WEB CONTROL OF THE ROBOTIC ARM

Jakub Arm
Doctoral Degree Programme (2), FEEC BUT
E-mail: xarmja00@stud.feec.vutbr.cz

Supervised by: Zdenek Bradac
E-mail: bradac@feec.vutbr.cz

Abstract: In this paper, control of the 5-DOF stationary robotic arm is presented. This robot uses RC servo-motors which are controlled using NXP LPC1756 at low level. This system is connected to server made up from Raspberry Pi running Java application server Glassfish. The robotic arm is controlled using only web browser using HTML5 and Javascript technology through internet connection. First purpose of this project is to popularize this type of robotic arm platform as a functional model of a true machine for hobbyists and researchers. Second purpose is to propose another approach of economic web based HMI for end user machines.

Keywords: Robotic arm, LPC1756, Raspberry Pi, RC servo, HTML5, Javascript, websocket

1 INTRODUCTION

One of the main scope of the industrial automation is to control robotic manipulators such as crane, arm, etc. These machines are used for many purposes, e.g. manipulation, cutting, or welding. Many hobbyists and researches focus on Cartesian machines of the portal type because of simple basic control [2]. The amount of hobby machine is growing among hobbyists, researchers and also end users. Most of this machines need to install specialized software which does not have to be supported by user OS. Therefore, web based HMI is good and easy solution for them.

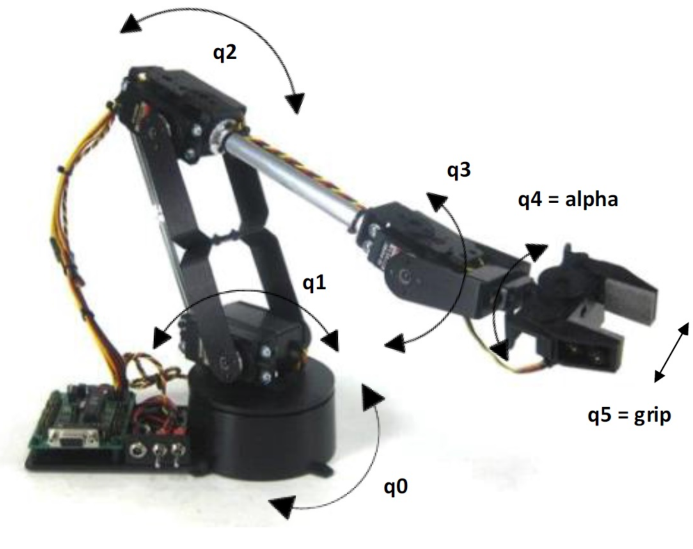

Figure 1: The robotic arm model [7]

2 RC SERVO MOTORS

RC (Remote Control) servo motor is a component for modellers which angle is controlled using DC motor by incoming PWM (Pulse Width Modulation) signal. This motor is controlled in closed loop which is an advantage because of low price of the whole component. A disadvantage is that the control loop tries to regulate the error using maximum motor power. Next disadvantage comes from the model construction. Some parts of the model are heavy enough to maintain the precise servo motor angle. There are some errors in RC servo motor precision caused by its construction, i.e. error in zero angle.

3 KINEMATIC EQUATIONS

The kinematic equations are needed for transformation of the coordinates. While motion is performing, the transformation from the Cartesian system to joint values is performed. This is called inverse kinematics. On the other hand, the operator wants to know the current Cartesian coordinates of the tool. So the current joint values are transformed to the Cartesian coordinates. This is called direct kinematics.

Regarding inverse kinematics, there are three issues. Firstly, the elbow joint value can be computed using two methods - using cosine rule or computing intersection points of two circles. The law of cosine is used in this project because it is computationally lesser demanding than the other method [3]. Secondly, points on Z axis and some other points can be reached by many movements according to base joint rotation. So when this kind of position is needed, the rotation of the whole robotic arm is not performed and the current rotation is used. Thirdly, all equations are computed using floating point data type, and trigonometric functions are computed by CPU using lookup tables and fast numeric approximation methods. All equations in velocity loop (indirect and direct kinematics) are solved within (381 ± 54) μs which is enough to meet deadline coming from velocity loop period 10 ms.

