Task 1

a)

For this task, I got a lot of help from https://www.math24.net/double-pendulum/.

```
clear all
clc

% Parameters
syms m M g L F real
% Variables
syms x  theta_1  theta_2 real
syms dx dtheta_1 dtheta_2 real

% Define symbolic variable q for the generalized coordinates
% x, theta1 and theta2
q = [x; theta_1; theta_2];
% Define symbolic variable dq for the derivatives
% of the generalized coordinates
dq = [dx; dtheta_1; dtheta_2];
% Write the expressions for the positions of the masses
p{1} = [q(1)+sin(theta_1)*L ;
        -cos(theta_1)*L];
p{2} = p{1} + [sin(theta_2)*L ;
               -cos(theta_2)*L];

% Kinetic energy of the cart
T = (m/2)*dq(1)^2;
% For loop that adds the kinetic energies of the masses
for k = 1:length(p)
    dp{k} = jacobian(p{k},q)*dq; % velocity of mass k
    T = T + (M/2)*(dp{k}(1)^2+dp{k}(2)^2); % add kinetic energy of mass k
end
T = simplify(T);

% Potential energy of the cart
V = 0;
% For loop that adds the potential energies of the masses
for k = 1:length(p)
    V = V + m*g*p{k}(2); % add potential energy of mass k
end
V = simplify(V);

% Generalized forces
Q = [F; 0; 0];

% Lagrangian
Lag = T - V;

Lag_q = simplify(jacobian(Lag,q)).';
Lag_qdq = simplify(jacobian(Lag_q.',dq));
Lag_dq = simplify(jacobian(Lag,dq)).';
Lag_dqdq = simplify(jacobian(Lag_dq.',dq));

% The equations have the form W*q_dotdot = RHS, with
W = Lag_dqdq;
RHS = Q + simplify(Lag_q - Lag_qdq*dq);

```

```
state = [q;dq];
param = [m;M;L;g];

matlabFunction(p{1},p{2}, 'file','PendulumPosition','vars',{state, param});
matlabFunction(W,RHS, 'file','PendulumODEMatrices','vars',{state,F,param});
```

b)

```
function [dstate] = PendulumDynamics(t, x, parameters)
%PENDULUMODEMATRICES
%    [W,RHS] = PENDULUMODEMATRICES(IN1,F,IN3)

%    This function was generated by the Symbolic Math Toolbox version 8.5.
%    22-Feb-2021 19:18:47
% state = [q;dq];
% param = [m;M;L;g];
q = x(1:3);
dq = x(4:6);
F= -10*x(1) - x(4);
[W, RHS] = PendulumODEMatrices(x, F, parameters);
dstate = [dq; W\RHS];
end
```

c)

After a lot of time spent debugging, I finally got the simulation to work. The issue was my inconsistent reference frame, sometimes y was up sometimes it was down.

