ME 4773/5493 Fundamental of Robotics
Fall 2016
San Antonio, TX, USA

TRAINING A ROBOTIC MANIPULATOR

Jonathan Sackett
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
jonathan.sackett@utsa.edu

ABSTRACT
A method for training a 3-link robotic manipulator was used to draw a picture on a whiteboard. This is a method that can be used to train any robot in a difficult to map motion. The training was done by manually manipulating the robot along the desired path while recording its' joint angles in Matlab. While the method is simple, the result contained large errors.

1. INTRODUCTION
A robot is a machine that can do the work of a person and that works either automatically or by user control. Robotics is a field of engineering that is concerned with the design, building, and operations of robots. A manipulator is a mechanism that performs an operation, it functions as if it was an arm and is actuated in a skillful manner. An operator can use a manipulator to manipulate objects indirectly. Manipulators can be used to handle hazardous materials, access inaccessible places, and precision actions like welding and medical surgery. While there are different methods to control a manipulator, the experiment below focuses on the process of training a manipulator to perform a task or follow a path. [1, 2]

2. NOMENCLATURE
D-H Table - Denavit-Hartenberg Table
DOF - Degrees of Freedom
PLA - Polylactic Acid
Inverse Kinematics - the use of the kinematics equations of a robot to determine the joint parameters that provide a desired position of the end-effector
Trajectory Generation - iterates the solution to the end-effector position and orientation by a specified time-interval
End Effector - the device at the end of a robotic arm, designed to interact with the environment

3. METHODS
The manipulator used in the experiment is a 6-degree of freedom multi-link robotic manipulator. This means that the manipulator can actuate, or operate, anywhere within in the space around it. The spans out about 1.7 feet of length. It is comprised of 3 modular rectangular PLA links to house servos, a PLA bracket for the gripper, a PLA base for the electronics, and 7 Dynamixel AX-12 servos for actuation. The robot was designed for understanding manipulator theory and for demonstrating the abilities of robotic manipulators.

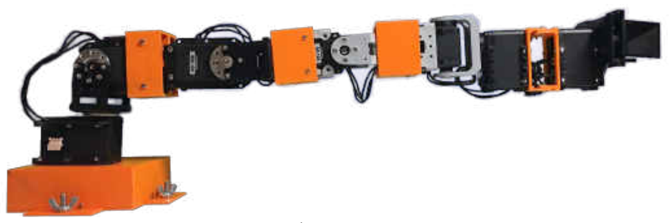


Figure 1

The joints are actuated by the Dynamixel AX-12A servomotors that allow position, speed and torque control. A block diagram of the system is provided in Figure 4. The computer will be transmitting and receiving instruction and status packets respectively via USB port through a half-duplex communication protocol.

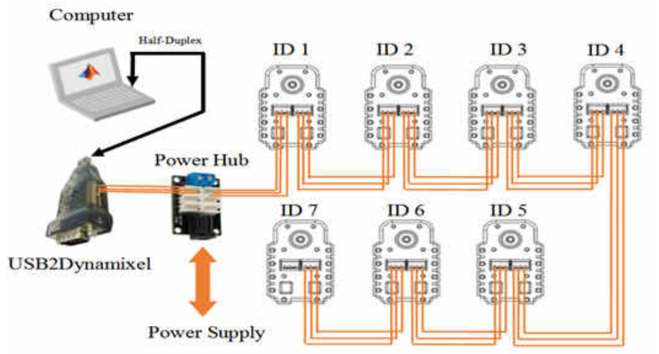


Figure 2

The 3-link robotic manipulator was trained to follow a path by recording the input required to copy a manual movement. Joint angle measurements were collected as the robot was moved manually along the required path to draw a smiley face, see Figure 3. Then, that data was input back so the robot could perform the motion on its' own.

The manipulator follows the parameters outlined in Table , the D-H Table, above. The following list describes the parameters used in the D-H table.

Denavit-Hartenberg Table Parameters

- Link # - number assigned to the link
- Theta - joint angle associated with the servo motor
- d - distance between adjacent x-axis
- a - distance between adjacent z-axis which correspond with link length
- Alpha - angle between adjacent z-axis along the x-axis

The actuation of the robotic manipulator is executed through the use of MATLAB and Simulink.

Trajectory Generation utilizes the techniques established in Inverse Kinematics but iterates the solution to the end-effector position and orientation by a specified time-interval. First, the trajectory is a time function that is described in a three-dimensional Cartesian space. To add, the end-effector is only allowed to actuate in the upper, ¼ of the space due to the limiting angles provided by the servo motors. Next, the solutions are executed iteratively as the resulting joint angles are inputted into the corresponding servomotors and the manipulator's end-effector moves in that specified trajectory.

