METHOD FOR CONTROLLING MULTI COMMAND TYPE SERVO MOTORS USING MATLAB

Vo Nhu Tien[1], Vo Nhu Thanh[2]

[1]College of Technology, The University of Danang; tienvonhu@yahoo.com
[2]University of Science and Technology, The University of Danang; thanhvous@gmail.com

Abstract - This study proposes a method to control multiple command type servo motors (CTSM) using Matlab. CTSM that was used in this study was a Futaba RS301CR-F3 motor which can be controlled multiply by a set of command lines through RS485 interface using Matlab. In order to setup the connection between Matlab and RS485 interface, an USB serial adapter was used. The advantages of servo command type motor are high speed, accuracy, and built-in feedback system. The control functions for motors are written in Matlab M-file. Then, a Graphic User Interface (GUI) was created to manually control motor's positions and visualize motor's actions. The system has been tested and showed its properly working behavior. As far as we know, so far this is the first work to use Matlab to control CTSMs with only a single RS485 port. This is a new approach for engineers to build a big system with multiple driving mechanisms which use a single RS485 port through USB serial adapter.

Key words - Command type servo motor (CTSM); RS485 port; using Matlab; Graphic User Interface (GUI); control functions

1. Introduction

Motor control system is very important in any electro mechanical system; thus, many researchers have been working on this aspect to improve and perfect the output performance of motors. For realistic application, the decision to select the right motor is far more important because choosing the right motor type for specific application can make the system work more effectively. For example, if an application requires continuous rotation motion then a DC or AC motor type is more suitable; meanwhile, if an application requires position or angle control then a stepper motor type or servo motor type will be more operative [1], [2], [3]. There are many kinds of motor such as DC motor, AC motor, stepper motor, servo motor… Among them, command type servo motor is a newly developed motor type that has many advantages such as high speed motion, accuracy, stability, durability, and so on. Nevertheless, it is complicated to control such motor type because it requires the controller to send a package of control data in multiple bytes sequence to the motors through serial interface port. Most of command type servomotors require the controlled signal to be sent through serial port such as RS 232 or RS 485.

In this study, two Futaba RS301CR-F3 CTSMs were put in investigation [5]. Firstly, the motors were assigned their address using Futaba IDwriter software. Then a function to establish the interconnection between a computer USB port and an RS485 interact was programmed. After that the control functions for motors and a GUI interface were created [4], [6], the system was tested with position control of two motors to verify the control method. Section 2 will describe the system configuration, section 3 will show methodology for controlling multi CTSMs using Matlab comment, then section 4 will show the test result, and finally a conclustion will follow in section 5.

2. System Configuration

Equipment used in this study are a computer with Matlab 2012b installed, a USB serial Adapter from VSCom, two Futaba RS301-F3 CTSMs, Power supply Alinco DM-330MV. Overall system configuration was shown in Figure 1; and the detailed working process will be described in Section 3.

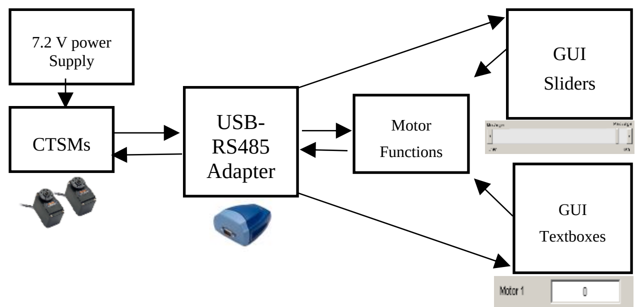

Figure 1. *Overall configurations*

2.1. Futaba Motor

Among very few companies that produce CTSM, Futuba ranks at the most popular manufacturer. Futaba RS301CR, which is a small and lightweight servo specially designed for robots,is about 21g in weight. However, it can generate a maximum torque of 7.1 kg-cm with very fast speed of 0.11 seconds for rotating 60 degree. This motor can be controlled by commands sent from a processor unit through RS485 port. Futaba CTSM also has various kinds of feedback information such as angular position, load, temperature, and warning signal that are embedded in feedback data package as showed in Figure 3. The detailed of the data package is referred to document [2]. CTSM is suitable for building rapid and high accuracy application such as humanoid robot mechanism, rapid molding machine, and high precision CNC machine. Figure 2 shows the pin assignment of connectors for CTSM which has 4 pins corresponding for RS485 A, B data cables, and power supply.