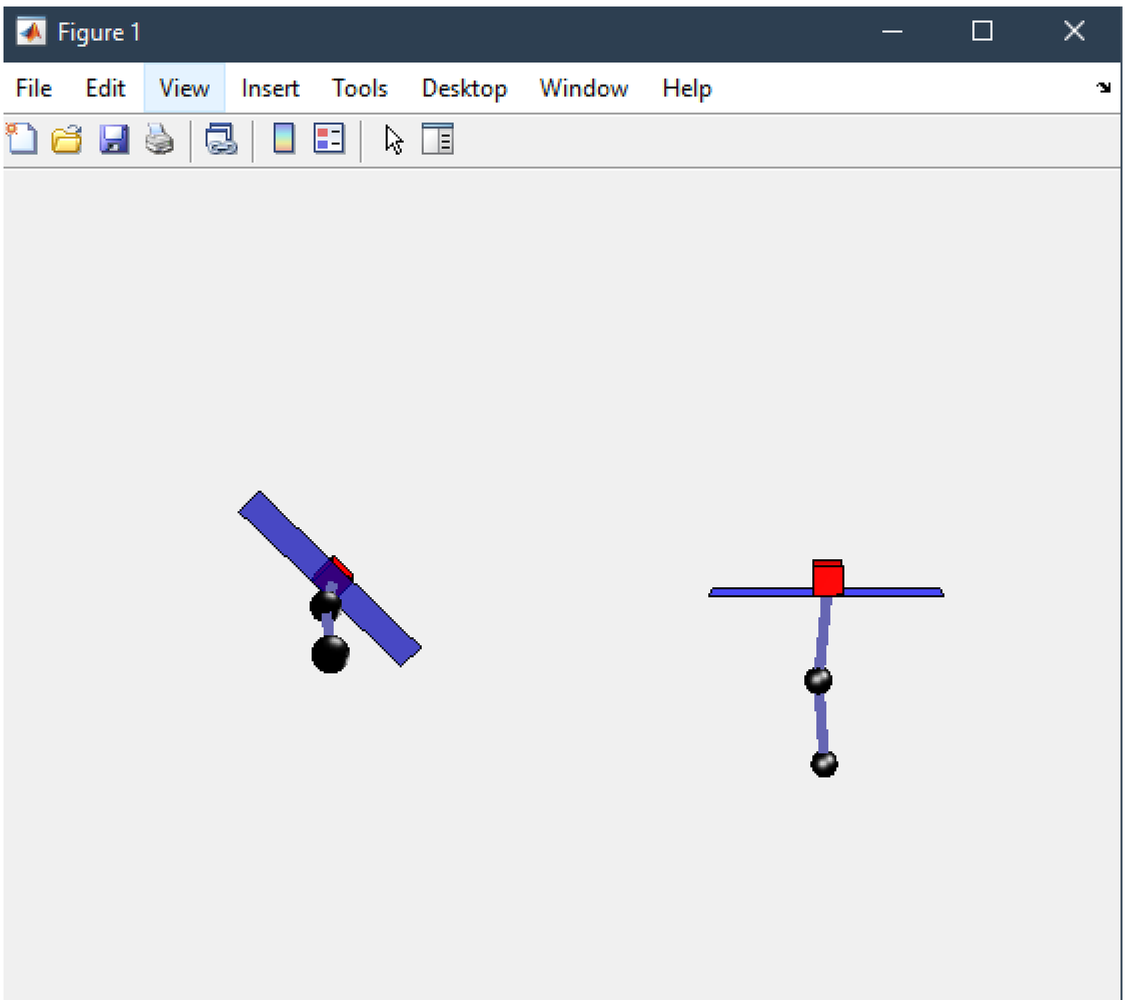
The simulation looks a bit weird as there are two pendulums seen from different angles, I don't know if that was the intention.

