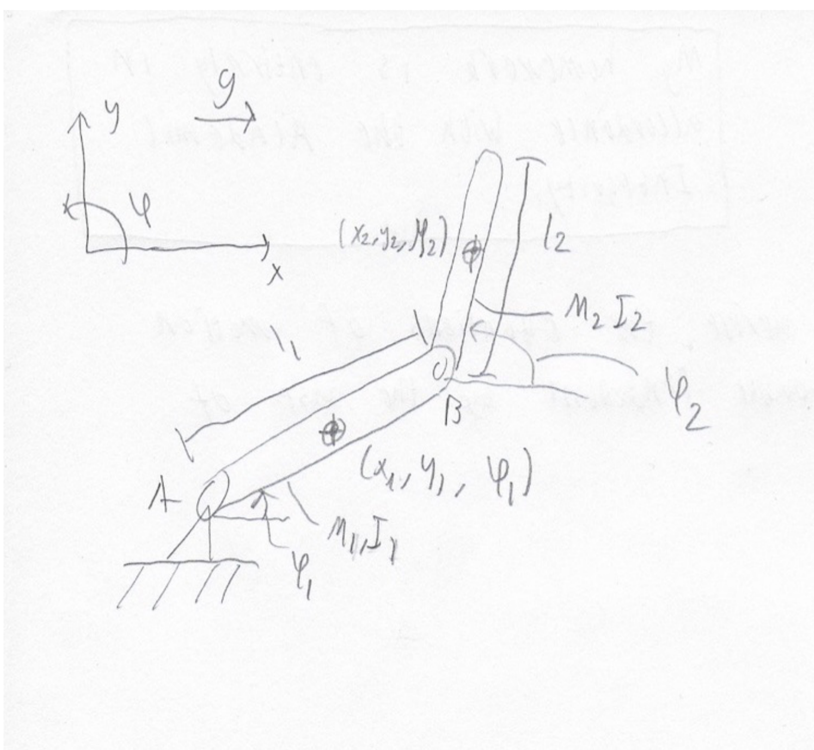
My homowork is entirely in accordance with the Academic Integrity.

Multibody Dynamics B - Assignment 2

July 4, 2019

a)

In this assignment we derive the equations of motion and constraints for a double pendulum using the systematic approach. This is done by defining the rigid bodies and formulate the constraints, after this is done we can apply an algorithm that derives the equations of motions. The double pendulum is described in the figure below.



When using the systematic approach to find DAEs we use the principal of virtual power on a system in dynamic equilibrium in accompany with Lagrange multipliers for the constraints on the system to get the constrained equations of motion, which are

$$F_i - Mi, j\ddot{x}_j - C_{k,i}\lambda_k = 0$$

that is our virtual power expression is

$$\delta P = (F_i - M_{ij}\ddot{x}_j) * \delta\dot{x}_i - \lambda_k C_{k,i} * \delta\dot{x}_i$$

These differential equations describe the dynamic equilibrium of the system. Furthermore we have to add the constraint equations, 'the glue', and since our unknowns are the accelerations, $\ddot{x}_j$, we have to differentiate the constraint equations twice with respect to time

$$\dot{C}_k = C_{k,i}\dot{x}_i = 0$$

$$\ddot{C}_k = C_{k,j}\ddot{x}_j + C_{k,ij}\dot{x}_i\dot{x}_j = 0$$

we can combine these equations to get our DAEs which can be written in matrix form

$$\begin{pmatrix} M_{ij} & C_{k,i} \\ C_{k,j} & 0_{kk} \end{pmatrix} \begin{pmatrix} \ddot{x}_j \\ \lambda_k \end{pmatrix} = \begin{pmatrix} F_i \\ -C_{k,ij}\dot{x}_i\dot{x}_j \end{pmatrix}$$

where $F_i = (F_{x1}, F_{y1}, M_1, F_{x2}, F_{y2}, M_2) = (m_1 * g, 0, 0, m_2 * g, 0, 0)$, $M_{ij} = diag(m_1, m_1, I_1, m_2, m_2, I_2)$, where $I_1 = I_2 = \frac{1}{12} * m_1 * l_1^2$ since we assume a slender beam, $\ddot{x}_j = (\ddot{x}_1, \ddot{y}_1, \ddot{\varphi}_1, \ddot{x}_2, \ddot{y}_2, \ddot{\varphi}_2)$, $\lambda_k = (\lambda 1, \lambda 2, \lambda 3, \lambda 4)$ and $k = 1, .., 4$ and $i, j = 1, .., 6$

Now we have two rigid bodies and our constraint equations are

$$X_A = x_1 - l_1/2 * cos(\varphi_1)$$

$$Y_A = y_1 - 1_1/2 * sin(\varphi_1)$$

$$X_B = (x_2 - l_2/2 * cos(\varphi_2)) - (x_1 + l_1/2 * cos(\varphi_1))$$

$$Y_B = (y_2 - l_2/2 * sin(\varphi_2)) - (y_1 + l_1/2 * sin(\varphi_1))$$

Since bar 1 is connected to the ground by a cylindrical joint and bar 1 and 2 are connected together by a cylindrical joint. We put these constraints in a zero-delimited form where we have

$$C_k = \begin{pmatrix} x_1 - l_1/2 * cos(\varphi_1) \\ y_1 - 1_1/2 * sin(\varphi_1) \\ (x_2 - l_2/2 * cos(\varphi_2)) - (x_1 + l_1/2 * cos(\varphi_1)) \\ (y_2 - l_2/2 * sin(\varphi_2)) - (y_1 + l_1/2 * sin(\varphi_1)) \end{pmatrix} = \mathbf{0}$$

