Table of Contents


```
% Lagrangian dynamics of three link robot (HW01)

clc
close all
clear all
```

System Constants

```
syms g L_torso L_leg1 L_leg2 m_torso m_leg1 m_leg2...
    I_torso I_leg1 I_leg2 real
```

Configuration Variables (Figure 1a)

```
syms x y q1 q2 q3 dx dy dq1 dq2 dq3 d2x d2y d2q1 d2q2 d2q3 real
% Generalized Coordinates
q = [x; y; q1; q2; q3] ;
% Generalized Velocities
dq = [dx; dy; dq1; dq2; dq3] ;
% Generalized Acceleration
d2q = [d2x; d2y; d2q1; d2q2; d2q3] ;
```

Define numerical values / vectors (cases (i) and (ii))

```
qi = [0.5, 0.5*sqrt(3), 150*pi/180, 120*pi/180, 30*pi/180]';
dqi = [-0.8049, -0.4430, 0.0938, 0.9150, 0.9298]';
qii = [0.3420, 0.9397, 170*pi/180, 20*pi/180, 30*pi/180]';
dqii = [-0.1225, -0.2369, 0.5310, 0.5904, 0.6263]';

L_torso_num = 1/2; L_leg1_num = 1; L_leg2_num = 1;
m_torso_num = 10; m_leg1_num = 5; m_leg2_num = 5;
I_torso_num = 1; I_leg1_num = 1/2; I_leg2_num = 1/2;
g_num = 9.81;
```

1(a)

Positions

```
p_torso = [L_torso.*sin(q3)/2 + x; L_torso.*cos(q3)/2 + y] ;
p_leg1 = [L_leg1.*sin(q1 + q3)/2 + x; L_leg1.*cos(q1 + q3)/2 + y] ;
p_leg2 = [L_leg2.*sin(q2 + q3)/2 + x; L_leg2.*cos(q2 + q3)/2 + y] ;

P_sym = [p_torso, p_leg1, p_leg2] ;

Pi = subs(P_sym, q, qi);
Pi = subs(Pi, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Pi = vpa(Pi,4) % m

Pii = subs(P_sym, q, qii);
Pii = subs(Pii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Pii = vpa(Pii,4) % m
```

```
Pi =

[ 0.625,   0.5,  0.75]
[ 1.083, 0.366, 0.433]
```

```
Pii =

[ 0.467,  0.171, 0.725]
[ 1.156, 0.4699, 1.261]
```

1(b)

Velocities

```
dp_torso = jacobian(p_torso, q) * dq ;
dp_leg1 = jacobian(p_leg1, q) * dq ;
dp_leg2 = jacobian(p_leg2, q) * dq ;

V_sym = [dp_torso, dp_leg1, dp_leg2] ;

Vi = subs(V_sym, q, qi);
Vi = subs(Vi, dq, dqi);
Vi = subs(Vi, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Vi = vpa(Vi,4) % m/s

Vii = subs(V_sym, q, qii);
Vii = subs(Vii, dq, dqii);
Vii = subs(Vii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Vii = vpa(Vii,4) % m/s
```

```
Vi =

[ -0.6036, -1.317,  -1.604]
[ -0.5592, -0.443, -0.9042]


Vii =

[  0.0131,  -0.6663,  0.2685]
[ -0.3152, -0.03899, -0.7029]
```

1(c)