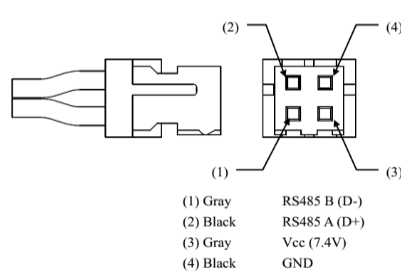

Figure 2. *Pins assignment of connectors for RS301CR-F3*

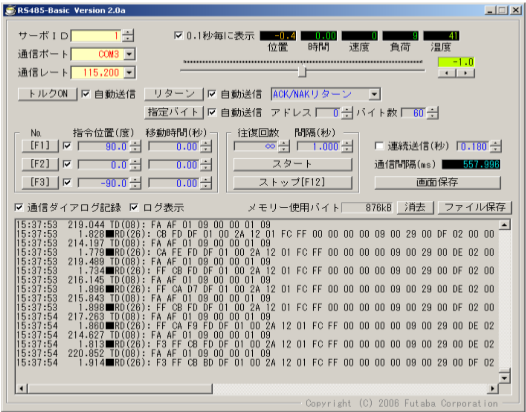

Figure 3. *Using Futaba software for controlling CTSM*

After CTSMs has been connected to power supply and USB serial adapter, it was tested with RS485-Basic software from Futaba to verify their working behavior. The software has sent package data and returned package data as shown in Figure 3. Then, in order to control multiple CTSMs, the ID for each CTSM need to be assigned by RS485 IDWriter software as shown in Figure 4. In this study, two motors were assigned with ID 1 and 2. The next step was to program the motor control function in Matlab M-file to send the controlled package data to two motors.

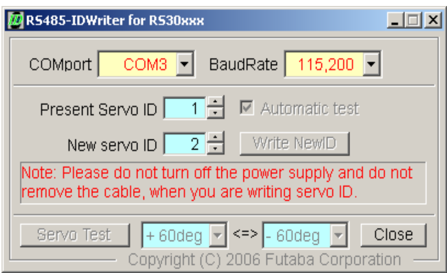

Figure 4. *Assigning ID for motor*

2.2. Matlab Programming

Firstly, the interconnection between computer and motors via USB serial adapter were programmed. Thank to Matlab serial interface support function, this step was very basic, and the programmed coded for the interconnection is shown below.

```
s=serial('com3');
set(s,'BaudRate',115200);
fopen(s);
```

Then the motor controlled data package was sent to control CTSMs using *fwrite* command. The structure for sending a package to control multiple motors is as follows:

Header (4- bytes) – ID – Flag – Address – Length – Count–ID– Data (4bytes)–Sum

**Sum is the XOR logic operation of all previous bytes (From header -> last Data)*

Example for sending header byte using fwrite command is as follow:

```
fwrite(s,[hex2dec('fa')],'uint8') %headerv
```
(1)

The motor control function also contains subfunction to calculate the Xor logic operation for the Sum (last data byte). This subfuction was written in another M-file and was called to calculate the Sum data for the final byte of sending package.

The receiving data from motor was stored to workspace of Matlab using *fget* command. The data of returning package is similar to sending package; and the "Data" bytes in the returning package contains the information of current situation of CTSMs.

3. Methodology

As indicated above, the sending package contain sequence of data in Hex format; however, Matlab serial supports sending decimal format, thus, it was necessary to convert hex data to decimal format as shown in command (1).

