Lab 1: Kinematic Characterization of the Lynx (MATLAB)

University of Pennsylvania

Wesley Yee, Shaun Fedrick
MEAM 520
September 23, 2020

1 Methods

For our experimental setup, we used MATLAB to code our forward kinematics calculations, which communicated to ROS running via Gazebo on a local VM running Ubuntu. A 6-DOF robot arm manipulator was programmed to run in Gazebo. Once the code was run in MATLAB on the host computer, the commands for operating the robot were then sent to Gazebo which then actuated the robot to the desired joint variable positions.

The ROS robot was modeled off of an actual physical robot located at Penn, but was not accessible due to COVID-19.

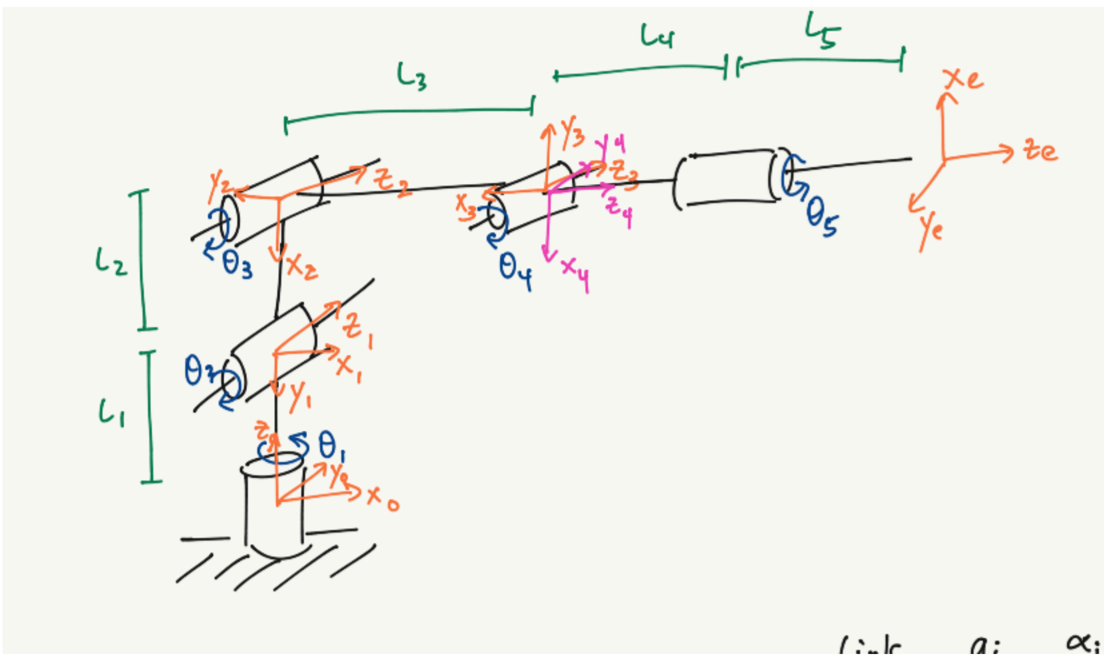

Figure 1: Revised symbolic representation

1.

2. FIX THESE BELOW

$$T_1^0 = \begin{bmatrix} 0 & 0 & 1 & 0mm \\ -1 & 0 & 0 & 0mm \\ 0 & -1 & 0 & 76.2mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$T_2^1 = \begin{bmatrix} 0 & -1 & 1 & 0mm \\ 1 & 0 & 0 & 146.05mm \\ 0 & 0 & 1 & 0mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$T_3^2 = \begin{bmatrix} -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & -132.4588mm \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 132.4588mm \\ 0 & 0 & 1 & 0mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$T_4^3 = \begin{bmatrix} 0 & 0 & -1 & 0mm \\ -1 & 0 & 0 & 132.4588mm \\ 0 & 1 & 0 & 0mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$$T_5^4 = \begin{bmatrix} 0 & 1 & 0 & 0mm \\ -1 & 0 & 0 & 0mm \\ 0 & 0 & 1 & 68mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$T_5^4 = \begin{bmatrix} -1 & 0 & 0 & 0mm \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 84.3755mm \\ 0 & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 14.5255mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

3.

4.

2 Results

3 Evaluation

1.

$$T_e^0 = \begin{bmatrix} 0 & 0 & 1 & 255.325mm \\ 0 & -1 & 0 & 0mm \\ 1 & 0 & 0 & 222.25mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

2. (a)

$$T_e^0 = \begin{bmatrix} 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 180.542mm \\ 0 & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 180.542mm \\ 1 & 0 & 0 & 222.25mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

(b)

$$T_e^0 = \begin{bmatrix} -1 & 0 & 0 & 0mm \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & -180.542mm \\ 0 & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 41.708mm \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

3.