Kinetic Energy

```
KE_torso = 1/2 * dp_torso' * m_torso * dp_torso + 1/2 * I_torso *...
    ( jacobian(q3, q) * dq )^2 ;
KE_leg1 = 1/2 * dp_leg1' * m_leg1 * dp_leg1 + 1/2 * I_leg1 *...
    ( jacobian(q3 + q1, q) * dq )^2 ;
```

```
KE_leg2 = 1/2 * dp_leg2' * m_leg2 * dp_leg2 + 1/2 * I_leg2 *...
    ( jacobian(q3 + q2, q) * dq )^2 ;

KE_sym = KE_torso + KE_leg1 + KE_leg2 ;

KE_i = subs(KE_sym, q, qi);
KE_i = subs(KE_i, dq, dqi);
KE_i = subs(KE_i, [L_torso, L_leg1, L_leg2, m_torso, m_leg1,
 m_leg2,...
    I_torso, I_leg1, I_leg2], [L_torso_num, L_leg1_num, L_leg2_num,...
    m_torso_num, m_leg1_num, m_leg2_num, I_torso_num, I_leg1_num,...
    I_leg2_num]);
KE_i = vpa(KE_i,4) % Joules

KE_ii = subs(KE_sym, q, qii);
KE_ii = subs(KE_ii, dq, dqii);
KE_ii = subs(KE_ii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1,
 m_leg2,...
    I_torso, I_leg1, I_leg2], [L_torso_num, L_leg1_num, L_leg2_num,...
    m_torso_num, m_leg1_num, m_leg2_num, I_torso_num, I_leg1_num,...
    I_leg2_num]);
KE_ii = vpa(KE_ii,4) % Joules
```

KE_i =

18.23

KE_ii =

3.928

1(d)

Potential Energy

```
PE_torso = m_torso * g * [0 1] * p_torso ;
PE_leg1 = m_leg1 * g * [0 1] * p_leg1 ;
PE_leg2 = m_leg2 * g * [0 1] * p_leg2 ;

PE_sym = PE_torso + PE_leg1 + PE_leg2;

PE_i = subs(PE_sym, q, qi);
PE_i = subs(PE_i, [m_torso, m_leg1, m_leg2, L_torso, L_leg1, L_leg2,
 g],[m_torso_num,...
     m_leg1_num, m_leg2_num, L_torso_num, L_leg1_num, L_leg2_num,
 g_num]);
PE_i = vpa(PE_i,4) % Joules

PE_ii = subs(PE_sym, q, qii);
PE_ii = subs(PE_ii, [m_torso, m_leg1, m_leg2, L_torso, L_leg1, L_leg2,
 g],[m_torso_num,...
```

```
    m_leg1_num, m_leg2_num, L_torso_num, L_leg1_num, L_leg2_num,
 g_num]);
PE_ii = vpa(PE_ii,4) % Joules
```

```
PE_i =

145.4


PE_ii =

198.3
```

1(e)

Lagrangian

```
L = KE_sym - PE_sym ;
```

Equations of Motion (LHS)

```
% State vector
x = [q;
     dq] ;
% Time-derivative of State
dx = [dq ;
      d2q] ;

EOM = jacobian(jacobian(L, dq), x) * dx - jacobian(L, q)' ;
EOM = simplify(EOM);

% Derive the dynamics in terms of the Robot Manipulator Dynamics
% D(q) d2q + C(q, dq) dq + G(q) = B(q) u

% set actuated coordinates
q_act = [q1; q2] ;

[D, C, G, B] = LagrangianDynamics(KE_sym, PE_sym, q, dq, q_act);

% Check if both sets of equations give the same results:
% The following expression below should give you zero.
simplify(D*d2q + C*dq + G  -  EOM)
```

```
ans =

 0
 0
 0
 0
```

```
    0
```

Numerical Matricies

D matrix

case (i)

```
Di = subs(D, q, qi);
Di = subs(Di, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2,...
    I_torso, I_leg1, I_leg2], [L_torso_num, L_leg1_num, L_leg2_num,...
    m_torso_num, m_leg1_num, m_leg2_num, I_torso_num, I_leg1_num,...
    I_leg2_num]);
Di = vpa(Di,4)

% case (ii)
Dii = subs(D, q, qii);
Dii = subs(Dii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2,...
    I_torso, I_leg1, I_leg2], [L_torso_num, L_leg1_num, L_leg2_num,...
    m_torso_num, m_leg1_num, m_leg2_num, I_torso_num, I_leg1_num,...
    I_leg2_num]);
Dii = vpa(Dii,4)
```

```
Di =

[   20.0,      0, -2.5, -2.165,  -2.5]
[      0,   20.0,    0,  -1.25,  -2.5]
[   -2.5,      0, 1.75,      0,  1.75]
[ -2.165,  -1.25,    0,   1.75,  1.75]
[   -2.5,   -2.5, 1.75,   1.75, 5.125]


Dii =

[   20.0,      0, -2.349,  1.607, 1.423]
[      0,   20.0, 0.8551, -1.915, -2.31]
[ -2.349, 0.8551,   1.75,      0,  1.75]
[  1.607, -1.915,      0,   1.75,  1.75]
[  1.423,  -2.31,   1.75,   1.75, 5.125]
```