The direct kinematic equations $\overline{P} = f(\overline{q})$ are [7] :

$$
\begin{gathered}
x = cos(q_0)[l_1 + l_3 cos(q_1 + q_2) + l_2 cos(q_1) + l_4 cos(q_1 + q_2 + q_3)][mm] \\
y = sin(q_0)[l_1 + l_3 cos(q_1 + q_2) + l_2 cos(q_1) + l_4 cos(q_1 + q_2 + q_3)][mm] \\
z = l_0 - l_3 sin(q_1 + q_2) - l_2 sin(q_1) - l_4 sin(q_1 + q_2 + q_3)[mm] \\
\alpha = q_4[^\circ] \\
\beta = q_1 + q_2 + q_3[^\circ] \\
g = q_5[mm]
\end{gathered} \tag{1}
$$

4 VELOCITY CONTROL

In this moment, the velocity is controlled according to ramp function. So instead of maximal velocity coming from low level RC servo loop, the velocity is controlled according to basic motion function in software. To overcome using only maximal velocity of the RC servo motor, the RC-motor control signal goes through points in high rate (velocity loop period) to make desired motion at desired speed. The result motion can be smooth because of motor load inertia and motor regulator time constant. When ramp motion function is used, there is a issue caused by jerk in consequence of the acceleration function. To eliminate high peaks in the jerk function, the S-curve motion profile can be used which is computationally more demanding.

The velocity controller is part of motion controller. Motion controller takes care about motion in a two axis plane defined by user command (G-code or manual command). In this moment, the motion control can only do the linear interpolation using one of the simplest algorithm. This algorithm uses also not much of computational resources during motion because it computes a lot of before. The algorithm flow :

- compute velocities in each axis
- find the axis with the longest motion
- recompute motion profile for the other axis (lowering velocity)

5 MCU SUBSYSTEM

The MCU subsystem is based on LPC1756 from NXP. LPC1756 has ARM-Cortex M3 architecture without FPU (Floating Point Unit). Thus every equation with float types has to be done using main processor. The MCU subsystem is communicating with the server via UART using special G-code like protocol and commands for manual control. The MCU takes care about math (motion planning, direct and inverse kinematic) and generating PWM signals for RC servo-motors. The MCU subsystem is located on PCB created on VUT before.

In the MCU subsystem, FreeRTOS is used as a real-time operating system to meet hard deadlines and to make the software easy to extend, e.g. for a command mailbox with a independent command manager. This condition, used MCU and careful written application code ensures deterministic behaviour of this part of the system. This is essential for low level signal generation and when the velocity control loop is created in software.

The MCU subsystem cares also about inappropriate movements. So when a planned moved is out of scope of robot possibilities, the movement is canceled.

6 SERVER SUBSYSTEM

The server subsystem consists of the Raspberry Pi development board. There is Raspbian OS installed. The prebuilt image includes also some components like SSH, USB WiFi adapter driver, Java, etc. The Java application server called Glassfish4 has to be installed. Then the server runs the server side of the web application. This server side provides dynamic (HTML5 + Javascript) web pages and communicates actively with the web client. The received commands are then transmitted over serial line to the MCU subsystem. The whole system is described on Fig. 2.

The server can communicate with clients either using raw TCP or using websockets. Raw TCP needs implementation in some programming language and the handshake is faster than using websockets [9]. On the other hand, websockets can be directly implemented in dynamic web pages which provides comfort for programmers. Websockets have also less overhead than HTTP protocol [8].

Other advantage is that websocket supports message encoding and decoding which can be used for making a trusted connection using some fast cipher method, e.g. AES. Also some other web based proven authentication method can be used before the actual control connection is established. The control connection is then full-duplex client-server type.

Using public internet, low signal connection or connection through many active network components can bring some other issue regarding the latency. If the communication latency with the server is higher than 2 s, the current model information will not be accurate, new user commands will not be executed and the client will be disconnected. The MCU subsystem performs last command and will stay waiting for next command. The server will be waiting for new connection which can be only one at the same time.

Using Java programming language for real-time application is not recommended. On the other hand, this system is considered as firm real-time system and the computational requirements for server subsystem are lesser thanks to MCU subsystem and human perception of real-time. Server source code is multi-platform across OS and Java provides programming comfort which saves costs.