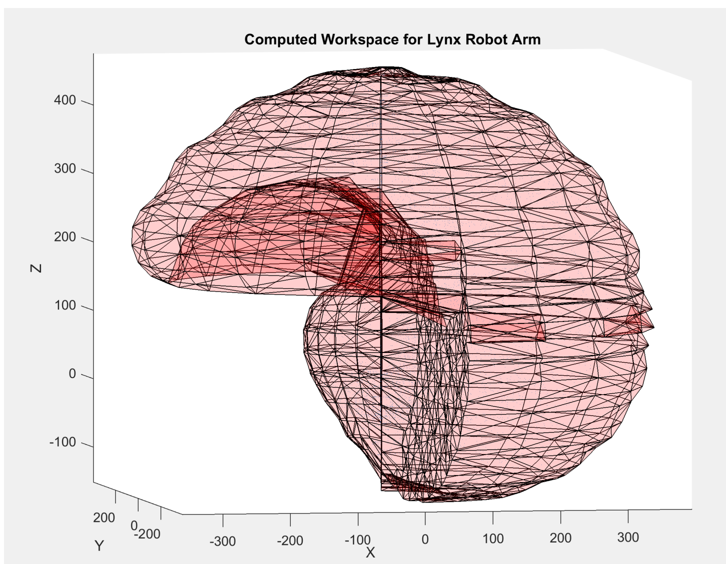

Figure 2: Computed Workspace of Lynx Robot

4.

5. As seen in the code below, the predicted joint positions are almost exactly aligned with the simulated joint positions, aside from a 0.002mm difference in the x position of Joint 3. This is negligible and can be explained due to possible compounded rounding errors in the ROS robot.

However, when examining the T_e^0 matrices, the Simulation T_e^0 records different values of x_0 and y_0 in the end effector frame. We believe that this is due to a mistake in the provided ROS robot code used to calculate Simulation T_e^0. Justification for this conjecture is provided in Problem 1 of the Analysis section.

Listing 1: Outputs of TestFK_Sim.m for $q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

```
Simulation Joint Positions =
         0         0         0
         0         0   76.2000
    0.0002    0.0000  222.2500
  187.3252   -0.0002  222.2500
  221.3252   -0.0003  222.2500
  255.3252   -0.0003  222.2500

Predicted Joint Positions =
         0         0         0
         0         0   76.2000
   -0.0000   -0.0000  222.2500
  187.3250   -0.0000  222.2500
  221.3250   -0.0000  222.2500
  255.3250   -0.0000  222.2500

Simulation T0e =
    0.0000   -0.0000    1.0000  255.3252
    0.0000    1.0000    0.0000   -0.0003
   -1.0000    0.0000    0.0000  222.2500
         0         0         0    1.0000

Predicted T0e =
   -0.0000    0.0000    1.0000  255.3250
    0.0000   -1.0000    0.0000   -0.0000
    1.0000    0.0000    0.0000  222.2500
         0         0         0    1.0000
```

4 Analysis

1. As expected, the results of our evaluation were correct. From the figure below, we use Right-Hand Rule convention to establish that the base frame's red axis is x_0, the green axis is y_0 and the blue axis is z_0. The same axes are applied to the end effector frame, respectively.

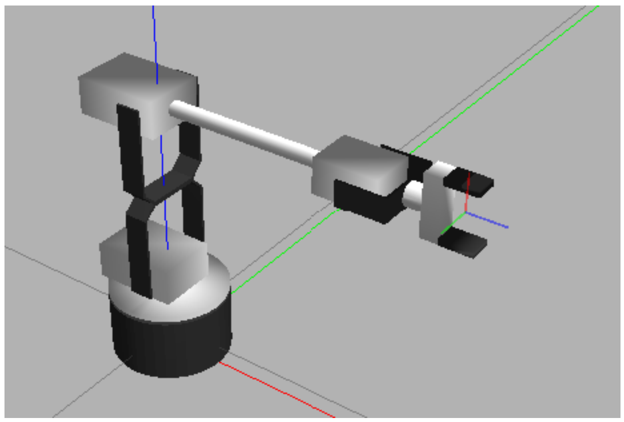

Figure 3: ROS Robot in Zero Configuration

If we use these axes to construct a rotation matrix R by observing each end effector's axis' projection on the base frame's axes, we can compute the following rotation matrix:

$$R = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{10}$$

For example, R states that the projection of the end effector's y axis y_e is in the opposite direction of the base frame's y axis y_0. This value validates the Predicted T_e^0 in Analysis Problem 2.4 and shows that the Simulated T_e^0 is incorrect.

Fortunately, as seen in the same problem, the joint positions were not affected by this error.

2.

