Development of Teach Pendent based Robotic Arm for Drawing using Mechatronics Approach

Harikrishnan Madhusudanan
Department of Mechatronics Engineering, SRM University, Chennai, India

Article Info

Article history:

Received Aug 27, 2016
Revised Oct 29, 2016
Accepted Nov 13, 2016

Keyword:

Control architectures and programming
Entertainment robotics
Learning and adaptive systems
Physical human-robot interaction
Teach pendent based systems.

ABSTRACT

The main scope of this project is to use mechatronics approach to design and develop a complete humanoid robotic arm which can be used to draw different figures. A mechatronic approach is basically a synergistic combination of mechanical, electrical and control systems such that each and every stage of the product life cycle of the final product gets optimised. So, this approach which is used in this project would ensure that the robotic arm which would be the final product would have its design metrics optimised at each and every stage of its development cycle. The motive of developing this arm is to ensure that the final arm which is produced could be easily attached to the complete humanoid robot and would enable it to draw any kind of complex figures using highly accurate teach pendent mechanism.

Corresponding Author:

Harikrishnan Madhusudanan,
Department of Mechatronics Engineering,
SRM University Chennai, India,
Email: hari_madhusudhanan@srmuniv.edu.in

1. INTRODUCTION

Humanoid robots are becoming quite popular now a day. Humanoids are nothing but human like robots which is capable of exhibiting complex actions which a human is capable of doing. The arm is one of the most important part of the humanoid robot as without it the robot cannot be called a humanoid itself as such. The arm should be capable of carrying out most of the complex activities as such picking objects, holding, writing, drawing etc. Drawing is considered as a vital part of humanoid activities because, it involves a lot of planning and actuation involved unlike other activities. So, in this paper the detailed idea about how a robotic arm can be developed using a mechatronics approach which would state mechanical, electrical and programming aspects of design is covered.

The methods which are used to incorporate the drawing procedure using a robotic arm consider real time scenarios. That is, while teaching a person or a small child to draw an unknown figure, we usually teach them manually by holding their hands and drawing along with them or we just show them how the figure which is to be drawn looks and make them practise the figures until they get perfection. A similar procedure is involved in the programming aspect of our robot too. The basic operation mode which will be used to enable the robot to draw figures is the teach pendent based mechanism in which the user will have to teach the humanoid robot how to draw by holding the robot hand and draw along with it.

This entire paper is structured in the following way. The section II details the mechanical aspects of the robotic arm design and also covers the actuator placement details. The section III covers the implementation of the programming aspect that is using the teach pendant method. The section IV discusses about the conclusion and the future enhancements yet to be made in this project.

2. DESIGN CONCEPT OF ROBOTIC ARM

For a successful actuation along all required directions for drawing, a minimum of three degree of freedom arm is required for drawing simple uniplanar 2-D figures [1]. In this project a three degree of freedom arm has been developed initially in order to test the programming implementation. The overall schematic of the robotic arm is as shown in the Figure 1.

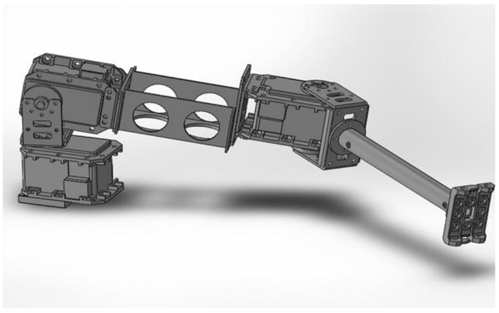

Figure 1. CAD model of the basic robotic arm developed initially for testing the programming implementation.

The base motor has been used for shoulder rotation in the actual humanoid robot. The motor attached to the base motor is used for lifting the end effector which is the drawing equipment away from the drawing plane. However, the actual robotic arm would be much more complex and would have many more added degrees of freedom. This three degree of freedom arm is quite sufficient to implement our project specific purpose and test the program functionality.

The Dynamixel MX-64T servo actuators are used in all the three joints. These servos are chosen specifically because the shaft positional feedback of the servo encoder is quite precise and accurate, which is very essential for teach pendent implementation. Moreover, it can provide a stall torque around 6.4 Nm which also satisfies the actual requirement.

The links are made using aluminium 6061 T-6 grade alloy material of 1.2mm thickness. The actual arm used in the project is as shown in the Figure 2.

