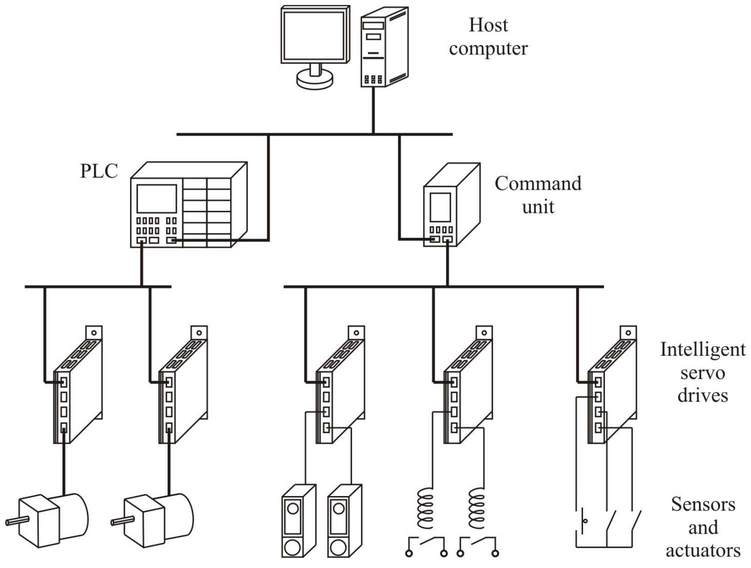


Fig. 1. *Distributed control architecture based on layers* [1], [4], [11]

The next layer is the control process where the execution times are a critical aspect. The intelligent units from this layer have the following main tasks: the generation of the trajectory and the coordination and the synchronization of the axis [5], [7].

The Programmable Logic Controllers (PLC) or the microcontrollers from the axis control layer receive motion trajectory for the actuator and executes it [5], [7].

The last layer contains sensors and actuators and it is close to the hardware of the automation [5], [7].

These new solutions are much more effective than the centralised ones in terms of wiring cost reduction, setup and diagnostic facilities, thanks to the serial bus communication [5].

The fieldbus is the critical component, of distributed control architecture, and some key features like synchronization, update rate or communication profile determine the whole system performances. The industrial network with CAN protocol can be used for a multi-axis motion control system because of its high speed, high reliability and low cost. The CAN serial bus is widely used in the automation and automotive industries; so, the hardware can be implemented at low cost [11].

3. The Model of the Robot

In Figure 2 is presented the model of the robot, which is actuated with five servos. Because the servos have their own control loops, including the position control loop, the local intelligent drive has only to generate the appropriate trajectory for the servo.

One servo controls the rotation of the articulated arm in X-Y plane, which is the same with the plane where is placed the robot. Three servos move the arm of the robot in the plane Z-W and the last servo is used to control the position of the gripper. The W axis represents the axis from the X-Y plan where the first servo positions the articulated arm.

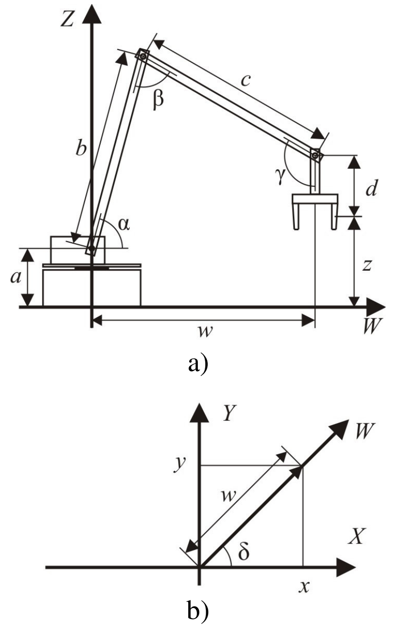


Fig. 2. *Robot model: a) arm plan view; b) rotation base plan view*

To simplify the model will be considered that the gripper of articulated arm will always be perpendicular on the surface of the X-Y plane.

