MASTER-SLAVE CONTROL OF DYNAMIXEL ACTUATORS

Radovan SIVÝ, Peter GIROVSKÝ
Department of Electrical Engineering and Mechatronics, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, tel. +421 55 602 4220,
e-mail: radovan.sivy@tuke.sk, peter.girovsky@tuke.sk

ABSTRACT

This paper presents the control of actuators Dynamixel from BIOLOID kits and creating graphical user interface. Beginning of this paper is devoted to description of the BIOLOID kits, focused especially for Dynamixel actuator AX-12A. The next parts of the paper describe the comunication of actuators with the control unit and ways how to control the actuators. One of the way for the controls of dynamixel actuators is control in program Matlab. At the end of this paper is a description of the created graphical user interface for control drives with identification of position robotic arm based on Denavit-Hartenberg notation.

Keywords: *BIOLOID, Matlab, GUI, S-functions, Dynamixel, actuator, control*

1. INTRODUCTION

Robotics is now not only part of industrial production, but also becomes a normal part of everyday life. Today, we can meet various robotic systems. Regarding the use of robotic systems in teaching process, it is a suitable alternative use of Bioloid kit. Bioloid kit is a modular system which consists of a control unit, Dynamixel actuators and some type of sensors. From parts of this kit it's possible to assemble many different robotic systems, from simplest to the most complex, for testing of basic principles control of robotic systems.

This paper deals with the possibilities of using of Dynamixel actuators from Bioloid kit, for construction of two robotic arms. Control program for that system, we can create in different environments and programming languages. For this solution we have chosen program MATLAB, which contains necessary tools to create communication and graphical user interface [1], [2].

The control of robotic arms was realized as a master-slave system. Created master-slave system consists of two robotic arms, where primary arm (Master) moves by operator and secondary arm (Slave) copy the move of the primary arm in real time. For easy operating was necessary also to create a graphical user interface, through which the robotic arm will be controlled and this interface will also serve to show the various states of Master-Slave system.

The main task of this work was to create a control program with graphical user interface that will show the endpoint of robotic arm for master-slave system.

2. SYSTEM CONTROL

The idea was to create a robotic master-slave system based on Dynamixel actuators - AX-12A. One of the functions of the graphical user interface that was created by Matlab is also displaying the position of endpoint of robotic arm in a coordinate system x, y, z. Calculation of position of arm endpoint is based on Denavit-Hartenberg notation [3].

By the definition of Denavit-Hartenberg notation, is each joint of robotic arm represented by its coordinate system and four parameters. This defines by which transformation we go to the nearest segment so that new z-axis will be in the axis of rotation of next joint. For Denavit-Hartenberg notation are important four basic parameters (Fig. 1):

d – Distance of the joints in axis z
Θ – Angle of rotation around axis z
r – Length of vertical on secondary axis
α – Angle of rotation of the coordinate system, in axis of rotation of the new joint

Multiplying the partial transformations we get the final transformation of matrix. Partial transformation represents a homogeneous matrix of transformation. Homogeneous matrix has the form:

$$P = \begin{bmatrix} R & r \\ 0^T & 1 \end{bmatrix} \tag{1}$$

R – Matrix of transformation of the rotation
r – Matrix of transformation shift
0^T – zero vector

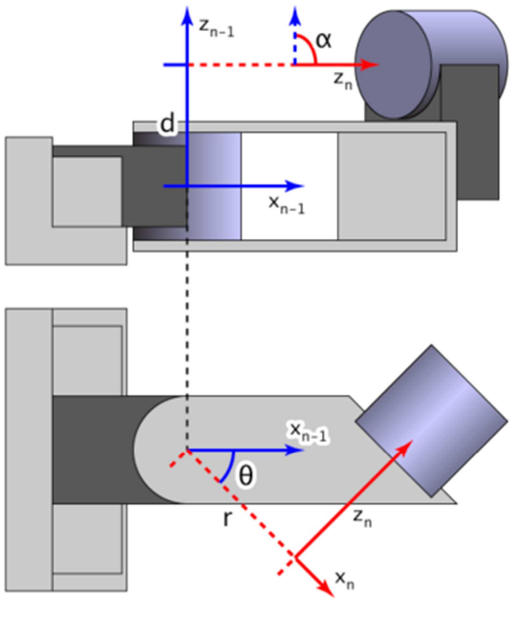


Fig. 1 Parameters of Denavit-Hartenberg notation

According to the rotation axis we can write 3 matrix of transformation of rotation:

$$Trz = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$Try = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \tag{3}$$

$$Trx = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \tag{4}$$

Designed arm, in this case, has five degrees of freedom according to fig. 2 and for each joint of this robotic arm were created transformations, based on Denavit-Hartenberg notation.