The motion (Figure 6 below) was created by slowly moving the manipulator along a smiley face that was drawn on a white board. During the motion, each of the joint angle values were recorded in Matlab. As the angles were being recorded, the robot was slowly manipulated by hand to trace the curves of the smiley face. These values were recorded by using a Simulink block diagram. The robot has a custom-built library used to integrate it with Simulink.

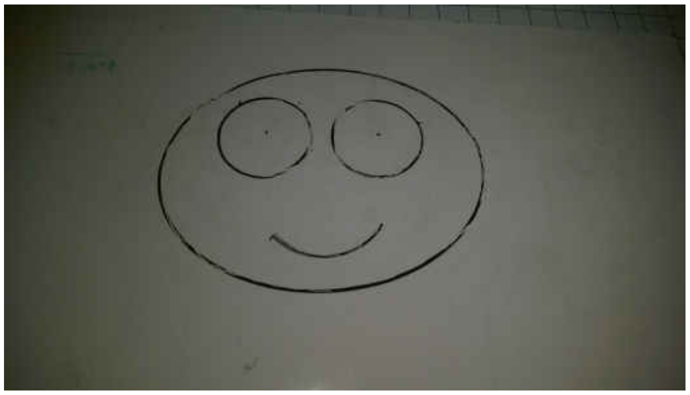


Figure 3

As you can see in Figure 4, the blocks are assembled to return an array of the joint angle positions. The box on the right is set up to loop this operation until stopped.

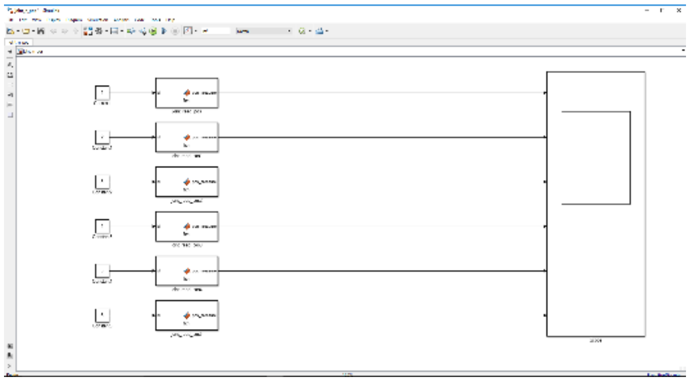
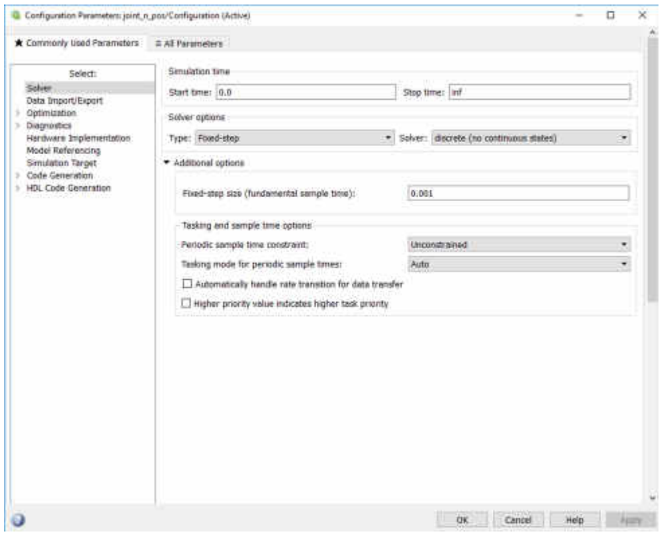


Figure 4


Configuration Parameters: joint_n_pos/Configuration (Active)

★ Commonly Used Parameters | ≡ All Parameters

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
- Hardware Implementation
- Model Referencing
- Simulation Target
- Code Generation
- HDL Code Generation

Simulation time
Start time: 0.0 Stop time: inf

Solver options
Type: Fixed-step Solver: discrete (no continuous states)

▾ Additional options
Fixed-step size (fundamental sample time): 0.001

Tasking and sample time options
Periodic sample time constraint: Unconstrained
Tasking mode for periodic sample times: Auto
☐ Automatically handle rate transition for data transfer
☐ Higher priority value indicates higher task priority

OK Cancel Help Apply

Figure 5

4. RESULTS

In Table 2 in the appendix, the first 50 joint angles of the 6 servos are recorded from the training of the robot. Recording the angle of every servo at .001 second intervals, resulted in 1792 total data points. This many data points per servo ensured that the smiley face output was accurate.