C matrix

case (i)

```
Ci = subs(C, [q, dq], [qi dqi]);
Ci = subs(Ci, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Ci = vpa(Ci,4);
display(Ci)
```

```
% case (ii)
Cii = subs(C, [q, dq], [qii dqii]);
Cii = subs(Cii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num]);
Cii = vpa(Cii,4);
display(Cii)
```

```
Ci =

[ 0, 0,     0, -2.306, -3.468]
[ 0, 0, 2.559,  3.994,   4.54]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]


Cii =

[ 0, 0, 0.9895,  -2.33,  -2.123]
[ 0, 0,  2.719, -1.955, -0.5924]
[ 0, 0,      0,      0,       0]
[ 0, 0,      0,      0,       0]
[ 0, 0,      0,      0,       0]
```

G matrix

case (i)

```
Gi = subs(G, [q, dq], [qi dqi]);
Gi = subs(Gi, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2,
 g],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num, g_num]);
Gi = vpa(Gi,4);
display(Gi)

% case (ii)
Gii = subs(G, [q, dq], [qii dqii]);
Gii = subs(Gii, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, m_leg2,
 g],...
     [L_torso_num, L_leg1_num, L_leg2_num, m_torso_num,...
     m_leg1_num, m_leg2_num, g_num]);
Gii = vpa(Gii,4);
display(Gii)
```

```
Gi =

      0
  196.2
```

```
     0
-12.26
-24.53


Gii =

     0
 196.2
 8.388
-18.79
-22.66
```

B matrix

case (i)

```
Bi = B;
display(Bi)

% case (ii)
Bii = B;
display(Bii)


Bi =

[ 0, 0]
[ 0, 0]
[ 1, 0]
[ 0, 1]
[ 0, 0]


Bii =

[ 0, 0]
[ 0, 0]
[ 1, 0]
[ 0, 1]
[ 0, 0]
```

Problem 2 - Change of Coordinates

2(a) (Handwritten portion included at end of document)

Work shown on paper

```
T = [1 0 0 0 0;0 1 0 0 0; 0 0 1 0 1; 0 0 0 1 1 ; 0 0 0 0 1];
```

```
d = [0;0;-pi;-pi;0];
```

2(b) (Handwritten portion included at end of document)

Work shown on paper

```
dT = T;
dd = zeros(5,1);

ddT = T;
ddd = dd;
```

2(c)

```
qi_tild = T*qi + d
dqi_tild = dT*dqi + dd

qii_tild = T*qii + d
dqii_tild = dT*dqii + dd
```

```
qi_tild =

    0.5000
    0.8660
         0
   -0.5236
    0.5236


dqi_tild =

   -0.8049
   -0.4430
    1.0236
    1.8448
    0.9298


qii_tild =

    0.3420
    0.9397
    0.3491
   -2.2689
    0.5236


dqii_tild =

   -0.1225
```

```
-0.2369
 1.1573
 1.2167
 0.6263
```

2(d)

Configuration Variables

```
syms x y th1 th2 th3 dx dy dth1 dth2 dth3 d2x d2y d2th d2th d2th real
% Generalized Coordinates
q_tild = [x; y; th1; th2; th3] ;
% Generalized Velocities
dq_tild = [dx; dy; dth1; dth2; dth3] ;
% Generalized Acceleration
d2q_tild = [d2x; d2y; d2th; d2th; d2th] ;
```

