Table of Contents

Homework 3

```
clc;clear;close all;
```

Generate the Transforms for COM

```
syms t1 t2 t3 t1_dot t2_dot t3_dot t1_dotdot t2_dotdot t3_dotdot real
syms l1 l2 l3 m1 m2 m3 lc1 lc2 lc3 real
% syms I1xx I1yy I1zz I2xx I2yy I2zz I3xx I3yy I3zz positive
% I1 = [I1xx 0 0 ; 0 I1yy 0; 0 0 I1zz];
% I3 = [I2xx 0 0 ; 0 I2yy 0; 0 0 I2zz];
% I2 = [I3xx 0 0 ; 0 I3yy 0; 0 0 I3zz];
I1 = sym('I1', [3 3]);
I2 = sym('I2', [3 3]);
I3 = sym('I3', [3 3]);
% dh table [theta d a alpha]
% generating transform to mass 1
dh_table_m1 = [t1 lc1 0 0];
T0_m1_total = get_fwdkin(dh_table_m1,true);
T0_m1 = T0_m1_total;
Jv1 = zeros(3,3);
Jw1 = [0 0 0; 0 0 0; 1 0 0];

% generating transform to mass 2
dh_table_m2 = [t1 l1 0 pi/2;
               t2 0 lc2 0];
T0_m2_total = get_fwdkin(dh_table_m2,true);
T0_m2 = T0_m2_total(:,:,2);
% jacobian stuff
T0_1_m2 = T0_m2_total(:,:,1);
pe_m2 = T0_m2(1:3,4);
Jv2 = simplify([cross([0;0;1],pe_m2)';
       cross(T0_1_m2(1:3,3),pe_m2-T0_1_m2(1:3,4))';
       0 0 0]','Steps',10);
Jw2 = [0 0 1;T0_1_m2(1:3,3)'; 0 0 0]';

% generating transform to mass 3
dh_table_m3 = [t1 l1 0 pi/2;
```

```
                t2 0 l2 0;
                t3 0 lc3 0];
T0_m3_total = get_fwdkin(dh_table_m3,true);
T0_m3 = T0_m3_total(:,:,3);
% jacobian stuff
T0_1_m3 = T0_m3_total(:,:,1);
T0_2_m3 = T0_m3_total(:,:,2);
pe_m3 = T0_m3(1:3,4);
Jv3 = simplify([cross([0;0;1],pe_m3)';
       cross(T0_1_m3(1:3,3),pe_m3-T0_1_m3(1:3,4))';
       cross(T0_2_m3(1:3,3),pe_m3-T0_2_m3(1:3,4))']','Steps',10);
Jw3 = [0 0 1;T0_1_m3(1:3,3)'; T0_2_m3(1:3,3)']';

% generating transform to tip
dh_table_tip = [t1 l1 0 pi/2;
                t2 0 l2 0;
                t3 0 l3 0];
T0_tip_total = get_fwdkin(dh_table_tip,true);
T0_tip = T0_tip_total(:,:,3);
% jacobian stuff
T0_1_tip = T0_tip_total(:,:,1);
T0_2_tip = T0_tip_total(:,:,2);
pe_tip = T0_tip(1:3,4);
Jvtip = simplify([cross([0;0;1],pe_tip)';
       cross(T0_1_tip(1:3,3),pe_tip-T0_1_tip(1:3,4))';
       cross(T0_2_tip(1:3,3),pe_tip-T0_2_tip(1:3,4))']','Steps',10);
Jwtip = [0 0 1;T0_1_tip(1:3,3)'; T0_2_tip(1:3,3)']';
```

