Introduction to Robotics

Lab 4b

Shaaf Farooque 08405, Mysha Zulfiqar 08443

Task 4.5 Homogeneous Transformations (5+5+3 points)

(PoE) Use MATLAB's symbolic math toolbox to determine the rigid-motion corresponding to each of the screw axes and the resultant transformation ${}^0T_4$. See below for specific deliverables.

(DH) Use MATLAB's symbolic math toolbox to determine the intermediate homogeneous transformations ${}^0T_1$, ${}^1T_2$, ${}^2T_3$, ${}^3T_4$, and the resultant transformation ${}^0T_4$.

(Common Instructions)

(a) Write a MATLAB script to create symbolic matrices for all the homogeneous transformations listed above. Note that one of the parameters will be a joint variable.

(b) Obtain ${}^0T_4$ by multiplying the previously determined homogeneous transformations in the appropriate order.

(c) Provide expressions for the position and orientation of the end-effector frame with respect to the base frame.

```
syms l_1 l_2 l_3 l_4 l_5
M = [1 0 0 0;
     0 1 0 0;
     0 0 1 l_1+l_2+l_3+l_4+l_5;
     0 0 0 1;
    ]
w1 = [0;0;1];
q1 = [0;0;l_1];
S1 = [w1;cross(-w1,q1)]
w2 = [1;0;0];
q2 = [0;0;l_1+l_2];
S2 = [w2;cross(-w2,q2)]
w3 = [1;0;0];
q3 = [0;0;l_1+l_2+l_3];
S3 = [w3;cross(-w3,q3)]
w4 = [1;0;0];
q4 = [0;0;l_1+l_2+l_3+l_4];
S4 = [w4;cross(-w4,q4)];
syms theta_1 theta_2 theta_3 theta_4
T = simplify(exp(S1,theta_1)*exp(S2,theta_2)*exp(S3,theta_3)*exp(S4,theta_4)*M)
position = T(1:3,4)
rotation = T(1:3,1:3)
```

T =

$$\begin{pmatrix} \cos(\theta_1) & -\sigma_1 \sin(\theta_1) & \sigma_3 \sin(\theta_1) & \sin(\theta_1)\,\sigma_2 \\ \sin(\theta_1) & \sigma_1 \cos(\theta_1) & -\sigma_3 \cos(\theta_1) & -\cos(\theta_1)\,\sigma_2 \\ 0 & \sigma_3 & \sigma_1 & l_1 + l_2 + l_4 \cos(\theta_2 + \theta_3) + l_3 \cos(\theta_2) + l_5\,\sigma_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = \cos(\theta_2 + \theta_3 + \theta_4)$$

$$\sigma_2 = l_4 \sin(\theta_2 + \theta_3) + l_3 \sin(\theta_2) + l_5\,\sigma_3$$

$$\sigma_3 = \sin(\theta_2 + \theta_3 + \theta_4)$$

position =

$$\begin{pmatrix} \sin(\theta_1)\,\sigma_1 \\ -\cos(\theta_1)\,\sigma_1 \\ l_1 + l_2 + l_4 \cos(\theta_2 + \theta_3) + l_3 \cos(\theta_2) + l_5 \cos(\theta_2 + \theta_3 + \theta_4) \end{pmatrix}$$

where

$$\sigma_1 = l_4 \sin(\theta_2 + \theta_3) + l_3 \sin(\theta_2) + l_5 \sin(\theta_2 + \theta_3 + \theta_4)$$

rotation =

$$\begin{pmatrix} \cos(\theta_1) & -\cos(\theta_2 + \theta_3 + \theta_4) \sin(\theta_1) & \sin(\theta_2 + \theta_3 + \theta_4) \sin(\theta_1) \\ \sin(\theta_1) & \cos(\theta_2 + \theta_3 + \theta_4) \cos(\theta_1) & -\sin(\theta_2 + \theta_3 + \theta_4) \cos(\theta_1) \\ 0 & \sin(\theta_2 + \theta_3 + \theta_4) & \cos(\theta_2 + \theta_3 + \theta_4) \end{pmatrix}$$

Task 4.6 FK Function (10 points)