Kinematics

Positions

```
p_torso_tild = [ L_torso.*sin(th3)/2 + x; L_torso.*cos(th3)/2 + y] ;
p_leg1_tild = [ - L_leg1.*sin(th1)/2 + x; - L_leg1.*cos(th1)/2 + y] ;
p_leg2_tild = [ - L_leg2.*sin(th2)/2 + x; - L_leg2.*cos(th2)/2 + y] ;

% Velocities
dp_torso_tild = jacobian(p_torso_tild, q_tild) * dq_tild  ;
dp_leg1_tild = jacobian(p_leg1_tild, q_tild) * dq_tild  ;
dp_leg2_tild = jacobian(p_leg2_tild, q_tild) * dq_tild ;
```

Kinetic Energy

```
KE_torso_tild = 1/2 * dp_torso_tild' * m_torso * dp_torso_tild +
 1/2*...
    I_torso * ( jacobian(th3, q_tild) * dq_tild )^2 ;
KE_leg1_tild = 1/2 * dp_leg1_tild' * m_leg1 * dp_leg1_tild + 1/2 *
 I_leg1 *...
    ( jacobian(th1, q_tild) * dq_tild )^2 ;
KE_leg2_tild = 1/2 * dp_leg2_tild' * m_leg2 * dp_leg2_tild + 1/2 *
 I_leg2 *...
    ( jacobian(th2, q_tild) * dq_tild )^2 ;

KE_sym_tild = KE_torso_tild + KE_leg1_tild + KE_leg2_tild ;
```

Potential Energy

```
PE_torso_tild  = m_torso * g * [0 1] * p_torso_tild  ;
PE_leg1_tild  = m_leg1 * g * [0 1] * p_leg1_tild  ;
PE_leg2_tild  = m_leg2 * g * [0 1] * p_leg2_tild  ;
PE_sym_tild = PE_torso_tild + PE_leg1_tild + PE_leg2_tild ;
```

Lagrangian

```
L_tild = KE_sym_tild - PE_sym_tild  ;
```

Equations of Motion (LHS)

Variables to find d/dt (partial L / partial dq) State vector

```
x_tild = [q_tild;
     dq_tild] ;
% Time-derivative of State
dx_tild = [dq_tild ;
      d2q_tild] ;

EOM_tild = jacobian(jacobian(L_tild, dq_tild), x_tild) * dx_tild -
 jacobian(L_tild, q_tild)' ;
EOM_tild = simplify(EOM_tild) ;
```

Derive the dynamics in terms of the Robot Manipulator Dynamics

D(q) d2q + C(q, dq) dq + G(q) = B(q) u

```
% set actuated coordinates
q_act_tild = [pi + th1 - th3; pi + th2 - th3] ;

[D_tild, C_tild, G_tild, B_tild] = LagrangianDynamics(KE_sym_tild,
 PE_sym_tild, q_tild,...
    dq_tild, q_act_tild) ;

% Check if both sets of equations give the same results:
% Check LHS of both sets to ensure they are the same
% The following expression below should give you zero.
simplify(D_tild*d2q_tild + C_tild*dq_tild + G_tild  -  EOM_tild)
```

```
ans =

 0
 0
 0
 0
 0
```

Numerical Matricies

D_tild matrix

case (i)

```
Di_tild = subs(D_tild, q_tild, qi_tild);
Di_tild = subs(Di_tild, dq_tild, dqi_tild);
Di_tild = subs(Di_tild, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Di_tild = vpa(Di_tild,4)

% case (ii)
Dii_tild = subs(D_tild, q_tild, qii_tild);
Dii_tild = subs(Dii_tild, dq_tild, dqii_tild);
Dii_tild = subs(Dii_tild, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Dii_tild = vpa(Dii_tild,4)
```

```
Di_tild =

[   20.0,      0, -2.5, -2.165, 2.165]
[      0,   20.0,    0,  -1.25, -1.25]
[   -2.5,      0, 1.75,      0,     0]
[ -2.165,  -1.25,    0,   1.75,     0]
[  2.165,  -1.25,    0,      0, 1.625]


Dii_tild =

[   20.0,      0, -2.349,  1.607, 2.165]
[      0,   20.0, 0.8551, -1.915, -1.25]
[ -2.349, 0.8551,   1.75,      0,     0]
[  1.607, -1.915,      0,   1.75,     0]
[  2.165,  -1.25,      0,      0, 1.625]
```