Deriving the D matrix

```
m = [m1 m2 m3];

Jv = sym('Jv',[3 3 3]);
Jv(:,:,1) = Jv1; Jv(:,:,2) = Jv2; Jv(:,:,3) = Jv3;

Jw = sym('Jw',[3 3 3]);
Jw(:,:,1) = Jw1; Jw(:,:,2) = Jw2; Jw(:,:,3) = Jw3;

I = sym('I',[3 3 3]);
I(:,:,1) = I1; I(:,:,2) = I2; I(:,:,3) = I3;

R = sym('R',[3 3 3]);
R(:,:,1) = T0_m1(1:3,1:3); R(:,:,2) = T0_m2(1:3,1:3); R(:,:,3) =
 T0_m3(1:3,1:3);

D = zeros(3,3);
for i = 1:3
    lin = simplify(m(i)*Jv(:,:,i)'*Jv(:,:,i),'Steps',30);
    ang =
 simplify(Jw(:,:,i)'*R(:,:,i)*I(:,:,i)*R(:,:,i)'*Jw(:,:,i),'Steps',30);
    D = D + lin + ang;
end
fprintf("Inertia Term: D")
```

```
pretty(D)
```

```
Inertia Term: D[[I13_3 + cos(t2 + t3) (I32_2 cos(t2 + t3) + I31_2
 sin(t2 + t3))

   + sin(t2 + t3) (I32_1 cos(t2 + t3) + I31_1 sin(t2 + t3))

                                             2
   + m3 (lc3 cos(t2 + t3) + l2 cos(t2))  + cos(t2) (I22_2 cos(t2) +
 I21_2

                                                               2
  2
   sin(t2)) + sin(t2) (I22_1 cos(t2) + I21_1 sin(t2)) + lc2  m2
 cos(t2) ,

  #5 + #3 + I22_3 cos(t2) + I21_3 sin(t2), #5 + #3],

  [#4 + #2 + I23_2 cos(t2) + I23_1 sin(t2), I23_3 + I33_3

        2             2                         2
   + lc2  m2 + m3 (l2  + cos(t3) l2 lc3 2 + lc3 ), #1],

                       2
  [#4 + #2, #1, m3 lc3  + I33_3]]

where

   #1 == I33_3 + lc3 m3 (lc3 + l2 cos(t3))

   #2 == I33_1 sin(t2 + t3)

   #3 == I31_3 sin(t2 + t3)

   #4 == I33_2 cos(t2 + t3)

   #5 == I32_3 cos(t2 + t3)
```

Deriving the C matrix with christofel symbols

```
C = sym('C',[3 3]);
q = [t1;t2;t3];
q_dot = [t1_dot;t2_dot;t3_dot];
for k = 1:3
    for j = 1:3
        C(k,j) = sym(0);
        for i = 1:3
            C(k,j) = C(k,j) + simplify(1/2*(diff(D(k,j),q(i)) +...
                diff(D(k,i),q(j)) -...
                diff(D(i,j),q(k)))*q_dot(i),'Steps',50);
        end
```

```
    end
end
fprintf("Centripetal Term: C\n");
pretty(C)
```

```
Centripetal Term: C
[[t2_dot #5 - t3_dot #6, t1_dot #5 + t2_dot (#15 - #14

   + I21_3 cos(t2) - I22_3 sin(t2)) + #4 - #3,

  I31_3 t2_dot cos(t2 + t3) - t1_dot #6 + #4 - I32_3 t2_dot

   sin(t2 + t3) - #3],

  [-t1_dot #5, - (t1_dot (#15 - #17 - #14 + #16

   + I21_3 cos(t2) - I23_1 cos(t2) - I22_3 sin(t2) + I23_2 sin(t2)))/2

   - #1, - #7 - #2 - #1],

  [t1_dot #6, #2 - #7, -#7]]

where

   #1 == l2 lc3 m3 t3_dot sin(t3)

   #2 == l2 lc3 m3 t2_dot sin(t3)

   #3 == I32_3 t3_dot sin(t2 + t3)

   #4 == I31_3 t3_dot cos(t2 + t3)

                                   I21_2 cos(2 t2)   I22_1 cos(2 t2)
   #5 == #13 + #12 + #11 - #10 + --------------- + ---------------
                                          2                 2

                                                     2
      I21_1 sin(2 t2)   I22_2 sin(2 t2)         l2  m3 sin(2 t2)
    + --------------- - --------------- - #8 - ----------------
             2                 2                        2

         2
      lc2  m2 sin(2 t2)
    - ----------------- - #9
              2

                                        l2 lc3 m3 sin(t3)   #9
   #6 == #10 - #12 - #11 - #13 + #8 + ----------------- + --
                                                2            2

         t1_dot (#15 - #17 - #14 + #16)
   #7 == ------------------------------
                       2
```

```
          2
       lc3  m3 sin(#18)
#8 == ----------------
             2

#9 == l2 lc3 m3 sin(2 t2 + t3)

        I32_2 sin(#18)
#10 == --------------
             2

        I31_1 sin(#18)
#11 == --------------
             2

        I32_1 cos(#18)
#12 == --------------
             2

        I31_2 cos(#18)
#13 == --------------
             2

#14 == I32_3 sin(t2 + t3)

#15 == I31_3 cos(t2 + t3)

#16 == I33_2 sin(t2 + t3)

#17 == I33_1 cos(t2 + t3)

#18 == 2 t2 + 2 t3
```

