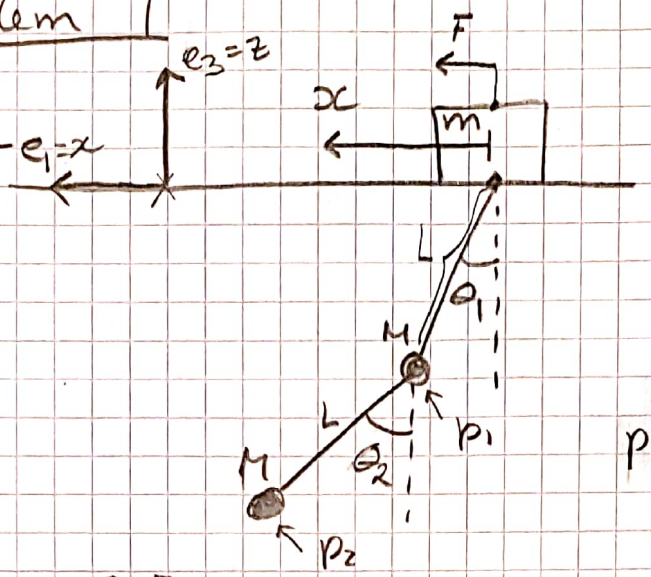
TTK4130 Assignment 4 Eirik Falch

Problem 1



$m = 1\,kg$

$M = 1\,kg$

$L = 1\,m$

$$p_1 = \begin{bmatrix} x + L\sin\theta_1 \\ -L\cos\theta_1 \end{bmatrix}$$

a) $$q = \begin{bmatrix} x \\ \theta_1 \\ \theta_2 \end{bmatrix} \qquad p_0 = \begin{bmatrix} x \\ 0 \end{bmatrix} \qquad p_2 = \begin{bmatrix} x + L\sin\theta_1 + L\sin\theta_2 \\ -L\cos\theta_1 - L\cos\theta_2 \end{bmatrix}$$

$$T(q,\dot{q}) = \frac{1}{2} m \dot{x}^2 + \frac{1}{2} M \dot{p}_1^T \dot{p}_1 + \frac{1}{2} M \dot{p}_2^T \dot{p}_2$$

$$V(q) = Mg\, p_{1,z} + Mg\, p_{2,z}$$

$p_{1,z}$, $p_{2,z}$: second element, i.e. height

$$\mathcal{L}(q,\dot{q}) = T(q,\dot{q}) - V(q)$$

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q} = Q$$

$$\frac{d}{dt} p_1(q) = \frac{\partial p_1}{\partial q} \cdot \dot{q} \quad , \quad \frac{d}{dt} p_2(q) = \frac{\partial p_2}{\partial q} \cdot \dot{q}$$

b) $$F = -10x - \dot{x} \rightarrow Q = \begin{bmatrix} -10x - \dot{x} \\ 0 \\ 0 \end{bmatrix}$$

The simulation shows very sensible results. The pendulums swing in the manner one would expect, and the box glides back and forth on the board.

```
%% Symbolic variables
% Generalized coordinates
syms x real;
syms theta1 real;
syms theta2 real;

% Time derivative of generalized coords
syms dx real;
syms dtheta1 real;
syms dtheta2 real;

q = [x; theta1; theta2];  % generalized coords. vector
dq = [dx; dtheta1; dtheta2];

% Parameters
syms m real;
syms M real;
syms L real;

% Constants
g = 9.81;  % acceleration due to gravity

% Position of masses M
p0 = [x; 0];
p1 = p0 + [L*sin(theta1); -L*cos(theta1)];
p2 = p1 + [L*sin(theta2); -L*cos(theta2)];

% Time derivaitve of position of masses M
dp1 = jacobian(p1, q) * dq;
dp2 = jacobian(p2, q) * dq;


%% Lagrange
% Kinetic energy
T = 1/2 * m * dx^2 + 1/2 * M * (dp1') * dp1 + 1/2 * M * (dp2') * dp2;
T = simplify(T);

% Potential energy
V = M * g * p1(2) + M * g * p2(2);
V = simplify(V);

% External force
F = -10*x - dx;
Q = [F; 0; 0];

% Lagrange function
L = T - V;


% Lagrange equation  (see (2) in task text)
W_hessian = hessian(L, dq);
other_vector = Q + jacobian(L,q)' - jacobian(W_hessian * dq, q) * dq;


%% Export function
state = [x; theta1; theta2; dx; dtheta1; dtheta2];
params = {L, M, m};