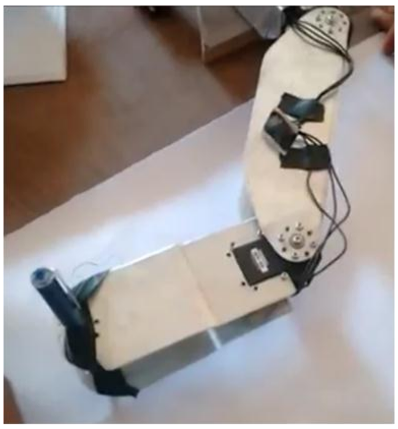

Figure 2. Actual robotic arm attached with a pen as its end effector, used for testing the developed program.

The overall system electrical schematic of the robotic arm is as depicted in the Figure 3. The Raspberry Pi 2 model B has been used as a central processor running Raspbian Wheezy system software with Python libraries installed in it [7]. A standard SMPS is used to provide regulated power supply to various components of the systems. Various input and output components such as keyboard, mouse and monitor display has also been connected to the central processor. The Dynamixel servo motors are connected sequentially as shown in the Figure 3, with one another. One end of the servo port is directly interfaced with the central processor through the USB to Serial converter device [4]. The power supply is connected to the other end port of the servo. In order to understand the data and power transmission within the servo series the exploded view of the internal connection diagram is as shown in the Figure 4.

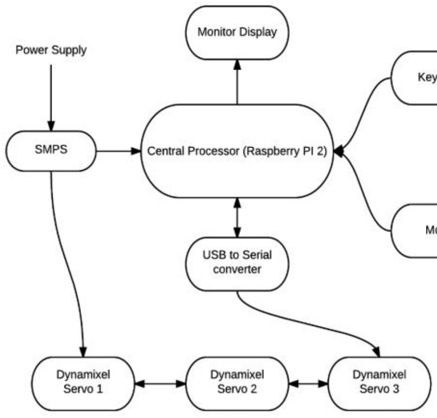

Figure 3. Overall electrical system schematic of the robotic arm

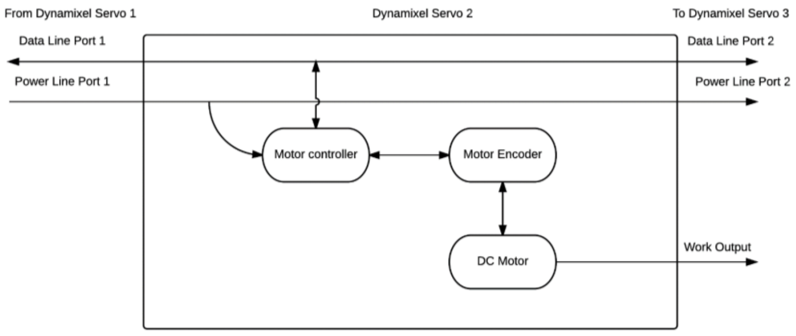

Figure 4. Internal connection diagram of Dynamixel series motor and interface with adjacent motors.

The data is serially transmitted in between the adjacent servo motors and the power supply is provided at one end of the series of motor connection as shown in the Figure 4. So, these smart servos considerably reduces the connection and interfacing in between various motors and moreover the data transmission between various motors is carried out using specific ID provided for each and every motor connected in the series which makes it much more efficient in terms of programming to handle these motors [2].

3. IMPLEMENTATION OF TEACH PENDENT MODE

Using this method, even a complex figure can be made to draw by the robot very clearly and the accuracy mostly depends upon the user and not the robot in this case because, the user is going to hold the robot's hand and teach what has to be drawn and the robot just repeats exactly what ever has been taught [3]. So overall, this method operates in two stages which are as explained in the following.

The first stage of operation is the Read Mode, in which the robot is in the recording state and waits for the user to teach it. As the user holds its arm and moves around the figure to be drawn, it continuously records each and every position of the servo motor shaft and continuously stores it in a database. The position read command is made to execute after every 500 milliseconds interval to ensure that the accuracy is enhanced even though the speed might get compromised.

After the entire figure is taught by the user, the Read Mode is complete. The next stage of operation in this mode is the Repeat Mode in which the taught figure is reproduced again by the robot only this time without the help of the user. The recorded positions are fed sequentially to the motors from the database which contains the recorded positions from the previous Read Mode [6]. An example case in which the map of India is made to be drawn by the robot accurately using this method is as shown in the Figure 5.

Using this method even complex figures which are difficult to be programmed is easily reproduced by the robot. Combinations of several figures are also possible through this method. That is by teaching the drawing procedure for simple figures such as basic shapes and structures, complex figures which are usually a combinations of these basic shapes and structures can be formed using Constructive Solid Geometry (CSG). So, whenever a complex figure has to be drawn by the robot which is usually difficult to be implemented using the visual perception method, this teaches pendent mode is highly preferred as the best method.

