```
% Ryan Dewsnap
% 32000408
% CS403 Homework 5

clear all
close all
clc

addpath('C:/Matlab/matlab_utils')

fprintf('Ryan Dewsnap\n32000408\nCS403 HW#5\n\n')

clf;

drawCoordinate3DScale(eye(3), zeros(3,1), 0.1);          % draw global frame

q = deg2rad([0 0 0 0 0 0]);                              % convert degrees to radian for easy input

p = {};
p{1} = [0; 0; 0];
p{2} = [0; 0; 0.15];
p{3} = [0.30; 0; 0];
p{4} = [0.15; 0; 0];
p{5} = [0.10; 0; 0];
p{6} = [0.07; 0; 0];
p{7} = [0.05; 0; 0];

S = {};                                  % joint orientations
S{1} = zeros(3,1); S{1}(3) = 1;
S{2} = zeros(3,1); S{2}(2) = 1;
S{3} = zeros(3,1); S{3}(2) = 1;
S{4} = zeros(3,1); S{4}(1) = 1;
S{5} = zeros(3,1); S{5}(2) = 1;
S{6} = zeros(3,1); S{6}(1) = 1;

T01 = SE3(eul2matrix([q(1), 0, 0]), [0; 0; 0]);          % calculate SE3 matrices
T12 = SE3(eul2matrix([0, q(2), 0]), [0; 0; 0.15]);
T23 = SE3(eul2matrix([0, q(3), 0]), [0.30; 0; 0]);
T34 = SE3(eul2matrix([0, 0, q(4)]), [0.15; 0; 0]);
T45 = SE3(eul2matrix([0, q(5), 0]), [0.10; 0; 0]);
T56 = SE3(eul2matrix([0, 0, q(6)]), [0.07; 0; 0]);
Tee = SE3(eul2matrix([0, 0, 0]), [0.05; 0; 0]);

T02 = T01*T12;                                           % calculate SE3 matrices relative to global
T03 = T02*T23;
T04 = T03*T34;
T05 = T04*T45;
```

```
 48 T06 = T05*T56;
 49 TEE = T06*Tee;
 50
 51 drawLine3D(T01(1:3,4), T02(1:3,4));                    % draw lines, taking XYZ position from SE3
 52 drawLine3D(T02(1:3,4), T03(1:3,4));
 53 drawLine3D(T03(1:3,4), T04(1:3,4));
 54 drawLine3D(T04(1:3,4), T05(1:3,4));
 55 drawLine3D(T05(1:3,4), T06(1:3,4));
 56 drawLine3D(T06(1:3,4), TEE(1:3,4));
 57
 58 drawCoordinate3DScale(TEE(1:3,1:3), TEE(1:3,4), 0.05);   % draw end effector frame
 59
 60 grid on
 61 view(60, 30);
 62
 63 xlabel('x', 'fontsize',20);
 64 ylabel('y', 'fontsize',20);
 65 zlabel('z', 'fontsize',20);
 66
 67 test = Jacob(q,p,S)
 68
 69 % functions
 70
 71 function out = Jacob(q,p,S)
 72     T01 = SE3(eul2matrix([q(1), 0, 0]), p{1});
 73     T12 = SE3(eul2matrix([0, q(2), 0]), p{2});
 74     T23 = SE3(eul2matrix([0, q(3), 0]), p{3});
 75     T34 = SE3(eul2matrix([0, 0, q(4)]), p{4});
 76     T45 = SE3(eul2matrix([0, q(5), 0]), p{5});
 77     T56 = SE3(eul2matrix([0, 0, q(6)]), p{6});
 78     Tee = SE3(eul2matrix([0, 0, 0]), p{7});              % calculate SE3 matrices
 79
 80     T02 = T01*T12;
 81     T03 = T02*T23;
 82     T04 = T03*T34;
 83     T05 = T04*T45;
 84     T06 = T05*T56;
 85     TEE = T06*Tee;                                       % calculate SE3 matrices relative to global