```

```
59 syms t real;  % supporting variagble for function creation
60 matlabFunction(W_hessian, 'File', 'prob1_W_hessian');
61 matlabFunction(other_vector, 'File', 'prob1_other_vector');
62 % Can't get parameters to work :(       , 'Vars', {t, state, params}
63
```

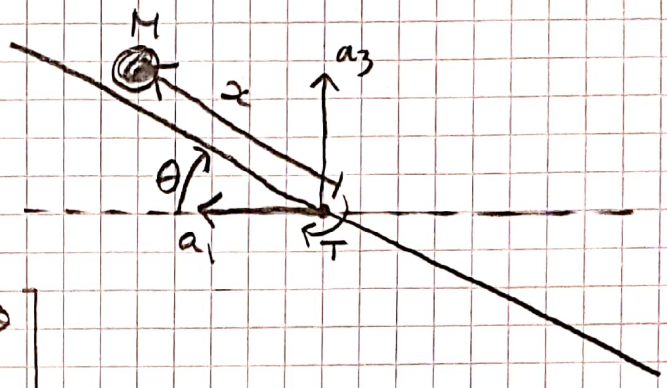
```
clear all
close all
clc

%% Declarations
% Parameters
L = 1;
M = 1;
m = 1;

% Initial position
x0 = 0;
theta1_0 = pi/4;
theta2_0 = pi/2;
q0 = [x0; theta1_0; theta2_0];

% Initial velocity
dq0 = zeros(3,1);

% Initial state
%      1  2       3       4   5        6
% x = [x; theta1; theta2; dx; dtheta1; dtheta2];
state = [q0; dq0];


%% Simulation
tf = 45;

% Function declarations
W = @(x) prob1_W_hessian(L,M,m,x(2),x(3));
other = @(x) prob1_other_vector(L,M,x(5),x(6),x(4),x(2),x(3),x(1));
simFunc = @(t, x) [x(4:6); W(x) \ other(x)];

[time, statetraj] = ode45(simFunc, [0 tf], state);


%% 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(statetraj(:,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
    state_animate = interp1(time,statetraj,SimSpeed*t_disp)';

    x = state_animate(1);
    theta1 = state_animate(2);
    theta2 = state_animate(3);

    p0 = [x; 0];  % box
    p1 = p0 + [L*sin(theta1); -L*cos(theta1)];  % sphere 1
    p2 = p1 + [L*sin(theta2); -L*cos(theta2)];  % sphere 2

    % shift coords
    p0_3d = [-p0(1); 0; p0(2)];
    p1_3d = [-p1(1); 0; p1(2)];
    p2_3d = [-p2(1); 0; p2(2)];

    % Input argument for DrawPendulm
    pos_disp = [p0_3d(1); p1_3d; p2_3d];

    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

```

Problem 2

$J = 1\,kg\,m^2$, $M = 10kg$, $R = 0{,}25\,m$

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix}$$



$$p = \begin{bmatrix} x\cos\theta \\ 0 \\ x\sin\theta \end{bmatrix}$$

Rotation velocity of ball

$$\omega = \dot{\theta} + \frac{\dot{x}}{R} \quad , \quad J_{ball} = \frac{2}{5}MR^2$$

$$T_{beam} = \frac{1}{2} J \dot{\theta}^2$$

$$T_{ball,x} = \frac{1}{2} M \dot{x}^2$$

$$T_{ball,\omega} = \frac{1}{2}\frac{2}{5} MR^2 \left(\dot{\theta} + \frac{\dot{x}}{R}\right)^2$$

$$\Rightarrow T(q,\dot{q}) = \frac{1}{2} M \dot{x}^2 + \frac{1}{2} J \dot{\theta}^2 + \frac{1}{2}\frac{2}{5} MR^2 \left(\dot{\theta} + \frac{\dot{x}}{R}\right)^2$$