From the Figure 2a and Figure 2b results easy the direct geometric model of the articulated arm:

$$\begin{cases} x = w\cos\delta, \\ y = w\sin\delta, \\ z = a + b\sin\alpha - c\cos(\alpha + \beta - 90) - d, \end{cases} \tag{1}$$

where x, y and z represents the position of the gripper, a, b, c and d are the dimensions of the robots arms, α, β and γ are the angles between the robots arms, δ is the angle of the robot arm from the X-Y plane and w is the distance from the base of the robot and the projection of the gripper in the X-Y plane, which can be determined from Figure 1a, resulting the following equation:

$$w = b\cos\alpha + c\sin(\alpha + \beta - 90)\,. \qquad (2)$$

Solving the system of equations consisting of the Equations 1, 2 and:

$$w = \sqrt{x^2 + y^2}\,. \qquad (3)$$

Result the inverse geometric model:

$$\begin{cases} \alpha = \arccos\left(\dfrac{w(b - c\cos\beta) - c(z - a + d)\sin\beta}{(z - a + d)^2 + w^2}\right), \\ \beta = \arccos\left(\dfrac{b^2 + c^2 - w^2 - (z - a + d)^2}{2bc}\right), \\ \gamma = 180 - (\alpha + \beta), \\ \delta = \arctan\left(\dfrac{y}{x}\right). \end{cases} \qquad (4)$$

It can be observed that α depends by β. It is preferred to use this equation because it is simpler, otherwise the second equation should be inserted in the first one and will result a very complicated one.

3. The Local Intelligent Drive

The local intelligent drive is designed to generate and synchronise ten pulse width modulated (PWM) signals, even if the articulated arm robot uses only five. It also has a CAN controller which it is used to receive the robot trajectory from the local processing unit. In this way the job of the local processing unit is easier because it only has to monitor the whole system, generate the trajectory of the robot and send the data to the local intelligent drive. The rest of the jobs like the generation of the appropriate PWM signals, the synchronisation of the five axes, are made by the intelligent drive.

The drive also has an ANSI C library with functions for the local processing unit or for other command devices. This library should be included in the program which communicates with the drive and it helps the programmer to create the appropriate messages for the local drive, depending on the trajectory it wants to move the articulated arm robot.

3.1. The Hardware Structure of the Distributed Control System

The distributed control structure of the articulated arm robot is presented in Figure 3. On the base of the structure is the robot which is actuated with five servos. On the upper layer is the local intelligent drive which consists of an 8-bit CMOS microcontroller ATCAN91 and a CAN controller.

The role of the local intelligent drive is to receive the trajectory of the robot from the command unit and to generate the appropriate PWM command signals for the five servos of the articulated arm robot.

On the second layer is the command unit and it also contains an ATCAN91 microcontroller and a CAN controller. The unit receives the trajectory of the articulated arm robot from the host computer, divide it in small segments of line and send them to the local intelligent drive.

The host computer allows the user to easily insert the trajectory of the articulated arm robot and to monitor its activity.

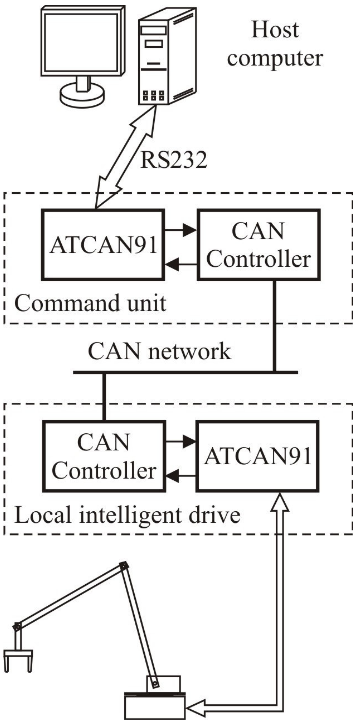


Fig. 3. *The command structure of the articulated arm robot*

3.2. The Axes Synchronisation Strategy

If the command unit needs to move the gripper of the robot to a (*x, y, z*) point from space, it has to calculate, using the system of Equations 4, the angles (α, β, γ, δ) and sends them to the local intelligent drive. The drive when receives the new command, it will generate the appropriate PWM command signals in order to move the gripper from current point to destination point in straight line.

To generate a more complex trajectory, the command unit has to divide the curve in an array of short lines and send them to the local intelligent drive.

In order to modify the speed of the gripper the command unit adds at the end of the command message the time in which the gripper should move, in straight line, from the initial point to the destination point. The variation of this time determines the variation of the speed of the motor.

Because the articulated arm robot has limitation of speed it is useful for the command unit to know, after it determines the angles for each axis, how to calculate the minimum time necessary for the move. This time can be calculated with the following equation:

$$t_{\min} = \max_{k=1}^{5}\left(\frac{\left|u_{ki} - u_{kf}\right| \cdot \tau_k}{90}\right), \tag{5}$$

where $t_{\min}$ represent the minimum time; k is an index for each motor; u_{ki} is the initial angle of the motor k; u_{kf} is the destination angle of the motor k and τ_k is the time that the k motor needs to change it position, at full speed, with 90 degree.