C_tild matrix

case (i)

```
Ci_tild = subs(C_tild, q_tild, qi_tild);
Ci_tild = subs(Ci_tild, dq_tild, dqi_tild);
Ci_tild = subs(Ci_tild, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Ci_tild = vpa(Ci_tild,4)

% case (ii)
Cii_tild = subs(C_tild, q_tild, qii_tild);
Cii_tild = subs(Cii_tild, dq_tild, dqii_tild);
Cii_tild = subs(Cii_tild, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
```

```
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Cii_tild = vpa(Cii_tild,4)
```

```
Ci_tild =

[ 0, 0,     0, -2.306, -1.162]
[ 0, 0, 2.559,  3.994, -2.013]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]


Cii_tild =

[ 0, 0, 0.9895,  -2.33, -0.7829]
[ 0, 0,  2.719, -1.955,  -1.356]
[ 0, 0,      0,      0,       0]
[ 0, 0,      0,      0,       0]
[ 0, 0,      0,      0,       0]
```

G_tild matrix

case (i)

```
Gi_tild = subs(G_tild, q_tild, qi_tild);
Gi_tild = subs(Gi_tild, dq_tild, dqi_tild);
Gi_tild = subs(Gi_tild, [L_torso, L_leg1, L_leg2, m_torso, m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2, g], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num, g_num]);
Gi_tild = vpa(Gi_tild,4)

% case (ii)
Gii_tild = subs(G_tild, q_tild, qii_tild);
Gii_tild = subs(Gii_tild, dq_tild, dqii_tild);
Gii_tild = subs(Gii_tild, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2, g], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num, g_num]);
Gii_tild = vpa(Gii_tild,4)
```

```
Gi_tild =

      0
  196.2
      0
 -12.26
 -12.26
```

```
Gii_tild =

      0
  196.2
  8.388
 -18.79
 -12.26
```

B_tild matrix

case (i)

```
Bi_tild = B_tild

% case (ii)
Bii_tild = B_tild
```

```
Bi_tild =

[  0,  0]
[  0,  0]
[  1,  0]
[  0,  1]
[ -1, -1]
```

```
Bii_tild =

[  0,  0]
[  0,  0]
[  1,  0]
[  0,  1]
[ -1, -1]
```

2(e) (Handwritten portion included at end of document)

```
% Linear algebra is on paper, final expression for D_tild_alt is
% implimented below
D_tild_alt = inv(T')*D*inv(T);

% Double check that the two versions of D_tild are equivalent by
 inputting
% numeric values into D_tild_alt for case (i). Then compare to the
% vector obtained from computing Lagrangian dynamics.

Di_tild_alt = subs(D_tild_alt, q, qi);
```

```
Di_tild_alt = subs(Di_tild_alt, dq, dqi);
Di_tild_alt = subs(Di_tild_alt, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Di_tild_alt = vpa(Di_tild_alt,4)

% This vector should output zeros
disp(Di_tild_alt - Di_tild)


Di_tild_alt =

[   20.0,      0, -2.5, -2.165, 2.165]
[      0,   20.0,    0,  -1.25, -1.25]
[   -2.5,      0, 1.75,      0,     0]
[ -2.165, -1.25,    0,   1.75,      0]
[  2.165, -1.25,    0,      0, 1.625]

[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
```