The system behaves as expected. It is chaotic in the beginning, but as there is a dampening part it goes toward equilibrium after a while.

```
clear all
close all
clc

% Parameters and initial states
tf = 45;
parameters = [1; 0.1; 1; 9.81];
state = [1; pi/4; 0; 0; 0; 0];

% Simulation
try

    %%%%%% MODIFY THE CODE AS YOU SEE FIT

    [tsim,xsim] = ode45(@(t,x)PendulumDynamics(t, x, parameters),[0,tf],state);

catch message
    display('Your simulation failed with the following message:')
    display(message.message)
    display(' ')

    % Assign dummy time and states if simulation failed
    tf = 0.1;
```

```
    tsim = [0,tf];
    xsim = 0;
end

%% 3D animation
DoublePlot = true;
FS = 30;
scale = 0.1;

% Create Objects
% Cube
vert{1} = 3*[ -1, -1, 0;  %1
               1, -1, 0;  %2
               1,  1, 0;  %3
              -1,  1, 0;  %4
              -1, -1, 2;  %5
               1, -1, 2;  %6
               1,  1, 2;  %7
              -1,  1, 2]/2; %8
fac{1} = [1 2 3 4;
          5 6 7 8;
          1 4 8 5;
          1 2 6 5;
          2 3 7 6;
          3 4 8 7];
Lrail = 1.2*max(abs(xsim(:,1)))/scale;
% Rail
a = 1.5;
vert{2} = [-Lrail,-a,-0.1;
           -Lrail, a,-0.1;
            Lrail, a,-0.1;
            Lrail,-a,-0.1];
fac{2} = [1,2,3,4];
% Sphere
[X,Y,Z] = sphere(20);
[fac{3},vert{3},c] = surf2patch(3*X/2,3*Y/2,3*Z/2);
% Animation
tic
t_disp = 0;
SimSpeed = 1;
 while t_disp < tf/SimSpeed
    % Interpolate state
    x_disp   = interp1(tsim,xsim,SimSpeed*t_disp)';

    % Unwrap state. MODIFY
    x = x_disp(1); % position cart
    p1 = 1*[x+sin(x_disp(2));-cos(x_disp(2))]; % position 1st ball
    p2 = p1 + 1*[sin(x_disp(3));-cos(x_disp(3))]; % position 2nd ball

    % Input argument for DrawPendulm
    pos_disp = [x(1);p1(1);0;p1(2);p2(1);0;p2(2)];

    figure(1);clf;hold on
    if DoublePlot
        subplot(1,2,1);hold on
        DrawPendulum( pos_disp, vert, fac, scale);
        campos(scale*[15    15     -70])
```

```
        camtarget(scale*[0,0,1.5])
        camva(30)
        camproj('perspective')
        subplot(1,2,2);hold on
    end
    DrawPendulum( pos_disp, vert, fac, scale);
    campos(scale*[1    70     20])
    camtarget(scale*[0,0,1.5])
    camva(30)
    camproj('perspective')
    drawnow
    if t_disp == 0
        display('Hit a key to start animation')
        pause
        tic
    end
    t_disp = toc;
end
```



Task 2

a)

The position of the ball is:

$$\boldsymbol{p} = x\vec{b_1} + R\vec{b_2} = \begin{bmatrix} x\cos(\theta) - R\sin(\theta) \\ -x\sin(\theta) + R\cos(\theta) \end{bmatrix}$$

b)

Inspired by the previous task, the total movement of the bal can be written as:

$$\text{jac}(\boldsymbol{p}, \boldsymbol{q}) * \dot{\boldsymbol{q}}$$

The angular velocity can be written as:

$$(\begin{bmatrix} \frac{1}{R} & 1 \end{bmatrix} * \dot{\boldsymbol{q}})^2$$

c)

The kinetic energy from translation of the mass center is:

$$T_{trans}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}M(\text{jac}(\boldsymbol{p}, \boldsymbol{q}) * \dot{\boldsymbol{q}})^T * (\text{jac}(\boldsymbol{p}, \boldsymbol{q}) * \dot{\boldsymbol{q}})$$

The kinetic energy from rotation of the mass center is:

$$T_{rot}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{5}MR^2(\begin{bmatrix} \frac{1}{R} & 1 \end{bmatrix} * \dot{\boldsymbol{q}})^2$$

d)

The kinetic energy of the rail is:

$$T_{rail}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}J\dot{\theta}^2$$

Finally the total kinetic energy is $T = T_{trans} + T_{rot} + T_{rail}$.