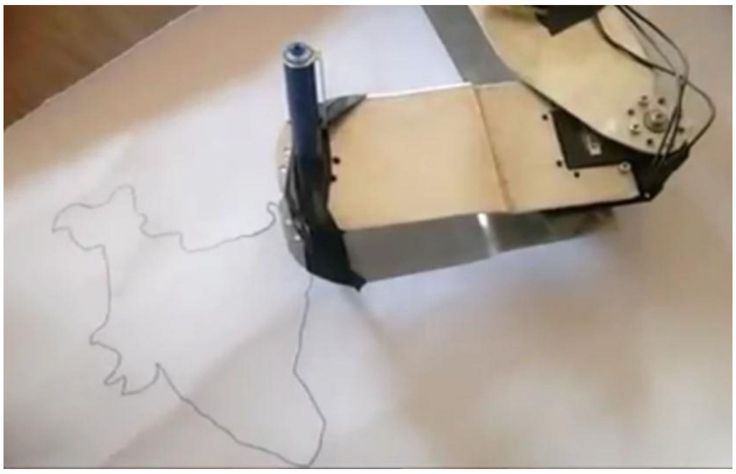
Figure 5. Map of India reproduced using teach pendent mode by the humanoid robotic arm

The entire project has been implemented using Python in a Linux based operating system. The entire flowchart or the sequence of program flow from the beginning for a very basic Read and Repeat operation mode has been depicted in the Figure 6.

In the very beginning the program prompts the user to enter the mode which has to be executed. In the Read mode, the torque of all the motors are switched off and the encoder values are fed back to the system which is continuously stored in a database. The interval after which the values have to be recorded is determined before executing the Read mode. In ideal case, 500 milliseconds delay was found to be sufficient to reproduce a decent figure like as shown in in Figure 5

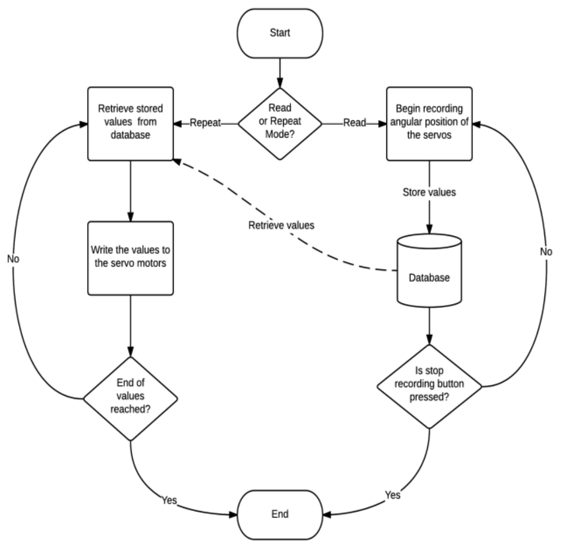
Start
Read or Repeat Mode?
Repeat
Read
Retrieve stored values from database
Begin recording angular position of the servos
Store values
Retrieve values
Write the values to the servo motors
Database
No
No
End of values reached?
Is stop recording button pressed?
Yes
Yes
End

Figure 6. This flowchart depicts the flow of program procedure to implement the Teach pendent mode in the robotic arm

The program code which is used to implement the above flowchart is as explained in the following. The programming language used is Python version 2.7. Some of the prebuilt libraries such as lib_robotis and lib_dynamixel which are utilized for interfacing the Dynamixel MX-64T smart servos with Python with the help of USB2Dynamixel serial convertor are downloaded beforehand.

```
# Program code for implementing read mode
fromlib_robotis import *
d=USB2Dynamixel_Device('/dev/ttyUSB0')
# USB2Dynamixel attached to USB0 port
import time
m1=Robotis_Servo(d,7) # Servo id 7
m2=Robotis_Servo(d,9) # Servo id 9
m3=Robotis_Servo(d,11) # Servo id 11
m1a=m1.read_angle()
m2a=m2.read_angle()
m3a=m3.read_angle()
i=0.0
while True:
        print "m1 ", m1.read_angle()
        print "m2 ", m2.read_angle()
        print "m3 ", m3.read_angle()
        m1a=m1.read_angle()
        m2a=m2.read_angle()
        m3a=m3.read_angle()
        with open('val.txt','a')as f:
                f.write (str(i)+'\n')
                f.write("m1 " +str(m1a)+'\n')
                f.write("m2 " +str(m2a)+'\n')
                f.write("m3 " +str(m3a)+'\n')
        time.sleep(0.5) # Time delay 500 millisecnds
        i = i + 0.5
# End of program code
```