The maximum and minimum angles for CTSMs range from -150 to +150 degree. If the set angle is larger than the set angle limit, the CTSM will rotate to the maximum value. The Data byte in sending package has 4 bytes and was sent to CTMSs with two fwrite commands. The working data value that can be sent ranges from -1500 to 1500 in decimal format as corresponding to "05DC" and "FA24" in hexadecimal format. The range from -1500 to 1500 also indicates the maximum and minimum angle rotation of CTMSs with step size of 0.1 degree. It is complicated to use hexadecimal format in realistic application; thus, the author programmed a subfunction in M-file to convert the input angle to hexadecimal format for easier control and visualization. The detailed coding is shown below:

```
    Function [ML,MH] = convertfile(inputangle)
    % input inputangel in decimal first;
    i = 32767;
    %this part to handle directive angle high and low
    data for
    if(inputangle<17&inputangle>-1)
x1 = inputangle;
mc10 = 0;
mc11 = 0;
mc12 = 0;
mc13 = x1;
ML = mc12*16 + mc13*1; %first motor data byte
MH = mc10*16 + mc11*1; %second motor data byte
elseif (num_dc1<256&num_dc1>16)
x1 = fix(num_dc1/16);
x2 = num_dc1-(x1*16);
mc10 = 0;
mc11 = 0;
mc12 = x1;
mc13 = x2;
ML = mc12*16 + mc13*1; %first motor data byte
MH = mc10*16 + mc11*1; %second motor data byte

    elseif(num_dc1<4096&num_dc1>255)
x1 = fix(num_dc1/256);
x2 = fix((num_dc1-(x1*256))/16);
x3 = num_dc1-(x1*256)-(x2*16);
mc10 = 0;
mc11 = x1;
mc12 = x2;
mc13 = x3;
ML = mc12*16 + mc13*1;%first motor data byte
MH = mc10*16 + mc11*1;%second motor data byte
    elseif(num_dc1 < 32767&num_dc1>4095)
x1 = fix(num_dc1 / 4096);
x2 = fix((num_dc1-(x1*4096))/256);
x3 = fix((num_dc1-(x1*4096)-(x2*256))/16);
x4 = num_dc1-(x1*4096)-(x2*256)-(x3*16);
mc10 = x1;
mc11 = x2;
mc12 = x3;
mc13 = x4;
ML = mc12*16 + mc13*1;%first motor data byte
MH = mc10*16 + mc11*1;%second motor data byte
    elseif(num_dc1<0)
e = i - abs(num_dc1) + 1;
x1 = fix(e / 4096);
```

```
x2 = fix((e-(x1*4096))/256);
x3 = fix((e-(x1*4096)-(x2*256))/16);
x4 = e-(x1*4096)-(x2*256)-(x3*16);
mc10 = x1;
mc11 = x2;
mc12 = x3;
mc13 = x4;
ML = mc12*16 + mc13*1;%first motor data byte
MH = mc10*16 + mc11*1;%second motor data byte
   end
```

This code is critical for showing the current angle of CTSMs in GUI interface.

3.1. GUI interface

To test the working behavior of the motors and their real-time data visualization, a Matlab GUI was built as shown in Figure 5. The GUI interface include “interconnection” button to establish the interconnection between CPU and CTSMs by calling the connection subfunction as described above. “Angle 1” and “Angle 2” textboxes contain value of current angle of CTSM 1 and CTSM 2, while “slider 1” and “slider 2” showed actual angle position of CTSM 1 and CTSM 2, respectively. Textboxes’ value was taken from data decimal value divided by 10 as described in previous part to displace the corrected angle value.

The angles of CTSM 1 and CTSM 2 can be controlled either by entering the angle values that the motors need to be reached in textboxes or by adjusting the slider to the designed position. The values in textboxes and position of sliders were also updated accordingly. This means, if a textbox is input with a value, the motor will rotate to that angle and the slider will also move to corresponding position of the textbox value. If the textbox value is out of designed range from -150 to 150 degree, the motor will rotate to either maximum angle position of 150 degree for greater than 150 input value or minimum angle position of -150 degree for smaller than -150 input value. Meanwhile, the slider position was only corresponding to designed value range. Also, the slider step was set at 0.1 incensement as described as the minimum resolution for CTSM used in this study. The maximum and minimum value for slider were set at -150 and 150, respectively. Thus, if the value in textbox is out of corresponding range of slider, the slider will disappear.