e)

```
clear all
clc

% Parameters
syms J M R g To real
% Variables
syms x theta  real
syms dx dtheta real

% Define symbolic variable q for the generalized coordinates
% x and theta
q  = [x; theta];
% Define symbolic variable dq for the derivatives
% of the generalized coordinates
dq = [dx; dtheta];
% Write the expressions for the position of
% the center of the ball:
p = [x*cos(theta)-R*sin(theta);
     x*sin(theta)+R*cos(theta)];
```

```

% Kinetic energy
T = 0.5*J*dtheta^2; % kinetic energy of beam

dp = jacobian(p,q)*dq; % linear velocity of ball
T  = T + 0.5*M*(dp'*dp); % add linear kinetic energy of ball

I     = (2/5)*M*R^2; % inertia in rotation of ball
omega = [1/R 1]*dq; % angular velocity of ball

T  = T + 0.5*I*omega^2; % add rotational kinetic energy of ball

T = simplify(T);

% Potential energy
V = p(2);

% Generalized forces
Q = [0; To];

% Lagrangian
Lag = T - V;

Lag_q = simplify(jacobian(Lag,q)).';
Lag_qdq = simplify(jacobian(Lag_q.',dq));
Lag_dq = simplify(jacobian(Lag,dq)).';
Lag_dqdq = simplify(jacobian(Lag_dq.',dq));

% The equations have the form W*q_dotdot = RHS, with
W = Lag_dqdq;
RHS = Q + simplify(Lag_q - Lag_qdq*dq);

state = [q;dq];
param = [J; M; R; g];

matlabFunction(p, 'file','BallPosition','vars',{state,param});
matlabFunction(W,RHS, 'file','BallAndBeamODEMatrices','vars',{state,To,param})
    ;
```

f)

```
function [dstate] = BallAndBeanDynamics(t, x, parameters)
%PENDULUMODEMATRICES
%    [W,RHS] = PENDULUMODEMATRICES(IN1,F,IN3)
%    This function was generated by the Symbolic Math Toolbox version 8.5.
%    22-Feb-2021 19:18:47
% state = [q;dq];
% param = [J; M; R; g];
q = x(1:2);
dq = x(3:4);
F= 200*(x(1) - x(2)) + 70*(x(3) - x(4));
[W, RHS] = BallAndBeamODEMatrices(x, F, parameters);
dstate = [dq; W\RHS];
end
```

g)

The simulation works, the controller appear to work, but the physics are not reasonable as the ball stay attached to the beam, even when the beam is rotated 180°.

```
clear all
close all
clc

% Parameters and initial states
tf = 15;
parameters = [1; 10; 0.25; 9.81];
state = [1;0;0;0];

% Simulation
try

    %%%%%% MODIFY THE CODE AS YOU SEE FIT

    [tsim,xsim] = ode45(@(t,x)BallAndBeamDynamics(t, x, parameters),[0,tf],
       state);

catch message
    display('Your simulation failed with the following message:')
    display(message.message)
    display(' ')

    % Assign dummy time and states if simulation failed
    tf = 0.1;
    tsim = [0,tf];
    xsim = 0;
end

%% 3D animation
DoublePlot = true;
scale = 0.25;
FS = 30;
ball_radius = 0.25;

% Create Objects
% Rail
Lrail = 2;
a = ball_radius;
vert{1} = [-Lrail,-a, 0;
           -Lrail, a, 0;
            Lrail, a, 0;
            Lrail,-a, 0];
fac{1} = [1,2,3,4];
% Sphere
[X,Y,Z] = sphere(20);
[fac{2},vert{2},c] = surf2patch(X,Y,Z);

% Animation
tic
t_disp = 0;
SimSpeed = 1;
while t_disp < tf/SimSpeed
    % Interpolate state
    x_disp   = interp1(tsim,xsim,SimSpeed*t_disp)';
```

```
    % Unwrap state. MODIFY
    theta = x_disp(2); % beam angle
    pos = x_disp(1)*[cos(theta);sin(theta)] + ball_radius*[-sin(theta);cos(theta)];
    pos = [pos(1);0;pos(2)]; % ball position

    figid = figure(1);clf;hold on
    if DoublePlot
        subplot(1,2,1);hold on
        DrawBallAndBeam(pos, theta, vert, fac, xsim, ball_radius);
        campos(scale*[10    10     20])
        camtarget(scale*[0,0,1.5])
        camva(30)
        camproj('perspective')
        subplot(1,2,2);hold on
    end
    DrawBallAndBeam(pos, theta, vert, fac, xsim, ball_radius);
    campos(0.4*scale*[1    70     20])
    camtarget(scale*[0,0,1.5])
    camva(30)
    camproj('perspective')
    drawnow

    if t_disp == 0
        display('Hit a key to start animation')
        pause
        tic
    end
    t_disp = toc;
end
```