Provide a MATLAB `function [x,y,z,R] = pincherFK(jointAngles)` or `function [x,y,z,R,theta,phi] = pincherFK(jointAngles)` that accepts joint angles of Phantom X Pincher and returns the end-effector position and orientation in the specified order. Make sure to add comments describing the arguments and corresponding units.

```
function [x, y, z, R] = pincherFK(theta_1, theta_2, theta_3, theta_4)
    % Computes the forward kinematics.
    % Inputs:
    %   theta_1 - Joint 1 angle (Base rotation) in radians
    %   theta_2 - Joint 2 angle (Shoulder rotation) in radians
    %   theta_3 - Joint 3 angle (Elbow rotation) in radians
    %   theta_4 - Joint 4 angle (Wrist rotation) in radians
    %
    % Outputs:
    %   x - X-coordinate of the end-effector in meters
    %   y - Y-coordinate of the end-effector in meters
    %   z - Z-coordinate of the end-effector in meters
    %   R - 3x3 Rotation matrix representing the end-effector orientation

    % Link lengths (converted from cm to meters)
    l_1 = 10 / 100;   % Length of Link 1
    l_2 = 4.5 / 100;  % Length of Link 2
    l_3 = 10.7 / 100; % Length of Link 3
    l_4 = 10.5 / 100; % Length of Link 4
    l_5 = 9.5 / 100;  % Length of Link 5 (End-effector offset)

    % Home position transformation matrix (end-effector frame at zero position)
    M = [1 0 0 0;
         0 1 0 0;
         0 0 1 l_1 + l_2 + l_3 + l_4 + l_5;
         0 0 0 1];
```

```
    % Define screw axes in the space frame
    w1 = [0; 0; 1];  % Rotation around Z-axis (Base)
    q1 = [0; 0; l_1];
    S1 = [w1; cross(-w1, q1)];

    w2 = [1; 0; 0];  % Rotation around X-axis (Shoulder)
    q2 = [0; 0; l_1 + l_2];
    S2 = [w2; cross(-w2, q2)];

    w3 = [1; 0; 0];  % Rotation around X-axis (Elbow)
    q3 = [0; 0; l_1 + l_2 + l_3];
    S3 = [w3; cross(-w3, q3)];

    w4 = [1; 0; 0];  % Rotation around X-axis (Wrist)
    q4 = [0; 0; l_1 + l_2 + l_3 + l_4];
    S4 = [w4; cross(-w4, q4)];

    % Compute the transformation matrix using product of exponentials
    T = exp(S1, theta_1) * exp(S2, theta_2) * exp(S3, theta_3) * exp(S4, theta_4) * M;

    % Extract end-effector position
    x = T(1, 4);
    y = T(2, 4);
    z = T(3, 4);

    % Extract rotation matrix
    R = T(1:3, 1:3);
end
```

```
theta_1 = deg2rad(0);
theta_2 = deg2rad(0);
theta_3 = deg2rad(0);
theta_4 = deg2rad(0);

[x,y,z,R] = pincherFK(theta_1, theta_2, theta_3, theta_4);
disp(x);
disp(y);
disp(z);
disp(R);
```

```
     0

     0

0.4520

     1     0     0
     0     1     0
     0     0     1
```

```
theta_1 = deg2rad(0);
theta_2 = deg2rad(0);
theta_3 = deg2rad(0);
theta_4 = deg2rad(-90);

[x,y,z,R] = pincherFK(theta_1, theta_2, theta_3, theta_4);
disp(x);
disp(y);
disp(z);
disp(R);
```

```
     0

0.0950

0.3570

1.0000         0         0
     0    0.0000    1.0000
     0   -1.0000    0.0000
```

Task 4.9 Identifying reachable workspace (10 points)

Use the outlined idea of determining end-effector positions for selected joint configurations (uniform or random) to plot the reachable workspace of our Phantom X Pincher robot arm. Provide an isometric view of the workspace as well as a top-view, i.e. a projection of your workspace onto $X-Y$ plane of your base frame. Remember to mark axes in your plots. What is the maximum horizontal reach according to your identified workspace?