As can be seen in Figure 5, the training method using a small step size produced smooth joint angle curves. However, it can also be seen in Figure 7 that the method still did not produce a very accurate picture. Of note, the circles for the head and eyes of the face did not connect properly. Additionally, every time the marker was applied and removed from the whiteboard, a streak was made that was not a part of the original drawing. Lastly, a systematic error is visible in the jagged curves produced by unsmooth training of the robot.

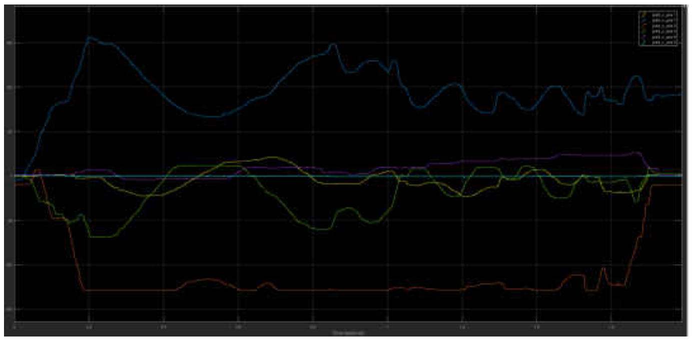


Figure 6

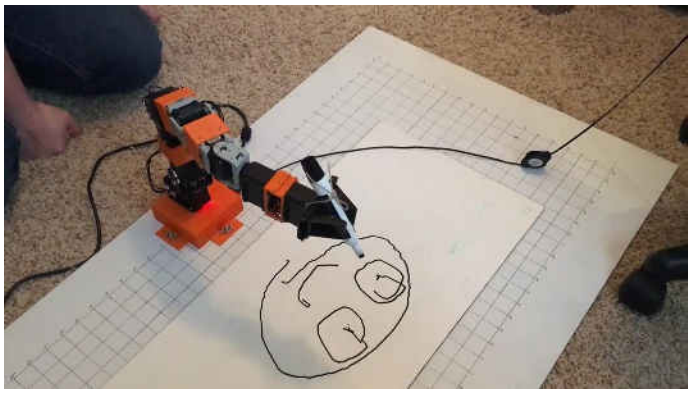


Figure 7

5. DISCUSSION

Even with a fractional step size, the user can still introduce large errors into the system. As notable in Figure 7, inputting the raw data back into the robot is a blunt force method that produces a drawing with great error.

An alternative method would be to use Fourier analyses to curve fit each part of the circle. This method would be much more user extensive as it would require time spent on analyses of the data. However, the method would also reduce errors created during the tracing step. Another alternate method would be to use trajectory motion to draw the face. This method would produce a highly accurate, and highly controllable result.

By using the training method for the robot's movement, the user can skip having to fully understand and apply using manipulator theories. The user would simply need a simple understanding of the process.

ACKNOWLEDGMENTS

Thank you to Sergio Molina for helping to being an extra set of hands for performing the experiment. Also, thank you to Dr. Bhounsule for teaching robotics in a way that makes a student want to continue to dive deeper and apply robotics.

REFERENCES

[1] Alabi, P. et al. (2015) Multi-link Robotic Manipulator, UTSA, 2016.

[2] Niku, Saeed B. *Introduction to Robotics: Analysis, Control, Applications*. Hoboken, N.J: Wiley, 2011.