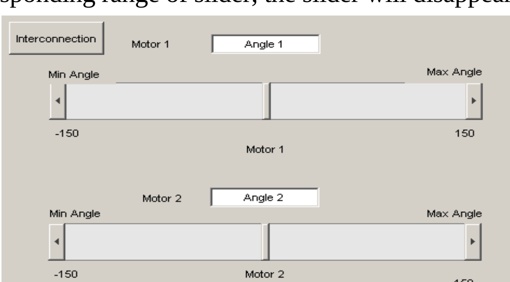

Figure 5. *Matlab GUI interface*

3.2. Experiment procedure

Regarding to Figure 1, the first step is power on the whole system and CTSMs need to be tested with Futaba software for verifying motors’ proper operation. Then, the motors function in M-file was tested to verify it was working appropriately.

After that, GUI was called as shown in Figure 5. “Interconnection” button was clicked to establish the connection between computer and CTSMs. When the connection was established, values of 0, 50, -50, and 150, were put in textbox 1; and values of 0.1, 50.1, 50.2, and 200 were put in textbox 2 in sequential steps. These steps were used to test how fast the motor responded to the input in the textboxes, and how accurate the responses were. Next, the sliders were moved to different locations to test if the motors can respond quickly enough and also the value data in textboxes would update accordingly. Programming flowchart was shown in Figure 6. The detailed results will be shown in the next section.

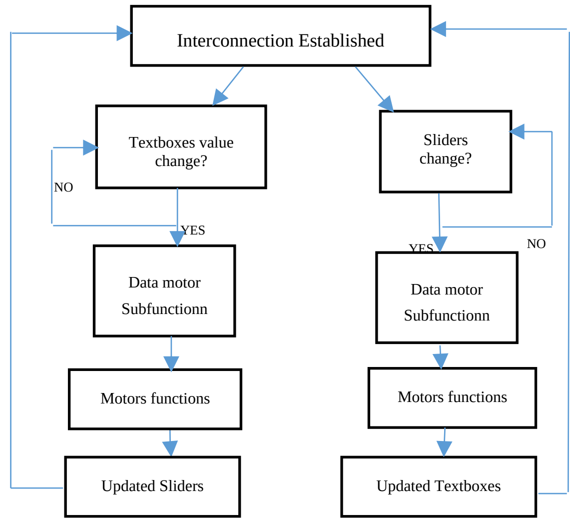

Figure 6. *Flow chart for experiment program*

4. Experiment results

The first test was to move motor 1 to 0 degree position and motor 2 to 0.1 degree position. After inputting the value for the two textboxes and “Enter” key was hit, the two motors seemed not moving at all. However, from the receiving data signal, it showed that motor 2 actually rotated 0.1 degree.

Function Name	Calls	Total Time	Self Time*
testcommandtypeservotest	1	0.480 s	0.003 s
...thworks.toolbox.instrument.SerialComm (Java method)	174	0.436 s	0.436 s
serial.fclose	1	0.250 s	0.000 s
serial.fopen	1	0.162 s	0.001 s
serial.fwrite	42	0.049 s	0.002 s
serial.get	84	0.020 s	0.011 s
serial.isvalid	128	0.009 s	0.003 s
hex2dec	50	0.008 s	0.006 s
serial.igetfield	257	0.006 s	0.006 s
serial.serial>serial.serial	1	0.003 s	0.002 s
parameters	1	0.002 s	0.002 s
serial.isa	127	0.002 s	0.002 s
fliplr	50	0.002 s	0.002 s
formdlmdh	2	0.002 s	0.002 s
...ment.instrument>instrument.instrument	2	0.001 s	0.001 s
...e.icinterface>icinterface.icinterface	2	0.001 s	0.000 s
serial.set	1	0.001 s	0.000 s
deal	42	0.001 s	0.001 s
usejava	2	0 s	0.000 s
serial.class	1	0 s	0.000 s
serial.length	43	0 s	0.000 s
iscellstr	50	0 s	0.000 s

Figure 7. *Timing table in Matlab*

The timing for Matlab program to run two motors was shown in Figure 7. As noticed, the most costly timing was serialcomm, serial open, and serialclose which were built in functions of Matlab; the others functions took less than 50 ms which was pretty fast for controlling motors.