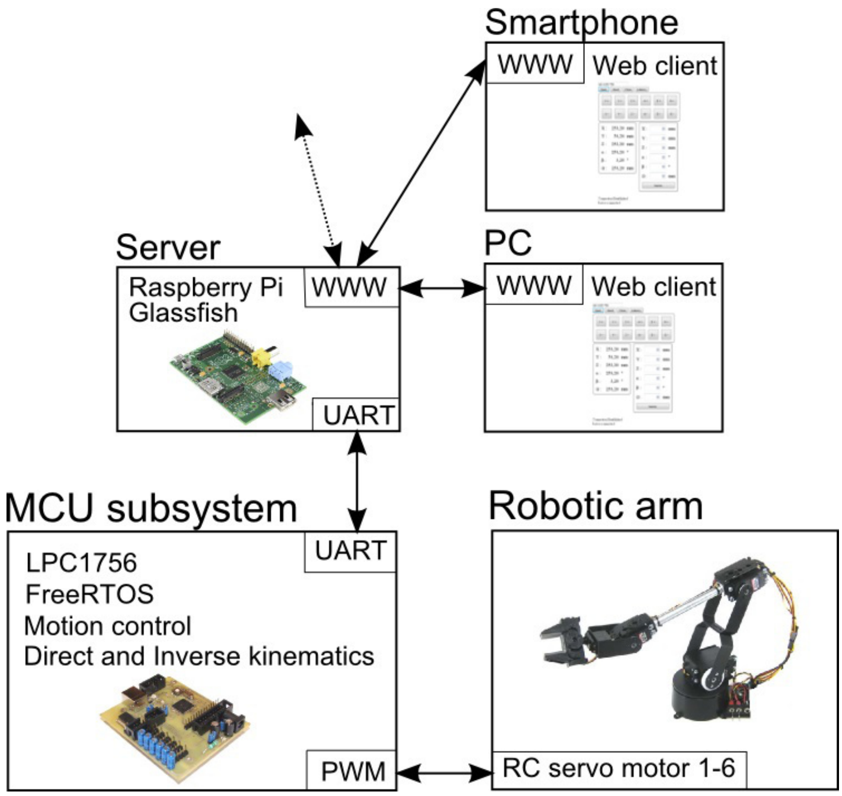

Figure 2: Block diagram of the system

7 WEB CLIENT

The web client is from category of thin clients. It runs completely in the web browser. It uses HTML5, CSS and Javascript technology to make pages dynamic and to communicate with the server nearly real-time. Real-time requirements come from human perception. In this project, a value 250 ms is considered as a soft deadline which is the average time of human reaction [1]. The reaction of the system has been measured using poll method (request followed by answer) with 60 character message as $(2,445 \pm 12,046)$ *ms* on the same network and $(4,550 \pm 29,402)$ *ms* on smartphone when five routers were traced. The highest latencies were $203,156$ *ms* on the same network and $937,687$ *ms* on smartphone. The message poll was done 10000 times per measurement.

This approach is intended to use in hobby low cost end user machines on local network. Some similar project [4] suggests using websockets in industry but only for data acquisition.

In this moment, it supports linear G-code commands and manual control of the robotic arm in the Cartesian coordinates. The current Cartesian axis values and the current joint values are monitored. The web client can run on any device with the web browser supporting Java and HTML5, e.g. PC (Firefox 15 and higher) or smartphone (Android 4.4 and higher).

8 CONCLUSION

This project aims to the firm real-time web control of the robotic arm built of the RC servo-motors using HTML5 and Javascript technologies. Some theory like kinematic equations, RC servo-steering, motion control and real-time control is presented. The proposed approach of web based HMI can be used in any other end user machines. It is economic, multi-platform and client does not have to install any specialized software. In present, user can monitor the robot state, and control it manually or use G-code commands. So it behaves like CNC hobby machine with basic functions. In future, motion control is going to be improved, and 3D model of the robotic arm for the web client is going to be implemented or the system will be extended by camera feedback to make telepresence system. The communication latencies between the web client and the server on the same network have been measured and found sufficient.

ACKNOWLEDGEMENT

The completion of this paper was made possible by the grant No. FEKT-S-14-2429 - "The research of new control methods, measurement procedures and intelligent instruments in automation" financially supported by the Internal science fund of Brno University of Technology.

REFERENCES

[1] Thomas Burger. How fast is realtime? human perception and technology. *www.pubnub.com*, 2015.

[2] Josef Cernohorsky. Zaklady robotiky. *Liberec : Technicka Univerzita v Liberci. Fakulta mechatroniky, informatiky a mezioborovych studii.*, 2015.

[3] Jan Crha, Miroslav Kis, and Jan Mares. Inverse kinematics - tool for robotic manipulator control. 2013.

[4] C. M. Grigorescu, S. A. Moraru, and C. Grama. Industrial software monitoring system extension for mobile devices based on glassfish and phonegap. *Electrical and Power Engineering (EPE), 2012 International Conference and Exposition on*, pages 756 – 761, 2012.

[5] V. Pimentel and B. G. Nickerson. Communicating and displaying real-time data with websocket. *IEEE Internet Computing*, pages 45 – 53, 2012.

[6] Timotej Pochlopen. Robotic arm control. *Brno University of Technology, Faculty of Electrical Engineering and Communication.*, 2012.

[7] Michal Semrad. 6 axis manipulator control. *Brno University of Technology, Faculty of Electrical Engineering and Communication.*, 2014.

[8] K. Shuang, X. Shan, Z. Sheng, and C. Zhu. An efficient zigbee-websocket based m2m environmental monitoring system. *Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on*, pages 322 – 326, 2014.

[9] D. Skvorc, M. Horvat, and S. Srbljic. Performance evaluation of websocket protocol for implementation of full-duplex web streams. *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on*, pages 1003 – 1008, 2014.

[10] Ondrej Sterba. Draughts playing robot. *Brno University of Technology, Faculty of Electrical Engineering and Communication.*

[11] Miroslav Vaclavek. 6 axis robot control in rtos. *Brno University of Technology, Faculty of Electrical Engineering and Communication.*, 2009.