APPENDIX

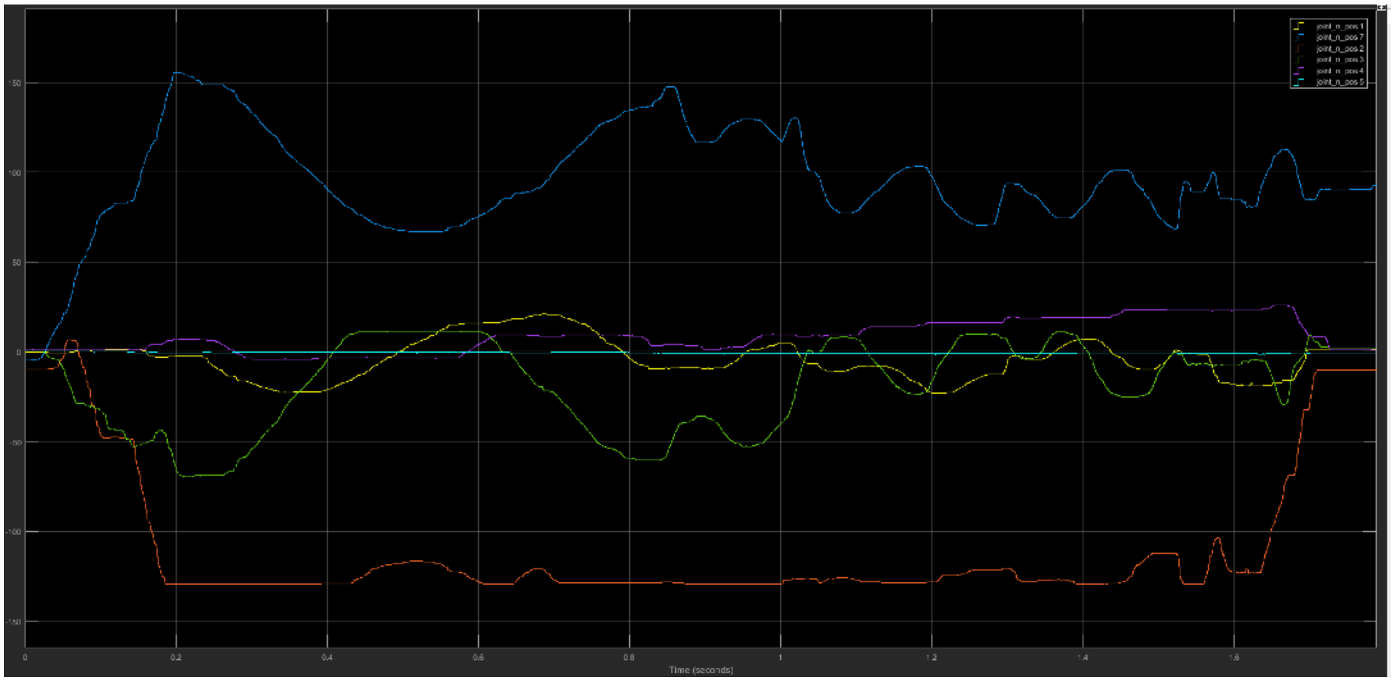


Figure : Plot of Angular Position Values with 100 Hz Sample Time

Script code

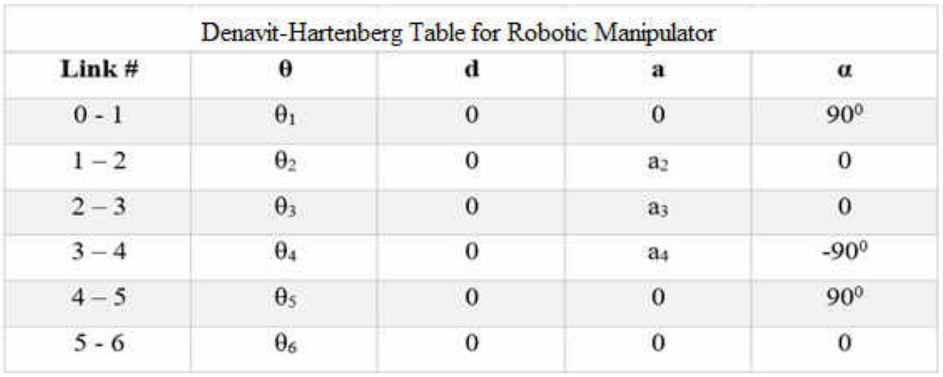
```
clc

loadlibrary dynamixel
calllib('dynamixel','dxl_initialize',3,1);

calllib('dynamixel','dxl_write_word',1,32,40)
calllib('dynamixel','dxl_write_word',2,32,40)
calllib('dynamixel','dxl_write_word',3,32,40)
calllib('dynamixel','dxl_write_word',4,32,40)
calllib('dynamixel','dxl_write_word',5,32,40)
calllib('dynamixel','dxl_write_word',6,32,40)

th1 = Joint_Read_Values.signals(1).values;
th2 = Joint_Read_Values.signals(2).values;
th3 = Joint_Read_Values.signals(3).values;
th4 = Joint_Read_Values.signals(4).values;
th5 = Joint_Read_Values.signals(5).values;
th6 = Joint_Read_Values.signals(6).values;

for i = 1:length(th1)

K1 = 4096/360;
joint_1 = round(K1*th1(i,1)+2048);
calllib('dynamixel','dxl_write_word', 1, 30, joint_1);

K2 = 940/90;
joint_2 = round(K2*th2(i,1)+2130);
calllib('dynamixel','dxl_write_word',2,30,joint_2);

K3 = 950/90;
joint_3 = round(K3*th3(i,1)+2120);
calllib('dynamixel','dxl_write_word',3,30,joint_3);

K = 3.406;
joint_4 = round(K*th4(i,1)+512);
calllib('dynamixel','dxl_write_word',4,30,joint_4);
```

```
joint_5 = round(K*th5(i,1)+512);
calllib('dynamixel','dxl_write_word',5,30,joint_5);

joint_6 = round(K*th6(i,1)+512);
calllib('dynamixel','dxl_write_word',6,30,joint_6);

end
```