5 Appendix

Listing 2: calculateFK.m for $q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

```
function [jointPositions,T0e] = calculateFK(q)
% CALCULATEFK -
%
% DO NOT MODIFY THE FUNCTION DECLARATION
%
% INPUT:
%   q - 1x6 vector of joint inputs [q1,q2,q3,q4,q5,
    lg]
%
% OUTPUT:
%   jointPositions - 6 x 3 matrix, where each row
    represents one
%                    joint along the robot. Each row
     contains the [x,y,z]
%                    coordinates of the respective
    joint's center (mm). For
%                    consistency, the first joint
    should be located at
%                    [0,0,0]. These values are used
    to plot the robot.
%   T0e              - a 4 x 4 homogeneous
    transformation matrix,
%                    representing the end effector
    frame expressed in the
%                    base (0) frame
%
%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% Lynx Dimensions in mm
L1 = 76.2;    % distance between joint 0 and joint 1
L2 = 146.05;  % distance between joint 1 and joint 2
L3 = 187.325; % distance between joint 2 and joint 3
```

```
L4 = 34;       % distance between joint 3 and joint 4
L5 = 34;       % distance between joint 4 and center
   of gripper

%% Your code here
joint1 = q(1);
joint2 = q(2);
joint3 = q(3);
joint4 = q(4);
joint5 = q(5);
jointPositions=zeros(6,3);
T0e = eye(4,4);
T = eye(4);
DH_params = [0, -pi/2, L1, joint1;
             -L2, 0, 0, joint2+pi/2;
             -L3, 0, 0, joint3+pi/2;
             0, pi/2, 0, joint4 - pi/2;
             0, 0, L4+L5, joint5 + pi];

        %Calculate each intermediate homogeneous
           transformation matrix
for link = 1:5
    a = DH_params(link,1);
    alpha = DH_params(link,2);
    d = DH_params(link,3);
    theta = DH_params(link,4);

    % calculate A
    A = createA(a, alpha, d, theta);
    T = T*A;
    jointPositions(link+1,:) = T(1:3,4)';
    if link == 4
        p5=[0;0;L4;1];
        T5=T*p5;
        %jointPositions(link+1,1) = jointPositions(
           link+1,1) + L4;
         jointPositions(link+1,:)=T5(1:3)';
    end
```

```
end

T0e = T;

%
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end
```

Listing 3: computeWorkspace.m for $q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

```
% Fill in this file with your code for Analysis
   question #4.

% # Discretizations for each joint variable
discStep = 9;

%Link Lengths
L1 = 76.2;
L2 = 146.05;
L3 = 187.325;
L4 = 34;
L5 = 34;

%Initialize T matrices, combined them into a single
   matrix to make easier
%to calculate
T = eye(4);
x = [];
y = [];
z = [];

% Compute the end effector Cartesian coordinate for
   each permutation of
% joint variable
for joint1 = -1.4:2.8/discStep:1.4
    for joint2 = -1.2:2.6/discStep:1.4
        for joint3 = -1.4:3.1/discStep:1.7
```

```
            for joint4 = -1.9:3.6/discStep:1.7
                for joint5 = -2:3.5/discStep:1.5
                    T = eye(4);
                    DH_params = [0, -pi/2, L1,
                       joint1;
                                 -L2, 0, 0, joint2 +
                                    pi/2;
                                 -L3, 0, 0, joint3 +
                                    pi/2;
                                 0, pi/2, 0, joint4 -
                                     pi/2;
                                 0, 0, L4+L5, joint5
                                    + pi];

                    %Calculate each intermediate
                       homogeneous transformation
                       matrix
                    for link = 1:5
                        a = DH_params(link,1);
                        alpha = DH_params(link,2);
                        d = DH_params(link,3);
                        theta = DH_params(link,4);

                        % calculate A
                        A = createA(a, alpha, d,
                           theta);
                        T = T*A;
                    end

                    % Append to the x, y, or z
                       vectors w/ each permutation
                    x = [x; T(1,4)];
                    y = [y; T(2,4)];
                    z = [z; T(3,4)];
                end
            end
        end
    end
```

```
end

% Plot coordinates
plot3(x,y,z,'.','MarkerSize',1);
xlabel('X');
ylabel('Y');
zlabel('Z');
hold on
k = boundary(x,y,z,1);
axis equal
trisurf(k,x,y,z,'FaceColor','red','FaceAlpha',0.1);
title('Computed Workspace for Lynx Robot Arm');
```

Listing 4: createA.m for $q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

```
function A = createA(a, alpha, d, theta)
    A = [cos(theta), -sin(theta)*cos(alpha), sin(
       theta)*sin(alpha), a*cos(theta);
        sin(theta), cos(theta)*cos(alpha), -cos(
           theta)*sin(alpha), a*sin(theta);
        0, sin(alpha), cos(alpha), d;
        0, 0, 0, 1];
end
```