 86
 87     out = zeros(6,6);
 88     v = [0; 0; 1];                                       % vector multiplied for 3rd column
 89
 90     out(1:3,1) = S{1};
 91     out(4:6,1) = cross(v, TEE(1:3,4));
 92
 93     out(1:3,2) = S{2};
```

```
 94      out(4:6,2) = cross(T01(1:3,3), TEE(1:3,4)-T01(1:3,4));
 95
 96      out(1:3,3) = S{3};
 97      out(4:6,3) = cross(T02(1:3,3), TEE(1:3,4)-T02(1:3,4));
 98
 99      out(1:3,4) = S{4};
100      out(4:6,4) = cross(T03(1:3,3), TEE(1:3,4)-T03(1:3,4));
101
102      out(1:3,5) = S{5};
103      out(4:6,5) = cross(T04(1:3,3), TEE(1:3,4)-T04(1:3,4));
104
105      out(1:3,6) = S{6};
106      out(4:6,6) = cross(T05(1:3,3), TEE(1:3,4)-T05(1:3,4));
107
108  end
109
110  function zero = check_rotm(R)    % checks integrity of rotational matrix against properties
111      zero = R(1:3, 1:3).' - inv(R(1:3, 1:3));        % inverse = transpose
112      zero = zero + det(R(1:3, 1:3)) - 1;             % determinant = 1
113      zero = zero + dot(R(1:3, 1), R(1:3, 2));        % dot product of 2 columns = 0
114      zero = zero + dot(R(1, 1:3), R(2, 1:3));        % dot product of 2 rows = 0
115  end
116
117  function x = SE3(R, t)
118      row3 = [0 0 0 1];
119      x = [R t; row3];
120  end
121
122  function matrix = eul2matrix(eul)
123      s = sin(eul);
124      c = cos(eul);
125
126      matrix = zeros(3,3);
127
128      matrix(1,1) = c(2)*c(1);     % build ZYX matrix from identities
129      matrix(1,2) = s(2)*s(3)*c(1) - s(1)*c(3);
130      matrix(1,3) = s(2)*c(3)*c(1) + s(1)*s(3);
131      matrix(2,1) = c(2)*s(1);
132      matrix(2,2) = s(1)*s(2)*s(3) + c(1)*c(3);
133      matrix(2,3) = s(1)*s(2)*c(3) - c(1)*s(3);
134      matrix(3,1) = -s(2);
135      matrix(3,2) = c(2)*s(3);
136      matrix(3,3) = c(2)*c(3);
137  end
138
139  function x = SE3Inv(T)
140      R = T(1:3, 1:3);
```

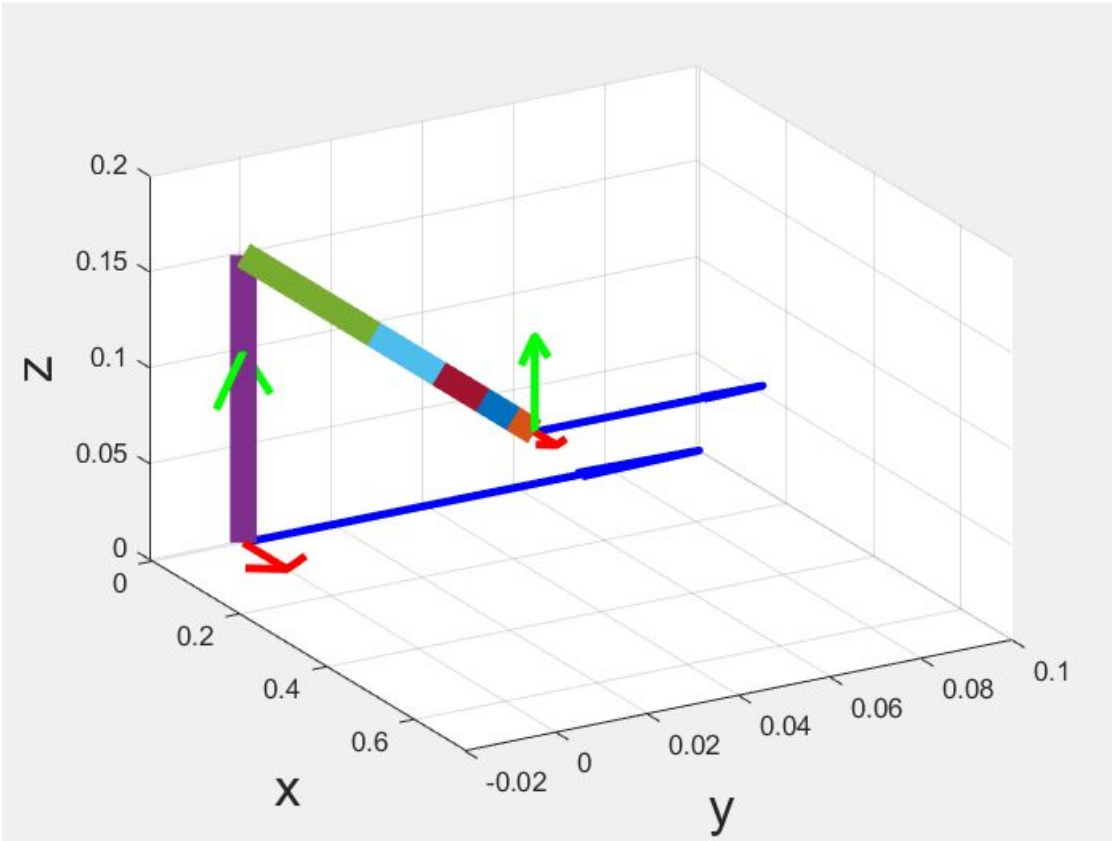
```
141     t = T(1:3, 4);
142
143     r3 = T(4, 1:4);
144     x = [R', -R'*t; r3];
145 end
146
147
148
```

```
Ryan Dewsnap
32000408
CS403 HW#5


jacobian =

         0         0         0    1.0000         0    1.0000
         0    1.0000    1.0000         0    1.0000         0
    1.0000         0         0         0         0         0
         0         0         0         0         0         0
    0.6700    0.6700    0.6700    0.3700    0.2200    0.1200
         0         0         0         0         0         0

>>
```


0.2
0.15
0.1
0.05
0
z
0
0.2
0.4
0.6
x
-0.02
0
0.02
0.04
0.06
0.08
0.1
y