Table 1: Servo Angles (first 50 values)

Servo 1	Servo 2	Servo 3	Servo 4	Servo 5	Servo 6
-0.17578125	-4.308510638	-9.852631579	0	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.404255319	-9.757894737	0	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.308510638	-9.852631579	0.29359953	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.404255319	-9.757894737	0	1.174398121	0
-0.17578125	-4.404255319	-9.757894737	0	1.174398121	0
-0.087890625	-4.5	-9.757894737	0	1.174398121	0
-0.17578125	-4.5	-9.757894737	0	1.467997651	0
-0.17578125	-4.5	-9.757894737	0	1.174398121	0
-0.17578125	-4.5	-9.757894737	0	1.174398121	0
-0.17578125	-4.5	-9.757894737	0	1.174398121	0
-0.17578125	-4.5	-9.757894737	0	1.174398121	0
-0.087890625	-4.404255319	-9.757894737	0	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.308510638	-9.757894737	0	1.174398121	0
-0.17578125	-4.212765957	-9.757894737	0	1.174398121	0
-0.17578125	-3.734042553	-9.757894737	0	1.174398121	0
-0.17578125	-3.446808511	-9.757894737	0	1.174398121	0
-0.263671875	-3.063829787	-9.757894737	0	1.174398121	0
-0.17578125	-2.20212766	-9.757894737	0	1.174398121	0
-0.17578125	-1.531914894	-9.757894737	0	1.174398121	0
-0.17578125	-0.957446809	-9.757894737	-0.29359953	1.174398121	0
-0.17578125	-0.478723404	-9.757894737	-0.58719906	1.174398121	0
-0.17578125	0.574468085	-9.757894737	-0.880798591	1.174398121	0
-0.263671875	1.531914894	-9.757894737	-1.174398121	1.174398121	0
-0.439453125	2.20212766	-9.663157895	-1.761597181	1.174398121	0
-0.3515625	3.35106383	-9.663157895	-2.055196712	1.174398121	0
-0.439453125	4.117021277	-9.663157895	-2.642395772	1.174398121	0
-0.439453125	4.882978723	-9.568421053	-2.935995302	1.174398121	0
-0.439453125	6.031914894	-9.568421053	-3.229594833	1.174398121	0
-0.439453125	6.989361702	-9.473684211	-3.523194363	1.174398121	0
-0.439453125	7.659574468	-9.568421053	-3.523194363	1.174398121	0
-0.439453125	8.425531915	-9.568421053	-3.816793893	1.174398121	0
-0.439453125	9.287234043	-9.473684211	-3.816793893	1.174398121	0
-0.52734375	9.861702128	-9.378947368	-3.816793893	1.174398121	0
-0.615234375	10.72340426	-9.189473684	-3.816793893	1.174398121	0
-0.615234375	11.77659574	-9.189473684	-4.110393423	1.174398121	0
-0.615234375	12.63829787	-9.189473684	-4.110393423	1.174398121	0
-0.615234375	13.30851064	-9.094736842	-4.110393423	1.174398121	0
-0.615234375	14.36170213	-8.905263158	-4.110393423	1.174398121	0
-0.615234375	14.93617021	-7.957894737	-4.110393423	1.174398121	0
-0.52734375	15.5106383	-6.915789474	-4.697592484	1.174398121	0
-0.615234375	15.79787234	-5.873684211	-5.578391075	1.174398121	0
-0.52734375	15.9893617	-4.926315789	-6.752789196	1.174398121	0
-0.52734375	16.65957447	-4.357894737	-7.339988256	1.174398121	0
-0.52734375	17.90425532	-3.978947368	-7.927187317	1.174398121	0

Table 2



Denavit-Hartenberg Table for Robotic Manipulator

Link #	**θ**	**d**	**a**	**α**
0 - 1	θ_1	0	0	90^0
1 – 2	θ_2	0	a_2	0
2 – 3	θ_3	0	a_3	0
3 – 4	θ_4	0	a_4	-90^0
4 – 5	θ_5	0	0	90^0
5 - 6	θ_6	0	0	0