When the local intelligent drive receives a new set of angles for each axis it will gradually change the reference of the each servo in order to generate a linear move for each servo and all of them to start and stop in the same time.

Because the frequency of the PWM signal for each motor is 50 Hz it means that at every 20 ms the local intelligent drive should update the position of the servo. When the driver received a new command and the robot is in transition from the initial point to the destination point the robot will calculate the current command for the servo with the following equation:

$$u_k = \frac{\left|u_{ki} - u_{kf}\right| \cdot \tau}{\tau_c}, \tag{6}$$

where τ is the time that passed from the beginning of the transition; τ_c is the time in which the robot should effectuate the transition and u_k is the angle of the servo at τ moment. If the driver finishes the transition and it does not receive a new one it will maintain the commands for each servo at the same value.

3.3. The Communication Protocol

The most important message of the communication protocol is from the robot command structure to the local intelligent drive and it contains information about the position that each servo should go and the time in which the transition should take place.

As is presented in Figure 4 the command message starts with '#' character than '*k*' represents the servo that should change the position followed by the character 'P' and by the new position of the servo in 'Data *k*' bytes. The three dots represents that the in the message can be added command for other servos in the same way. The command message ends with the character 'T' followed by the time of transition ('Time') and by the end of message character '*'.

#	*k*	P	Data *k*	. . .	T	Time	*

Fig. 4. *The format of the command message*

The tests proved that the communication protocol is very effective because, for a message to be validated it has to pass two filters. First it has to respect the exact format of the message including the end of message character. The second test verifies if it has the exact number of characters. The local intelligent driver determines if the number of character of a message is correct with the equation:

$$n = 7m + 6, \tag{7}$$

where n is the number of characters and m is a natural number bigger than one.

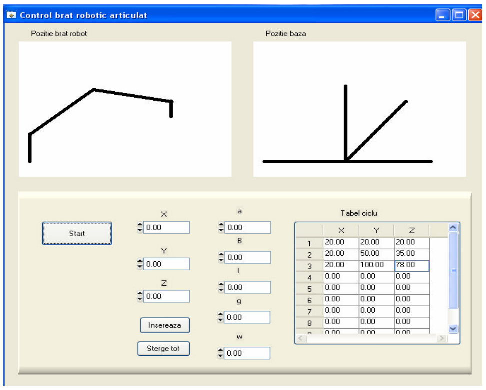
4. Application Software

The software of the drive contains the command strategy, the synchronisation algorithm and the communication protocol. It was designed and simulated in CodeVisionAVR, using a C compiler.

The local intelligent drive has an ANSI C library, which it is useful for the programmer who makes the application for the command unit, because it contains functions for creating the commands and manage the message from/to the local intelligent drive.

The most important functions of the protocol are those who transform from direct geometric model to inverse geometric model, and reverse, using the Equations 1 and 4. Another function calculates the message that should be sent to the local intelligent driver in order to move the gripper of the robot arm to a specific point in the space.

The functions from the library were used to create an application for the computer from the command unit. The interface of the program is presented in Figure 5 and it allows the user to create an array of messages, which is transmitted to the local intelligent driver in order to do a certain move. After that the user can monitor the moves of the articulated arm robot (see Figure 6) on the two canvas controls of the interface. The left one represents the robot in the X-W plan and the other one in the X-Y plan.



X 0.00
Y 0.00
Z 0.00
a 0.00
B 0.00
I 0.00
g 0.00
w 0.00

Tabel ciclu

	X	Y	Z
1	20.00	20.00	20.00
2	20.00	50.00	35.00
3	20.00	100.00	78.00
4	0.00	0.00	0.00
5	0.00	0.00	0.00
6	0.00	0.00	0.00
7	0.00	0.00	0.00
8	0.00	0.00	0.00

Fig. 5. *The interface of the control program*

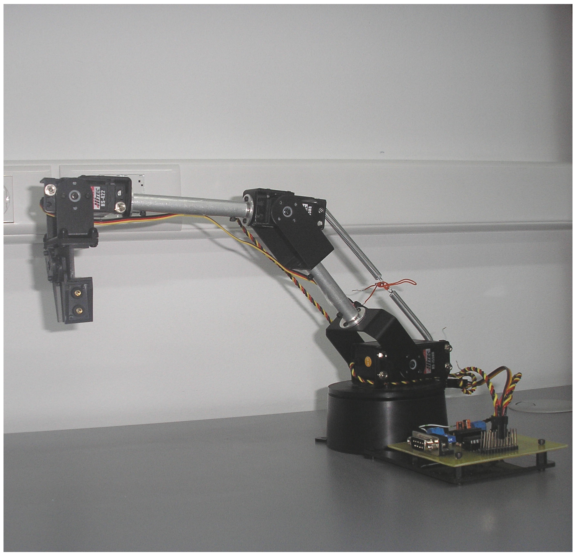


Fig. 6. *The articulated arm robot*

5. Conclusions

The distributed control structure allowed the designer to develop and test separately each module of the control system. As a result of that the time necessarily for designing and debugging was shorter. Further more, because each module has only a few tasks it could be implemented on cheaper microcontrollers.