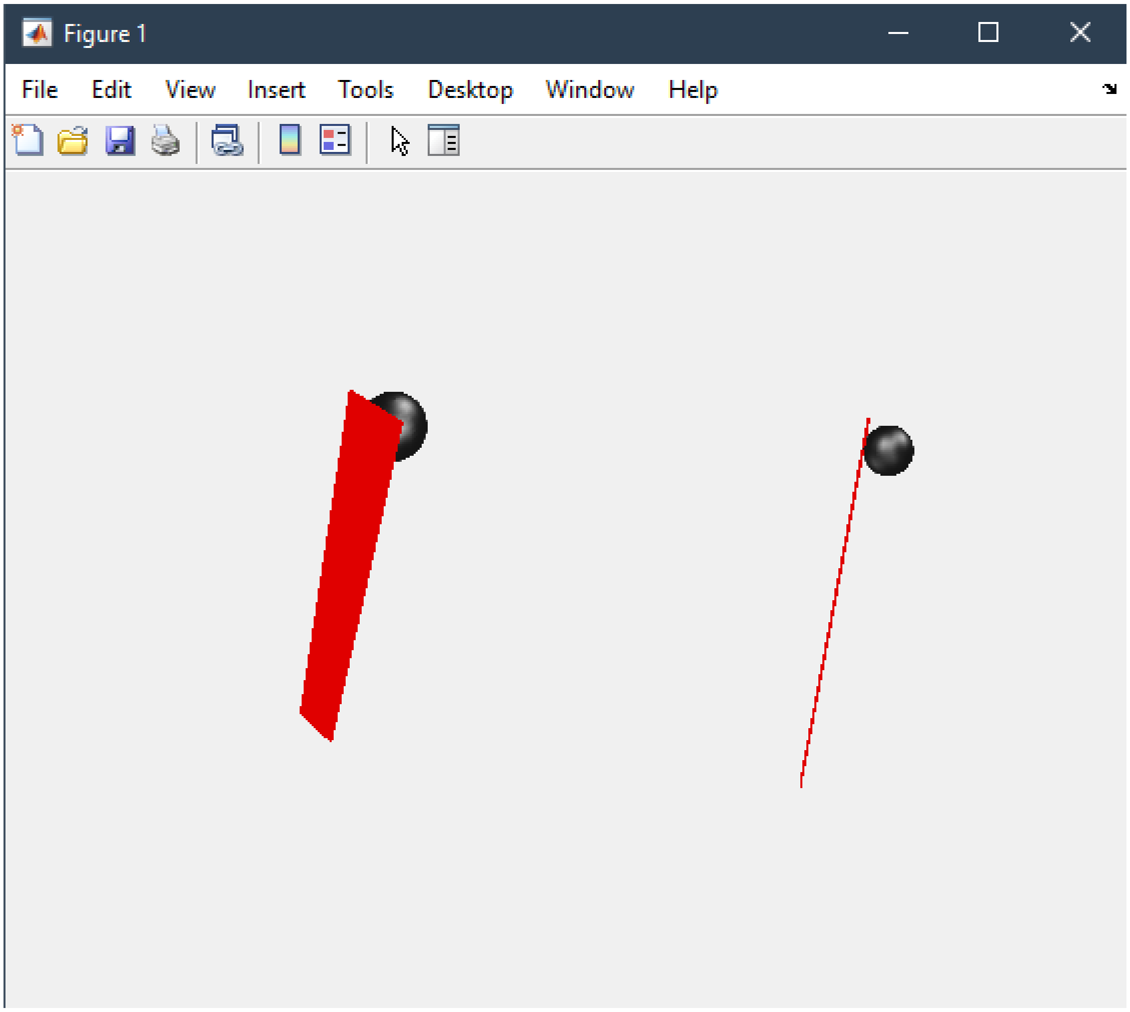


Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help