2(f) (Handwritten portion included at end of document)

```
% Linear algebra is on paper, final expressions for C_tild_alt,
 G_tild_alt,
% and B_tild_alt are implemented below

C_tild_alt = inv(T')*C*inv(T) ;
G_tild_alt = inv(T')*G ;
B_tild_alt = inv(T')*B ;

% Double check that the two versions of B_tild_alt, G_tild_alt,
% and C_tild_alt are equivalent by inputting
% numeric values into each matrix for case (i). Then compare to the
% vectors obtained from computing Lagrangian dynamics.

Ci_tild_alt = subs(C_tild_alt, q, qi);
Ci_tild_alt = subs(Ci_tild_alt, dq, dqi);
Ci_tild_alt = subs(Ci_tild_alt, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2 ], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num]);
Ci_tild_alt = vpa(Ci_tild_alt,4)
```

```
Gi_tild_alt = subs(G_tild_alt, q, qi);
Gi_tild_alt = subs(Gi_tild_alt, dq, dqi);
Gi_tild_alt = subs(Gi_tild_alt, [L_torso, L_leg1, L_leg2, m_torso,
 m_leg1, ...
    m_leg2, I_torso, I_leg1, I_leg2, g], [L_torso_num, L_leg1_num,...
    L_leg2_num, m_torso_num, m_leg1_num, m_leg2_num, I_torso_num,...
    I_leg1_num, I_leg2_num, g_num]);
Gi_tild_alt = vpa(Gi_tild_alt,4)

Bi_tild_alt = B_tild_alt

% These vectors should output all zeroes
display(Ci_tild - Ci_tild_alt)
display(Gi_tild - Gi_tild_alt)
display(Bi_tild - Bi_tild_alt)
```

```
Ci_tild_alt =

[ 0, 0,     0, -2.306, -1.162]
[ 0, 0, 2.559,  3.994, -2.013]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]
[ 0, 0,     0,      0,      0]


Gi_tild_alt =

      0
  196.2
      0
 -12.26
 -12.26


Bi_tild_alt =

[  0,  0]
[  0,  0]
[  1,  0]
[  0,  1]
[ -1, -1]


ans =

[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0]


ans =
```

```
 0
 0
 0
 0
 0


ans =

[ 0, 0]
[ 0, 0]
[ 0, 0]
[ 0, 0]
[ 0, 0]
```

Published with MATLAB® R2019a

ME 292B | HW # 1

2(a). From comparing the geometries of Figure 1a and Figure 1b,

$q(1) = \tilde{q}(1) = x$, $q(2) = \tilde{q}(2) = y$, $q(5) = \tilde{q}(5) = \theta_3 = q_3$

However, $q(3) \neq \tilde{q}(3)$ and $q(4) \neq \tilde{q}(4)$, but we can relate these variables by,

$\theta_1 = q_1 + q_3 - \pi$ and $\theta_2 = q_2 + q_3 - \pi$

- To incorporate all of these equations, we can write a configuration variable transformation as,

$\tilde{q} = Tq + d$ or

$$\underbrace{\begin{bmatrix} x \\ y \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}}_{\tilde{q}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{T} \underbrace{\begin{bmatrix} x \\ y \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}}_{q} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ -\pi \\ -\pi \\ 0 \end{bmatrix}}_{d}$$

* Note that T is invertible and $q = T^{-1}\tilde{q} - T^{-1}d$

2(b). To find mapping between $\dot{q}$ and $\dot{\tilde{q}}$, differentiate the equations from 2(a) and find that,

$\dot{q}(1) = \dot{\tilde{q}}(1) = \dot{x}$, $\dot{q}(2) = \dot{\tilde{q}}(2) = \dot{y}$, $\dot{q}(5) = \dot{\tilde{q}}(5) = \dot{\theta}_3 = \dot{q}_3$

and, $\dot{\theta}_1 = \dot{q}_1 + \dot{q}_3$, $\dot{\theta}_2 = \dot{q}_2 + \dot{q}_3$

$\therefore$ $\dot{\tilde{q}} = T\dot{q} + d$ where T is the same as 2(a) and $d = [0; 0; 0, 0, 0]^T$

- Now differentiate again to find,

$\ddot{q}(1) = \ddot{\tilde{q}}(1) = \ddot{x}$, $\ddot{q}(2) = \ddot{\tilde{q}}(2) = \ddot{y}$, $\ddot{q}(5) = \ddot{\tilde{q}}(5) = \ddot{\theta}_3 = \ddot{q}_3$