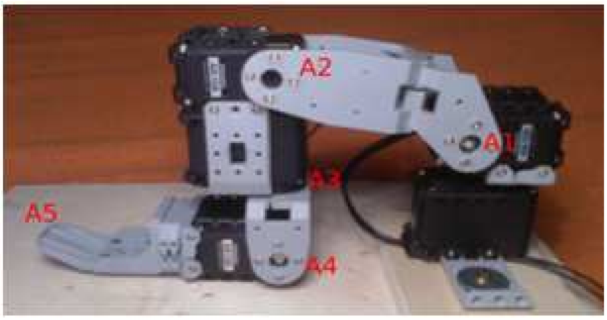


Fig. 2 Scheme of the robotic arm

$$\begin{aligned} A_1 &= Trz(\theta1) \times Tpz(70) \times Tpx(25) \times Trx(-90) \\ A_2 &= Trz(\theta2) \times Tpy(15) \times Tpx(77) \times Trx(0) \\ A_3 &= Trz(\theta3) \times Tpz(0) \times Tpx(50) \times Try(90) \\ A_4 &= Trz(\theta4) \times Tpz(25) \times Tpx(0) \times Try(-90) \\ A_5 &= Trz(\theta5) \times Tpz(0) \times Tpx(110) \times Trx(90) \end{aligned} \tag{5}$$

Multiplying partial transformation and unit matrix A_0, we get a final transformation matrix.

$$P = A_1 \times A_2 \times A_3 \times A_4 \times A_5 \times A_0 \tag{6}$$

On the basis of the final transformation we get the position of the robotic arm in the space [3].

3. MASTER-SLAVE ROBOTIC SYSTEM

Proposal for a master-slave robot system was made for the robotic arms, constructed from Dynamixel actuators AX-12A. Structure of the robotic arm has been chosen so that the more corresponded with the structure of a human arm.

3.1. Dynamixel AX-12A

Dynamixel, from the firm Robotics is a smart actuator, designed for robotic applications (Fig. 3). The actuator module is composed from gear box, DC motor and control circuit. Communication is realized by MultiDrop interface, where is used protocol TTL Half Duplex UART. MultiDrop allows connecting multiple modules to a single data line that has a major impact to the creation of communication with other platforms. Communication consists from construction of digital packets that contains the information of sender, recipient and individual values of variables. In one digital packet we can send several information (position, speed, torque limit, etc.).

This allows efficient to send large data packets to the individual modules at once. Control circuit provides for communication, monitoring and evaluation of the physical variables [4].

Fig. 3 Dynamixel AX-12A

Dynamixel actuator can be controlled with relatively good accuracy, where speed and position resolution is defined to 1024 steps. Despite its compact size it can produce a torque of 1.62 Nm, when power supply is 10 V. On the basis of these properties was chosen this type of actuator to construct master-slave robot system [4].

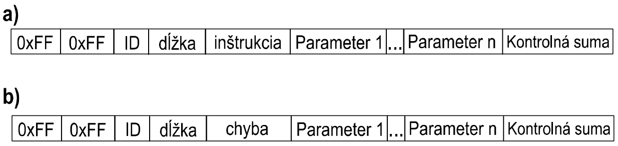
3.2. System control

Control of Dynamixel actuators AX-12A we can realize on PC, by different ways. One of ways how to realize control is to use program written in C language or C-mex language for Matlab. In our case, was the control program written by C-mex language in Matlab.

Dynamixel actuators can communicate with Matlab by using USB2Dynamixel converter. USB2Dynamixel converter makes the conversion from USB signal to one of the next three possible modes of communication:

- TTL logic for servomotors with 3-pin connector
- RS485 logic for servomotors with 4-pin connector
- RS232 for communication over the serial port with control unit, or for wireless communication

In our case it was a half-duplex asynchronous serial communication between Matlab and servomotors by the TTL mode. The communication with the servomotors is realized by the packets, where packet of the instructions, shown in Fig. 4a, is transmitted to the control module of Dynamixel actuator. Status packet, shown in Fig. 4b, is response to the instruction packet, which is sent from the Dynamixel actuator to the control module.



a)

0xFF	0xFF	ID	dĺžka	inštrukcia	Parameter 1	...	Parameter n	Kontrolná suma

b)

0xFF	0xFF	ID	dĺžka	chyba	Parameter 1	...	Parameter n	Kontrolná suma

Fig. 4 a) Packet of instructions, b) Status packet

A. Communication of servomotors with Matlab

Communication with servodrives is realised by the instructions of the “file” type (fopen, fclose, etc.) which concern the work with files (reading. writing, etc.). In special case they can be set to a mode that uses the serial port for communication. In our case, when working in the GUIDE (the GUI development environment), it is suitable to define a global variable handles.s as a serial (obj) and insert parameters for communication like it is shown here:

```
handles.s = serial (COM1);
set (handles.s,'BaudRate',
200000,'Parity','None','DataBits',8, 'StopBits', 1);
```