$$V(q) = Mg\,p_3 = Mg\,x\sin\theta$$

$$\Rightarrow \mathcal{L} = T - V$$

$$\text{Torque} = 200(x - \theta) + 70(\dot{x} - \dot{\theta})$$

$$\Rightarrow Q = \begin{bmatrix} 0 \\ \text{Torque} \end{bmatrix}$$

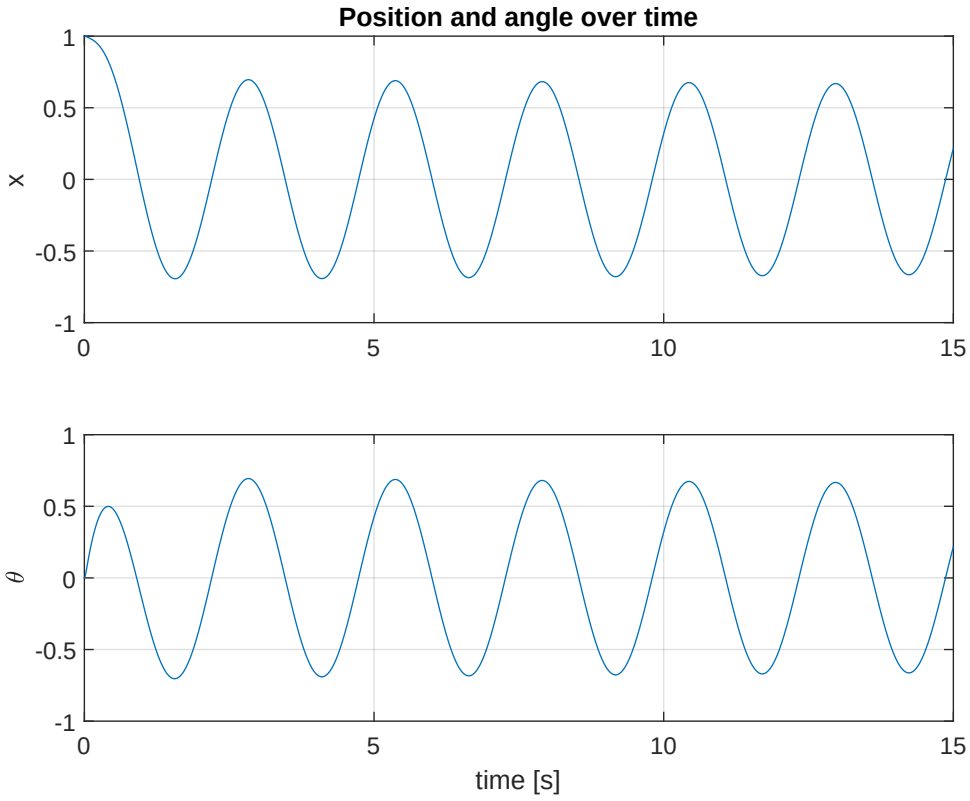
The simulation shows how the PD-controller tries to keep the ball on the beam by rotating the beam.

The results are reasonable. See plot for coordinates vs. time.

```
%% Symbolic variables
% Generalized coordinates
syms x real;
syms theta real;

% Time derivative of generalized coords
syms dx real;
syms dtheta real;

q = [x; theta];  % generalized coords. vector
dq = [dx; dtheta];

% Parameters
syms R real;
syms M real;
syms J real;

% Constants
g = 9.81;  % acceleration due to gravity


% Position of ball in frame a
p_a = [x*cos(theta); 0; x*sin(theta)];

%% Lagrange
% Kinetic energy
T = 1/2 * M * dx^2 + 1/2 * J * dtheta^2 + 1/2 * 2/5 * M * R^2 * (dtheta + dx/R)^2;
T = simplify(T);

% Potential energy
V = M * g * p_a(3);
V = simplify(V);

% External torque
Torque = 200 * (x - theta) + 70 * (dx - dtheta);
Q = [0; Torque];

% Lagrange function
L = T - V;


% Lagrange equation  (see (2) in task text)
W_hessian = hessian(L, dq);
other_vector = Q + jacobian(L,q)' - jacobian(W_hessian * dq, q) * dq;


%% Export function
state = [q; dq];
params = [J; M; R];

matlabFunction(W_hessian, 'File', 'prob2_W_hessian', 'Vars', {state, params});
matlabFunction(other_vector, 'File', 'prob2_other_vector', 'Vars', {state, params});
% Can't get parameters to work :(       , 'Vars', {state, params}

```

```
clear all
close all
clc


%% Declarations
% Parameters
J = 1;
M = 1;
R = 0.25;

% Initial position
x0 = 1;
theta_0 = 0;
q0 = [x0; theta_0];

% Initial velocity
dq0 = zeros(2,1);

% Initial state
%      1  2      3   4
% x = [x; theta; dx; dtheta];
state = [q0; dq0];


%% Simulation
tf = 15;

% Function declarations
W = @(x) prob2_W_hessian(x,[J,M,R]');
other = @(x) prob2_other_vector(x, [J,M,R]');
simFunc = @(t, x) [x(3:4); W(x) \ other(x)];

[tsim, xsim] = ode45(simFunc, [0 tf], state);


%% 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
    x = x_disp(1);
    theta = x_disp(2); % beam angle
    pos = x*[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

```


Position and angle over time
x
θ
time [s]
1
0.5
0
-0.5
-1
0
5
10
15