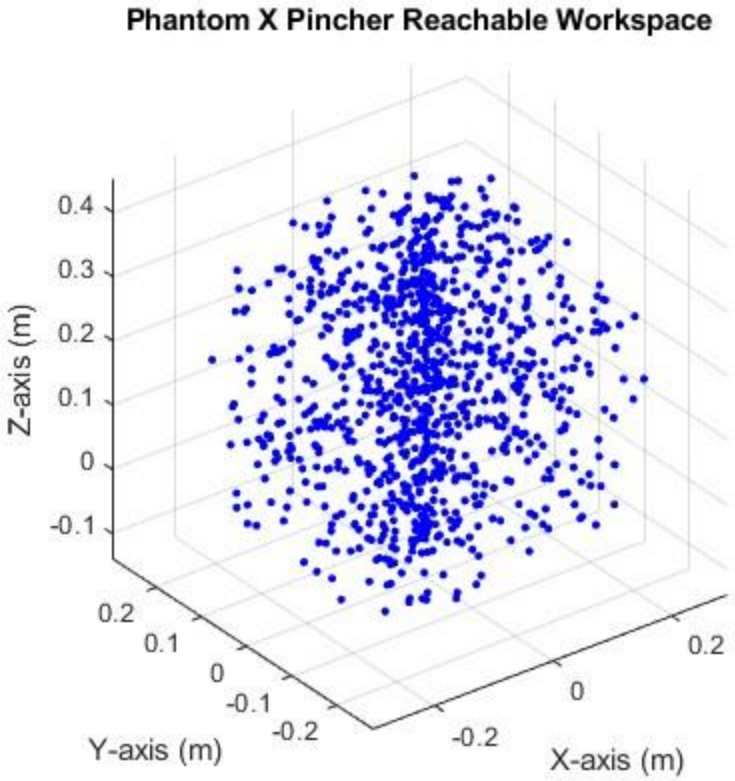
- The MATLAB function `rand` generates uniform pseudorandom numbers in $[0, 1]$. We can generate N samples for a joint angle θ_i, with lower bound θ_i^{min} and upper bound θ_i^{max} using the expression:

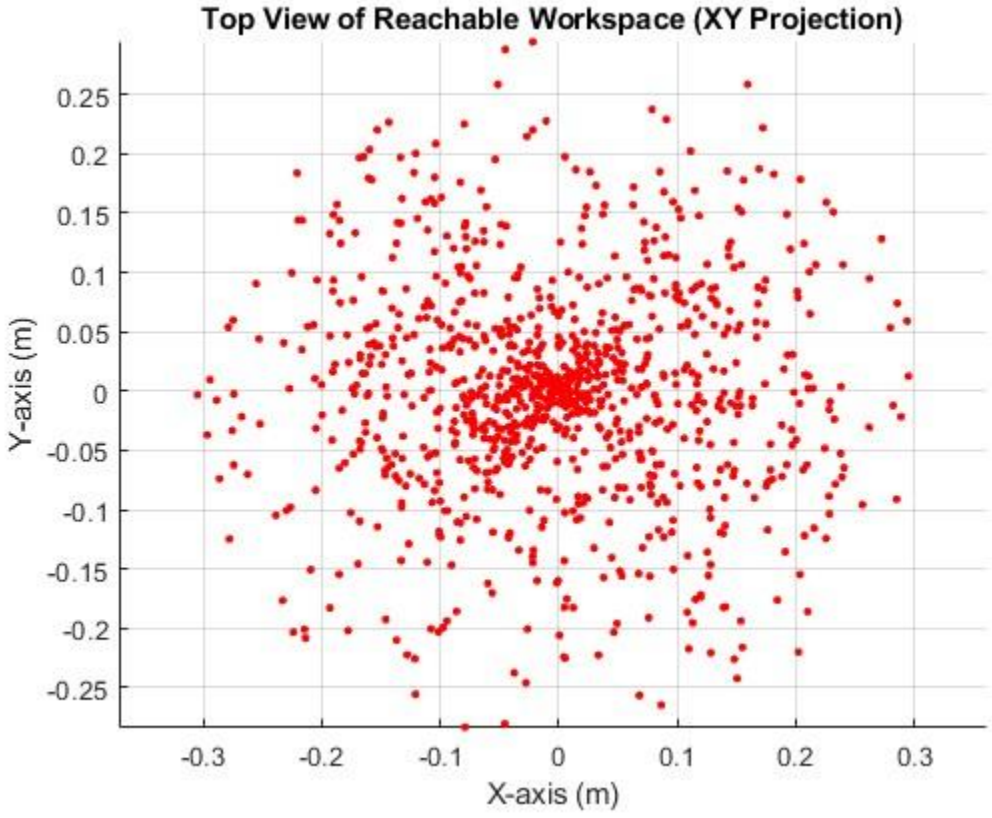
$$\theta_i = \theta_i^{\text{min}} + \left(\theta_i^{\text{max}} - \theta_i^{\text{min}}\right) \times \texttt{rand(N,1)}.$$