Deriving Gravity Term

```
syms g real
g_vec = [0 0 g]';
P = 0;
pos = sym('pos',[3 3]);
pos(:,1) = T0_m1(1:3,4); pos(:,2) = T0_m2(1:3,4); pos(:,3) =
 T0_m3(1:3,4);
for i = 1:3
    P = P + m(i)*g_vec'*pos(:,i);
end
G = simplify([diff(P,t1); diff(P,t2); diff(P,t3)],'Steps',20);
fprintf("Gravity Term: G\n");
pretty(G)
```

```
Gravity Term: G
/                            0                            \
|                                                         |
```

```
| g m3 (lc3 cos(t2 + t3) + l2 cos(t2)) + g lc2 m2 cos(t2) |
|                                                          |
\                  g lc3 m3 cos(t2 + t3)                   /
```

Fully Dynamical Equation

```
q_dotdot = [t1_dotdot; t2_dotdot; t3_dotdot];
tau = D*q_dotdot + C*q_dot + G;
fprintf("Dynamical Model: tau\n");
pretty(tau)
```

```
Dynamical Model: tau
--
|  [t2_dot (t1_dot #11 + t2_dot (#21 - #20 + I21_3 cos(t2) - I22_3
 sin(t2))
--

   + #6 - #5) - t3_dot (t1_dot #12 - I31_3 t2_dot cos(t2 + t3) - #6

   + I32_3 t2_dot sin(t2 + t3) + #5) + t1_dotdot (I13_3

   + cos(t2 + t3) (I32_2 cos(t2 + t3) + I31_2 sin(t2 + t3))

   + sin(t2 + t3) (I32_1 cos(t2 + t3) + I31_1 sin(t2 + t3))

          2
   + m3 #1  + cos(t2) (I22_2 cos(t2) + I21_2 sin(t2))

                                                    2           2
   + sin(t2) (I22_1 cos(t2) + I21_1 sin(t2)) + lc2  m2 cos(t2) )

   + t3_dotdot (#10 + #8) + t1_dot (t2_dot #11 - t3_dot #12)

   + t2_dotdot (#10 + #8 + I22_3 cos(t2) + I21_3 sin(t2))],

                          2
  [t3_dotdot #2 - t1_dot  #11 + t2_dotdot (I23_3 + I33_3

        2            2                          2
   + lc2  m2 + m3 (l2  + cos(t3) l2 lc3 2 + lc3 )) - t3_dot (#13 + #4
 + #3)

   - t2_dot ((t1_dot (#21 - #23 - #20 + #22 + I21_3 cos(t2) - I23_1

   cos(t2) - I22_3 sin(t2) + I23_2 sin(t2)))/2 + #3)

   + t1_dotdot (#9 + #7 + I23_2 cos(t2) + I23_1 sin(t2))

   + g m3 #1 + g lc2 m2 cos(t2)],

  --                2                                 2
  |  t3_dotdot (m3 lc3  + I33_3) + t2_dotdot #2 + t1_dot  #12
```

```
  --

   - t2_dot (#13 - #4) + t1_dotdot (#9 + #7)

     t1_dot t3_dot (#21 - #23 - #20 + #22)                          --
 --
   - ------------------------------------ + g lc3 m3 cos(t2 + t3)  |
  |
                     2                                              --
 --

where

   #1 == lc3 cos(t2 + t3) + l2 cos(t2)

   #2 == I33_3 + lc3 m3 (lc3 + l2 cos(t3))

   #3 == l2 lc3 m3 t3_dot sin(t3)

   #4 == l2 lc3 m3 t2_dot sin(t3)

   #5 == I32_3 t3_dot sin(t2 + t3)

   #6 == I31_3 t3_dot cos(t2 + t3)

   #7 == I33_1 sin(t2 + t3)

   #8 == I31_3 sin(t2 + t3)

   #9 == I33_2 cos(t2 + t3)

   #10 == I32_3 cos(t2 + t3)

                                  I21_2 cos(2 t2)   I22_1 cos(2 t2)
   #11 == #19 + #18 + #17 - #16 + --------------- + ---------------
                                         2                 2

                                                        2
        I21_1 sin(2 t2)   I22_2 sin(2 t2)         l2  m3 sin(2 t2)
      + --------------- - --------------- - #14 - ----------------
               2                  2                      2

           2
        lc2  m2 sin(2 t2)
      - ----------------- - #15
                2

                                        l2 lc3 m3 sin(t3)   #15
   #12 == #16 - #18 - #17 - #19 + #14 + ----------------- + ---
                                                2            2

          t1_dot (#21 - #23 - #20 + #22)
   #13 == ------------------------------
                        2
```

```
             2
          lc3  m3 sin(#24)
#14 == ----------------
                 2

#15 == l2 lc3 m3 sin(2 t2 + t3)

       I32_2 sin(#24)
#16 == --------------
             2

       I31_1 sin(#24)
#17 == --------------
             2

       I32_1 cos(#24)
#18 == --------------
             2

       I31_2 cos(#24)
#19 == --------------
             2

#20 == I32_3 sin(t2 + t3)

#21 == I31_3 cos(t2 + t3)

#22 == I33_2 sin(t2 + t3)

#23 == I33_1 cos(t2 + t3)

#24 == 2 t2 + 2 t3
```