The next step was to examine the working of motor 1 and motor 2 with value of 50 and 50.1 respectively. The result showed that two motors rapidly rotated to 50 and 50.1 angle. Also, the slider 1 and slider 2 on GUI moved to 50 and 50.1 location accordingly. This proved that the program worked properly to control the motors. In the next experiment, the value of textbox 1 was changed to -50 and motor 2 was changed to 50.2. After hitting "enter key", motor 1 rapidly rotated counter clockwise to position -50 degree while motor 2 did not seem to move at all because the changing angle of motor 2 was so small; thus, it was nearly unnoticeable. Then, in order to test what happened if the input value in textbox was out of working range of motor, the value of 150 and 200 was put in textbox 1 and textbox 2, respectively. The two motors quickly rotated to 150-angle position, also the two sliders moved to their right end position. This verified that even though the input value was out of range, the motors still rotated to their end limit.

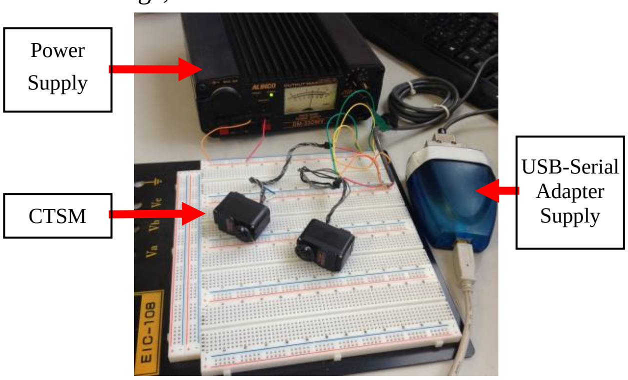

Figure 8. *Experiment setup*

The authors also tested with changing sliders. When one of the slider changed, the corresponding motors also rotated and the value of textbox was updated consequently.

5. Conclusion

A method to control multiple command type servomotors (CTSMs) using Matlab has been successfully implemented. Multiple CTSMs can be controlled by set of command lines through RS485 interface using Matlab. Real-time visualization using Matlab GUI is also applied to test and verify the properly working behavior of motors. The advantages of servo command type motor are high speed, accuracy, and built-in feedback system, which was suitable for many applications requiring high speed and accuracy such as humanoid robot mechanism, rapid molding machine, and high precision CNC machine.

This method that controls multiple CTSMs using only one USB port connected with USB-RS485 adapter can be applied to build a very big system with multiple driving mechanisms. Moreover, using Matlab allowed users to access to a variety of toolboxes and add-in functions of Matlab that would save a lot of time in designing and building an application.

REFERENCE

[1] Jui-Hong Horng, *Hybrid MATLAB and LabVIEW with neural network to implement a SCADA system of AC servo motor*, Advances in Engineering Software (Thomson Reuters), Pg 149-155, 2008, Elsevier, France

[2] Munadi; M. A. Akbar, *Simulation of fuzzy logic control for DC servo motor using Arduino based on MATLAB/Simulink*, International Conference on Intelligent Autonomous Agents, Networks and Systems Bandung, Pg 42-46, August 19-21, 2014, Indonesia

[3] Aranjo, Benetta; Soori, Prashant Kumar; Talukder, Puja, *Stepper motor drives for robotic applications*, International Power Engineering and Optimization Conferencey – Melaka, Malaysia,, Pg 361-366, 2012

[4] Bastian Keller; Alexandre M. S. Costa, *A Matlab GUI for calculating the solar radiation and shading of surfaces on the earth*, Computer Applications in Engineering Education, Wiley Blackwell, Vol 19, Issue 1, Pg 161-170, 2011

[5] Futaba Co., *RS301CR/RS302CD User's Manual Ver.1.14*, *October 2012*

[6] Edward.B.Magrab, S.Azarm, Balakumar.B, James.Duncan, Keith Herold, Gregory Walsh, *An Engineers Guide to MATLAB (3rd Edition)*, Prentice Hall, January 17, 2010.

(The Board of Editors received the paper on 05/03/2015, its review was completed on 05/16/2015)