- The MATLAB functions `linspace`, `ndgrid`, and `scatter3` may be of help for this task.

```
% Number of random configurations to sample
numSamples = 1000;

% Joint limits (all -150° to 150°)
theta_min = deg2rad(-150);
theta_max = deg2rad(150);

% Generate random joint configurations within limits
theta_1 = theta_min + (theta_max - theta_min) * rand(numSamples, 1);
theta_2 = theta_min + (theta_max - theta_min) * rand(numSamples, 1);
theta_3 = theta_min + (theta_max - theta_min) * rand(numSamples, 1);
theta_4 = theta_min + (theta_max - theta_min) * rand(numSamples, 1);

% Initialize arrays for storing end-effector positions
X = zeros(numSamples, 1);
Y = zeros(numSamples, 1);
Z = zeros(numSamples, 1);

% Compute end-effector positions for each sampled configuration
for i = 1:numSamples
    [X(i), Y(i), Z(i), ~] = pincherFK(theta_1(i), theta_2(i), theta_3(i), theta_4(i));
end

% 3D Scatter Plot - Isometric View
figure;
scatter3(X, Y, Z, 10, 'b', 'filled');
xlabel('X-axis (m)');
ylabel('Y-axis (m)');
zlabel('Z-axis (m)');
title('Phantom X Pincher Reachable Workspace');
grid on;
axis equal;
view(3); % Isometric View

% 2D Projection - Top-down View (XY Plane)
figure;
scatter(X, Y, 10, 'r', 'filled');
xlabel('X-axis (m)');
ylabel('Y-axis (m)');
title('Top View of Reachable Workspace (XY Projection)');
grid on;
axis equal;
```

```
% Maximum Horizontal Reach (Farthest point from origin in XY plane)
max_horizontal_reach = max(sqrt(X.^2 + Y.^2));
disp(['Maximum Horizontal Reach: ', num2str(max_horizontal_reach), ' meters']);
```





Maximum Horizontal Reach: 0.305 meters

Task 4.10 Communicating with motors (7 points)

Provide a MATLAB `function [x,y,z,R] = findPincher()` or `function [x,y,z,R,theta,phi] = findPincher()` that queries the current servo angles from Phantom X Pincher motor encoders and returns the current end-effector position and orientation in the specified order. The function should be properly commented.

- Physically measure and note the end-effector position and orientation. How does it compare to the pose returned by your function?
- Do you think the pose returned by this process will ever be accurate? If not, what do you think are the sources of error?

```
function [x,y,z,R] = findPincher(arm)
    % Send arm as input
    angles = arm.getpos(); % Get angles of all joints
    [x,y,z,R] = pincherFK(-angles(1),-angles(2),-angles(3),-angles(4));
    % Call previously made pincherFK function to get the x, y, z
    % coordinates and the rotation matrix. The angles are input as negative
    % because of the way we assigned our screw axes in the forward mapping.
end
```

```
arm.setpos(1,0)
arm.setpos(2,0)
arm.setpos(3,pi/2)
arm.setpos(4,0)
[x,y,z, R] = findPincher(arm)
```

```
x = -0.0021
y = 0.2055
z = 0.2316
R = 3×3
    0.9999    0.0010   -0.0102
    0.0102   -0.1011    0.9948
         0   -0.9949   -0.1011
```

The physical measurements are with 1cm which means that our forward mapping has been very successful.

The pose returned by the forward kinematics function will never be perfectly accurate due to several sources of error. Sensor inaccuracies, including encoder errors and miscalibration may contribute to this error.