The variable with defined parameters of serial communication is inserted into the instruction fopen that opens the specific serial port:

```
handles.s = serial (COM1);
```

The functions fwrite and fread enable sending and receiving the data through the serial port. Here under term “the data” we have in mind an instruction packet for the Dynamixel servodrive. Due to calculation of a checksum the packet originally developed in the 16-bit system has to be re-calculated to the 10-bit system. Using the instruction fwrite the packet starts to travel along the communication line. On other side the instruction fread is used to read the state packet. When handling with more servodrives at once it is necessary to develop more similar functions. After finishing the communication it is recommended to close the opened communication channel what is done by instruction fclose (handles.s). When starting the GUI more times a duplication of the serial channels in the programme MATLAB can occur there. To avoid this fault, the instruction fclose(instrfind) is used. It finds all previously opened serial connections that have remained in the memory and closes them.

B. Master-Slave control of servomotors

The basis of the principle of the Master-Slave control is tracking of individual the servomotors of the Master robot arm and sends them to the Slave robot arm (Fig.5).

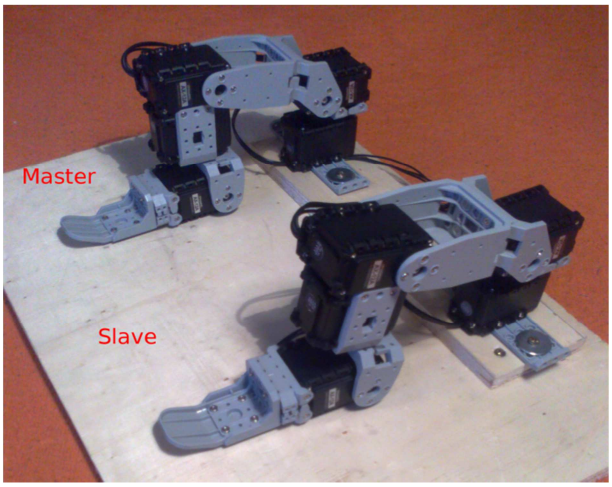


Fig. 5 Master-Slave robotic system

This principles was implemented by the function Real-Time, which makes basic function of the master-slave control and it is a copying in real time. After starting, the program reads data from the graphical user interface, to open the port for communication between the PC and servomotors. If opening of the port is failed, the program writes the error message "Error opening USB2Dynamixel", for release the port and terminate the program.

After successful opening of the port, is set speed of the servo motor to the desired value and starting the cycle. In this cycle, Slave arm copying the movement of Master arm, unless isn't function stopped. In the body of this cycle is the current position loaded into the matrix "PresentPos" and in the cycle using the function NUM_SERVO are identifies the number of actuators and then the actuator ID numbers of the Master robotic arm are saved in the matrix „idm“. The same way for identification of actuators is used for servomotors of the Slave robotic arm, where ID numbers are saved in the matrix „ids“. Consequently, it is created and sent Syncwrite packet to the address in the matrix „ids“. In the case of the previous Master robotic arm, by the first movement was started the program and unlock the arm. For reduce the burden on the communication lines, the program compares actual with previous positions of the Master arm actuators, and if they are equal, the packet is not sent. Principled algorithm of the solving the Master-Slave control robotic arm is shown in Fig. 6.

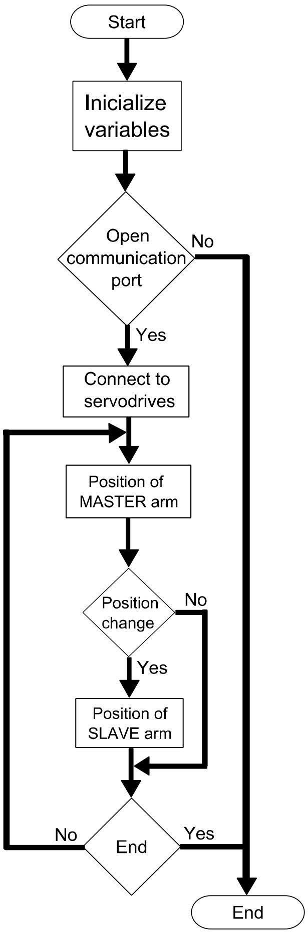


Fig. 6 Principled scheme of control algorithm

4. RESULTS

For manage and verify the correctness of the design structure of the robotic system Master-Slave, was created graphical user interface (GUI). GUI proposal stemmed from the possibility of Real-Time control, sequential control and display of view using Denavit-Hartenberg rotation (Fig.7).