$\ddot{\theta}_1 = \ddot{q}_1 + \ddot{q}_3$, $\ddot{\theta}_2 = \ddot{q}_2 + \ddot{q}_3$

$\therefore$ $\ddot{\tilde{q}} = T\ddot{q} + d$ where T is the same as 2(a) and $d = [0, 0, 0, 0, 0]^T$

2(e). Start with given equation,

$$\cancel{\tfrac{1}{2}}\dot{q}^T D(q)\dot{q} = \cancel{\tfrac{1}{2}}\dot{\tilde{q}}^T \tilde{D}(\tilde{q})\dot{\tilde{q}}$$

- plug in expression for $\dot{\tilde{q}}$,

$$\dot{q}^T D(q)\dot{q} = (T\dot{q})^T \tilde{D}(\tilde{q})(T\dot{q})$$

- Distribute transpose,

$$\dot{q}^T \underline{D(q)}\dot{q} = \dot{q}^T \underline{T^T \tilde{D}(\tilde{q}) T}\dot{q}$$

- Isolate terms inbetween $\dot{q}^T$ and $\dot{q}$ for each side of the equation

$$D(q) = T^T \tilde{D}(\tilde{q}) T$$

- Multiply by inverse of T^T and T

$$(T^T)^{-1} D(q) T^{-1} = \cancel{(T^T)^{-1} T^T} \tilde{D}(\tilde{q}) \cancel{T T^{-1}}$$

$$\therefore \boxed{\tilde{D}(\tilde{q}) = (T^T)^{-1} \cdot D(q) \cdot T^{-1}}$$

2(F). Start by rearranging equation (2) as,

$$D(q)\ddot{q} = B(q)u - C(q,\dot{q})\dot{q} - G(q)$$

- Next, introduce equation (4) with results from 2(e) plugged in,

$$((T^T)^{-1} \cdot D(q) \cdot T^{-1})\ddot{\tilde{q}} + \tilde{C}\dot{\tilde{q}} + \tilde{G} = \tilde{B}u$$

- introduce expressions for $\ddot{\tilde{q}}$ and $\dot{\tilde{q}}$,

$$((T^T)^{-1} \cdot D(q) \cdot \cancel{T^{-1}})(\cancel{T}\ddot{q}) + \tilde{C}(T\dot{q}) + \tilde{G} = \tilde{B}u$$

- Rearrange,

$$(T^T)^{-1} \cdot D(q)\ddot{q} = -\tilde{C}(T\dot{q}) - \tilde{G} + \tilde{B}u$$

- Multiply by T^T,

$$D(q)\ddot{q} = T^T(-\tilde{C}(T\dot{q}) - \tilde{G} + \tilde{B}u)$$

- Continued...

- Now set eqtn (2) and modified eqtn (4) equal to another, eliminating their left hand terms,

$$B(q)u - C(q,\dot{q})\dot{q} - G(q) = T^T(-\tilde{C}(T\dot{q}) - \tilde{G} + \tilde{B}u)$$

- Multiply by $(T^T)^{-1}$ and distribute,

$$(T^T)^{-1} \cdot B(q)u - (T^T)^{-1} \cdot C(q,\dot{q})\dot{q} - (T^T)^{-1} \cdot G(q)$$
$$= -\tilde{C}(T\dot{q}) - \tilde{G} + \tilde{B}u$$

- By setting $\dot{q}$ terms equal, we get,

$$-(T^T)^{-1} C \dot{q} = -\tilde{C} T \dot{q}$$

or equivalently,

$$\boxed{\tilde{C} = (T^T)^{-1} \cdot C \cdot T^{-1}}$$

- By setting singular (G) terms equal, we get,

$$-(T^T)^{-1} G(q) = -\tilde{G}$$

or equivalently,

$$\boxed{\tilde{G} = (T^T)^{-1} \cdot G}$$

- By setting u terms equal, we get,

$$(T^T)^{-1} \cdot B \cdot u = \tilde{B}u$$

or equivalently,

$$\boxed{\tilde{B} = (T^T)^{-1} \cdot B}$$