Even if the local intelligent servo drive is based on a simple 8 bit microcontroller, it can manage all the tasks. This is possible because of the simplicity of the synchronisation algorithm.

The experiment proved that the synchronisation algorithm move the axes in a perfect interpolated motion.

The practical experiments also validated that communication protocol is an efficient one, because there were communication errors but the receiver detected them and asked the transmitter to resend the command. For testing the communication protocol, it has been made an application that allows the designer to easily send messages to the local intelligent drive. In this way the designer could send to the local intelligent drive command messages that contains errors and all of them were detected.

Because all of the simulations and of the practical test were successful, the next stage of development will consist in designing a distributed command structure for the robot. There will be five smaller local intelligent drives on a CAN network, one for each motor, which will control the servos.

Acknowledgement

This paper is supported by the Sectoral Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and by the Romanian Government under the contract number POSDRU/6/1.5/S/6.

References

1. Dumitru, A., Ştefan, I.: *Comanda numerică a acţionărilor electrice de tip sensorless* (*Numeric Command of*

Sensorless Electrical Acuation). Bucureşti. ICPE, 2000.

2. He, F., Tong, W., Wang, Q.: *Synchronization Control Strategy of Multi-motor System Based on Profibus Network*. In: Proceedings IEEE Inter. Conf. on Automation and Logistics, Jinan, China, August 2007, p. 3029-3034.
3. He, F., Zhao, S.: *Research on Synchronous Control of Nodes in Distributed Network System*. In: Proceedings IEEE Int. Conf. on Automation and Logistics, Qingdao, China, 2008, p. 2999-3004.
4. Jouve, D., Bui, D.: *Distributed Intelligence Allows New Dimension in Manufacturing Processes Control*. In: Proceedings 37th Inter. Intelligent Motion Conf. - PCIM'2000, Nuremberg, Germany, June 2000, p. 343-348.
5. Jouve, D., Bui, D.: *CANopen Servo Drives Provides High Performance Motion control*. In: Proceedings 7th Int. Intelligent Motion Conf., Nuremberg, Germany, 2002, p. 1-6.
6. Kim, Y.S., Kim, H.S., Kwon, W.H.: *A Design of a Network-based Robot Control System using FIP*. In: Proceedings 3rd Asian Control Conf., Shanghai, China, July 2000, p. 2229-2234.
7. Lee, K.C., Lee, S., Lee, M.H.: *Worst Case Communication Delay of Real-Time Industrial Swhiched Ethernet with Multiple Levels*. In: IEEE Trans. on Industrial Electronics **53** (2006) No. 5, p. 1669-1676.
8. Molinari-Tosatti, L.: *The ABB Flexible Automation Approach for Developing New Software Tools Dedicated To Advanced Manufacturing and Servicing Technologies*. In: Proceedings IEEE Inter. Symp. Industrial Electronics – ISIE'2002, Milano, Italy, July 2002, p. 46-51.
9. Pacas, J.M.: *Drives 2000 End of the Roadmap? – From the State-of-the Art to the Future Trend*s. In: Proceedings 5th Int. Intelligent Motion Conference, Nuremberg, 2000, p. 41-50.
10. Pereira, C.E., Carro, L.: *Distributed Real-Time Embeded Systems: Recent Advances, Future Trends and Their Impact on Manufacturing Plant Control*. In: Annual Reviews in Control **31** (2007) No. 1, p. 81-92.
11. Rothwell, A.: *The Use of Advanced Fieldbus to Build High Performance Distributed Control Systems*. In: Proceedings 3rd Int. Intelligent Motion Conference, Nuremberg, Germany, 1998, p. 323-334.
12. Schonfeld, R., Franke, M., Hasan, H., Muller, F.: *Intelligent Drives in Systems with Decentralized Intelligence*. In: Proceedings 5th European Conf. Power Electronics and Applications, Brighton, United Kingdom, September 1993, p. 489-494.