Change dynamical model

```
tau2 = subs(tau,[lc1,lc2,lc3],[l1,l2,l3]);
tau2 = subs(tau2, [I1 I2 I3],zeros(3,9));
fprintf("Second Dynamical Model: tau\n");
pretty(tau2)
```

```
Second Dynamical Model: tau
                  2     2           2
[[t1_dotdot (m3 #1  + l2  m2 cos(t2) ) - t1_dot (t3_dot #4 + t2_dot
 #3)

   - t1_dot t3_dot #4 - t1_dot t2_dot #3],

            2
  [#3 t1_dot  - t3_dot (l2 l3 m3 t2_dot sin(t3) + l2 l3 m3 t3_dot
 sin(t3))
```

```
                 2             2                        2
  + t2_dotdot (l2  m2 + m3 (l2  + 2 cos(t3) l2 l3 + l3 ))

  + g m3 #1 + l3 m3 t3_dotdot #2 + g l2 m2 cos(t2) - l2 l3 m3 t2_dot
t3_dot

  sin(t3)],

        2        2
 [t1_dot  #4 + l3  m3 t3_dotdot + l3 m3 t2_dotdot #2 + g l3 m3

                              2
  cos(t2 + t3) + l2 l3 m3 t2_dot  sin(t3)]]

where

  #1 == l3 cos(t2 + t3) + l2 cos(t2)

  #2 == l3 + l2 cos(t3)

                 2                  2
               l2  m2 sin(2 t2)   l2  m3 sin(2 t2)
  #3 == #5 + ---------------- + ---------------- + #6
                      2                  2

               l2 l3 m3 sin(t3)   #6
  #4 == #5 + ---------------- + --
                      2           2

            2
          l3  m3 sin(2 t2 + 2 t3)
  #5 == -----------------------
                    2

  #6 == l2 l3 m3 sin(2 t2 + t3)
```

Use configuration

```
variables = [l1 l2 l3 m1 m2 m3 g];
knowns = [0.3 0.3 0.3 0.5 0.5 0.5 9.8];
joints = [0,0,0];
tau2_val = subs(tau2,[t1 t2 t3], joints);
tau2_val = subs(tau2_val,variables,knowns);
fprintf("Output at given configuration\n");
pretty(tau2_val)
```

```
Output at given configuration
/           9 t1_dotdot          \
|           -----------          |
|                40              |
```

```
|                                       |
| 9 t2_dotdot   9 t3_dotdot   441 |
| ----------- + ----------- + --- |
|      40           100         100 |
|                                       |
| 9 t2_dotdot   9 t3_dotdot   147 |
| ----------- + ----------- + --- |
\     100           200         100 /
```

Appendix

```
function T = get_fwdkin(dh_table,is_sym)
    rows = size(dh_table,1);
    if is_sym
        T = sym('T',[4,4,rows]);
    else
        T = zeros(4,4,rows);
    end
    for i = 1:rows
        if i == 1
            T(:,:,i) = tdh(dh_table(i,:));
        else
            T(:,:,i) = T(:,:,i-1)*tdh(dh_table(i,:));
            if is_sym
                T(:,:,i) = simplify(T(:,:,i),'Steps',20);
            end
        end
    end
end

function T = tdh(dh)
theta = dh(1);
d = dh(2);
a = dh(3);
alpha = dh(4);
T = [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
```

Published with MATLAB® R2019b