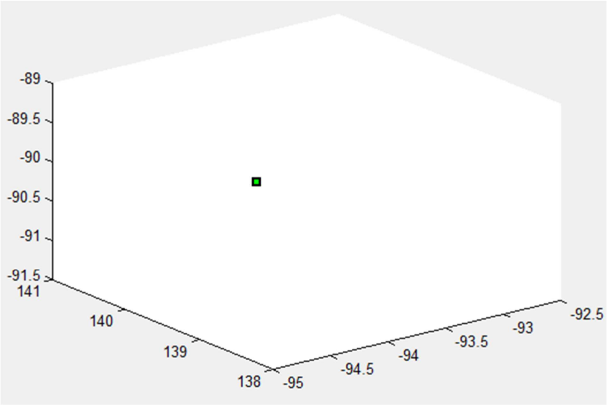
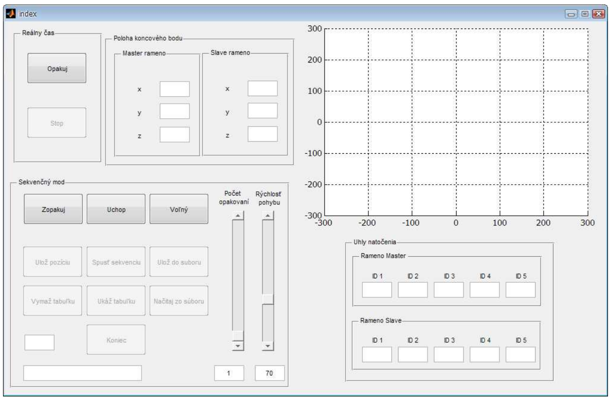


Fig. 7 Display of endpoint using Denavit-Hartenberg notation.

Sequence control was added into the final solution. Sequence control works on the principle of recording sequences of movements and their subsequent repetitions. This makes it possible to form a repeating cycle of movement of the robotic arms. Adding of the sequential type of the control of robotic arms, we wanted to expand the capabilities of the overall master-slave system.



Počet opakovaní 1

Rýchlosť pohybu 70

Fig. 8 Overall Graphical user interface of robotic system.

5. CONCLUSIONS

In this paper was presented the proposal of the master-slave of the control robotic arm, based on Dynamixel actuators AX-12A. The control and graphical user interface for the operation the robot's arms were implemented in MATLAB software. User graphical interface was designed for the master-slave control in real time and for monitoring position of servomotors and position of the end point robotic arms, calculated by Denavit-Hartenberg notation. Control system and graphical user interface, has been extended about sequence control, for increase in the application of robotic systems in terms of learning processes.

ACKNOWLEDGMENTS

Thank you for supporting the project **KEGA-042TUKE-4/2012**.

Paper is the result of the Project implementation: **University Science Park TECHNICOM for Innovation Applications Supported by Knowledge Technology**, ITMS: **26220220182**, supported by the Research & Development Operational Programme funded by the ERDF.

Európska únia
Európsky fond regionálneho rozvoja

Operačný program VÝSKUM a VÝVOJ

UVP TECHNICOM

"We support research activities in Slovakia/This project is being co-financed by the European Union"

REFERENCES

[1] SCOTT, S.: Matlab Advanced GUI Development.

[2] KARBAN, P.: Matlab a Simulink: Calculation and Simulation in the Programms. Brno: Computer Press vyd.2006. ISBN 80-251-1301-9 (in Czech).

[3] The Denavit Hartenberg Convention, Robotic Institute Carnegie Mellon University. Available at: http://pds5.egloos.com/pds/200707/30/01/denavithartenberg.pdf.

[4] Dynamixel AX-12 Manual. Available at: http://www.robotis.com/xe/bioloid_en

[5] SIVÝ, R.: Control of Servodrives for the Robot BIOLOID, FEEaI TU Košice, 2012 (in Slovak).

[6] Developing S-Functions. Available at: http://www.mathworks.com/help/pdf_doc/simulink/sfunctions.pdf

[7] BAŠISTA, J.: Control of BIOLOID servodrives in MATLAB environment. Diploma thesis, FEEaI TU Košice, 2013 (in Slovak).

[8] Bioloid. Available at: http://www.robotis.com/xe/bioloid_en

Received March 30, 2014, accepted May 15, 2014

BIOGRAPHIES

Radovan Sivý was born in Humenné (Slovakia) in 1988. He received his master degree in electrical engineering in 2012. He is currently working towards a PhD. degree at Dept. of Electrical Engineering and Mechatronics, Technical University of Košice, Slovakia. His interest is in the area of robotics and autonomous vehicles.

Peter Girovský was born on 1979. In 2009 he graduated at the department of Electrical Drives and Mechatronics of the Faculty of Electrical Engineering and Informatics at Technical University in Košice. His research interests include artificial intelligence, control systems, robotics.