This sample program which is used to implement the Read operation mode, records the angular position values of the individual motor shafts and stores it in a file named val.txt continuously in the format,

```
m1 50.6087
m2 48.6589
m3 89.3658

m1 48.6987
m2 87.3658
m3 89.3214

m1 78.6987
m2 69.1254
m3 78.3695
…
```

The values are stored in this format as long as the end of drawing is sensed or manually stopped by the programmer. After this procedure, the Repeat mode is executed in order to reproduce the same drawing for which the program code is implemented as shown in the following,

```
# Program code for implementing repeat mode
importlib_dynamixel as ld
import math
import time
m1a=[]
m2a=[]
m3a=[]
m=ld.Dynamixel_Chain(dev='/dev/ttyUSB0',baudrate=100000)
with open("val.txt")as f:
        array=f.readlines()
ids = [7, 9, 11]
```

```
i=0
j=0
whilei<len(array):
        if 'm1' in array[i]:
                j=j+1
                x=array[i].find(' ')
                m1a.append(float(array[i][x+1:]))
        if 'm2' in array[i]:
                x=array[i].find(' ')
                m2a.append(float(array[i][x+1:]))
        if 'm3' in array[i]:
                x=array[i].find(' ')
                m3a.append(float(array[i][x+1:]))
        i=i+1
i=0
whilei<j:
        ifi==1:
                time.sleep(0.5)
        angs1=[math.radians(m1a),math.radians(m2a),math.radians(m3a)]
        print m1a, m2a, m3a
        m.move_angles_sync(ids, angs1)
        i=i+1
# End of program code
```

This program basically, retrieves the angular values for each motor from the val.txt file and writes it to the motor continuously. This process continues until the end of val.txt is reached. Thereby it reproduces exactly the same figure which was taught previously by the programmer during the Read mode operation.

4. CONCLUSION AND FUTURE WORK

In this project, the robotic arm was developed to implement teach pendent based drawing algorithm which can easily draw any complex figures without the involvement of conventional complex programming procedures. Moreover, this arm has been developed with a motive to make it as a part of a complete humanoid structure.

Furthermore, with this method as basis a much more complex system which has already been taught basic figures by the programmer can be used to generate and draw much complex structures using different solid modelling techniques such as Constructive Solid Geometry (CSG) is being developed [5].

ACKNOWLEDGEMENTS

This work was supported in part by Department of Mechatronics, SRM University. The components required for the completion of this project was partly funded by SRM Team Humanoid Robotics Research Centre Lab. The guidance from all the faculty members of Department of Mechatronics, SRM University was really helpful.

REFERENCES

[1] Gómez-Espinosa, P.D. Lafuente-Ramón, C. Rebollar-Huerta, M.A. Hernández-Maldonado, E.H. Olguín-Callejas, H. Jiménez-Hernández, E.A. Rivas-Araiza and J. Rodríguez-Reséndiz, "Design and Construction of a Didactic 3-DOF Parallel Links Robot Station with a 1-DOF Gripper," Journal of Applied Research and Technology, Vol 12, June, 2014, pp. 435-443.

[2] M. Pérez Chavarría, and E. Estudillo Zamora, "Development of a data acquisition system for an oceanographic buoy", Journal of Applied Research and Technology, Vol. 7, No. 2, August, 2009, pp. 193-201.

[3] Fei Chao, Zhengshuai Wang, Changjing Shang, QinggangMeng, Min Jiang,Changle Zhou, Qiang Shen, "A developmental approach to robotic pointing via human–robot Interaction," Information Sciences 283, 2014, pp. 288–303.

[4] J. Rodríguez-Reséndiz, G. Herrera-Ruíz, and E. A. Rivas-Araiza,"Adjustable Speed Drive Project for Teaching a Servo Systems Course Laboratory," IEEE Trans. Educ., Vol. 54, No. 4, November, 2011, pp. 657-666.

[5] Faez Ahmed, Kalyanmoy Deb, Bishakh Bhattacharya, " Structural topology optimization using multi-objective genetic algorithm with constructive solid geometry representation," Applied Soft Computing, Vol. 39, February 2016, pp. 240-250.

[6] Guanfeng Sun, Tetsuo Sawaragi, Yukio Horiguchi, Hiroaki Nakanishi, "Knowledge-Intensive Teaching Assistance System for Industrial Robots Using Case-Based Reasoning and Explanation-Based Learning," 19th IFAC World Congress, Vol. 47, Issue 3, 2014, pp. 4535-4540.
[7] Angeles Hoyo, Jose Luis Guzman, Jose Carlos Moreno, Manuel Berenguel, "Teaching Control Engineering Concepts using Open Source tools on a Raspberry Pi board," IFAC Workshop on Internet based control education IBCE 15, Vol. 48, Issue 29, 2015, pp. 99-104.