We use the Matlab, see Apendix A, to derive $C_{k,j}$ and $-C_{k,ij}\dot{x}_i\dot{x}_j$, notice that $C_{k,i} = C_{k,j}^T$, which gives us

$$C_{k,j} = \begin{pmatrix} 1 & 0 & 1/2 * l_1 * sin(\varphi_1) & 0 & 0 & 0 \\ 0 & 1 & -1/2 * l_1 * cos(\varphi_1) & 0 & 0 & 0 \\ -1 & 0 & 1/2 * l_1 * sin(\varphi_1) & 1 & 0 & 1/2 * l_2 * sin(\varphi_2) \\ 0 & -1 & -1/2 * l_1 * cos(\varphi_1) & 0 & 1 & -1/2 * l_2 * cos(\varphi_2) \end{pmatrix}$$

and

$$C_{k,ij}\dot{x}_i\dot{x}_j = \begin{pmatrix} 1/2 * l_1 * \dot{\varphi_1}^2 cos(\varphi_1) \\ 1/2 * l_1 * \dot{\varphi_1}^2 sin(\varphi_1) \\ 1/2 * l_1 * \dot{\varphi_1}^2 cos(\varphi_1) + 1/2 * l_2 * \dot{\varphi_2}^2 cos(\varphi_2) \\ 1/2 * l_1 * \dot{\varphi_1}^2 sin(\varphi_1) + 1/2 * l_2 * \dot{\varphi_2}^2 sin(\varphi_2) \end{pmatrix}$$

Now for b)-d) we solve

$$\begin{pmatrix} \ddot{x}_j \\ \lambda_k \end{pmatrix} = \begin{pmatrix} M_{ij} & C_{k,i} \\ C_{k,j} & 0_{kk} \end{pmatrix}^{-1} \begin{pmatrix} F_i \\ -C_{k,ij}\dot{x}_i\dot{x}_j \end{pmatrix}$$

with the help of Matlab.

b)

We put our initial values as $\varphi_1 = \varphi_2 = \dfrac{\pi}{2}$ and $\dot{\varphi}_1 = \dot{\varphi}_2 = 0 rad/s$. Matlab gives us:

$$\ddot{x}_j = \begin{pmatrix} 6.3064 m/s^2 \\ 0 m/s^2 \\ -21.0214 rad/s^2 \\ 10.5107 m/s^2 \\ 0 m/s^2 \\ 7.0071 rad/s^2 \end{pmatrix}$$

and

$$\lambda_k = \begin{pmatrix} 0.1690 N \\ 0 N \\ -0.0423 N \\ 0 N \end{pmatrix}$$

These results are the same as we got in Assignment 1, but with opposite signs, i.e. same values, opposite signs.
Now we see that the Lagrange multiplier have the same values but opposite signs of the reactive forces from Assignment 1. We can thus conclude that the Jacobian matrix $C_{k,i}$ represents the applied forces on the center of mass of the bodies for which the system is in equilibrium„ i.e. the static equilibrium holds.

$$F_i = C_{k,i} \lambda_k$$

c)

We put our initial values as $\varphi_1 = \varphi_2 = 0$ and $\dot{\varphi}_1 = \dot{\varphi}_2 = 0 rad/s$. Matlab gives us:

$$\ddot{x}_j = \begin{pmatrix} 0 m/s^2 \\ 0 m/s^2 \\ 0 rad/s^2 \\ 0 m/s^2 \\ 0 m/s^2 \\ 0 rad/s^2 \end{pmatrix}$$

and

$$\lambda_k = \begin{pmatrix} 1.1831 N \\ 0 N \\ 0.5915 N \\ 0 N \end{pmatrix}$$

These results are the same as we got in Assignment 1, but with opposite signs, i.e. same values, opposite signs.
Now we see that the Lagrange multiplier have the same values but opposite signs of the reactive forces from Assignment 1. We can thus conclude that the Jacobian matrix $C_{k,i}$ represents the applied forces on the center of mass of the bodies for which the system is in equilibrium, i.e. the static equilibrium holds.

$$F_i = C_{k,i} \lambda_k$$

d)

We put our initial values as $\varphi_1 = \varphi_2 = 0$ and $\dot{\varphi}_1 = \dot{\varphi}_2 = \pi rad/s$. Matlab gives us:

$$\ddot{x}_j = \begin{pmatrix} 2.9609 m/s^2 \\ 0 m/s^2 \\ 0 rad/s^2 \\ 8.8826 m/s^2 \\ 0 m/s^2 \\ 0 rad/s^2 \end{pmatrix}$$

and

$$\lambda_k = \begin{pmatrix} 0.4689 \\ 0 \\ 0.0559 \\ 0 \end{pmatrix}$$

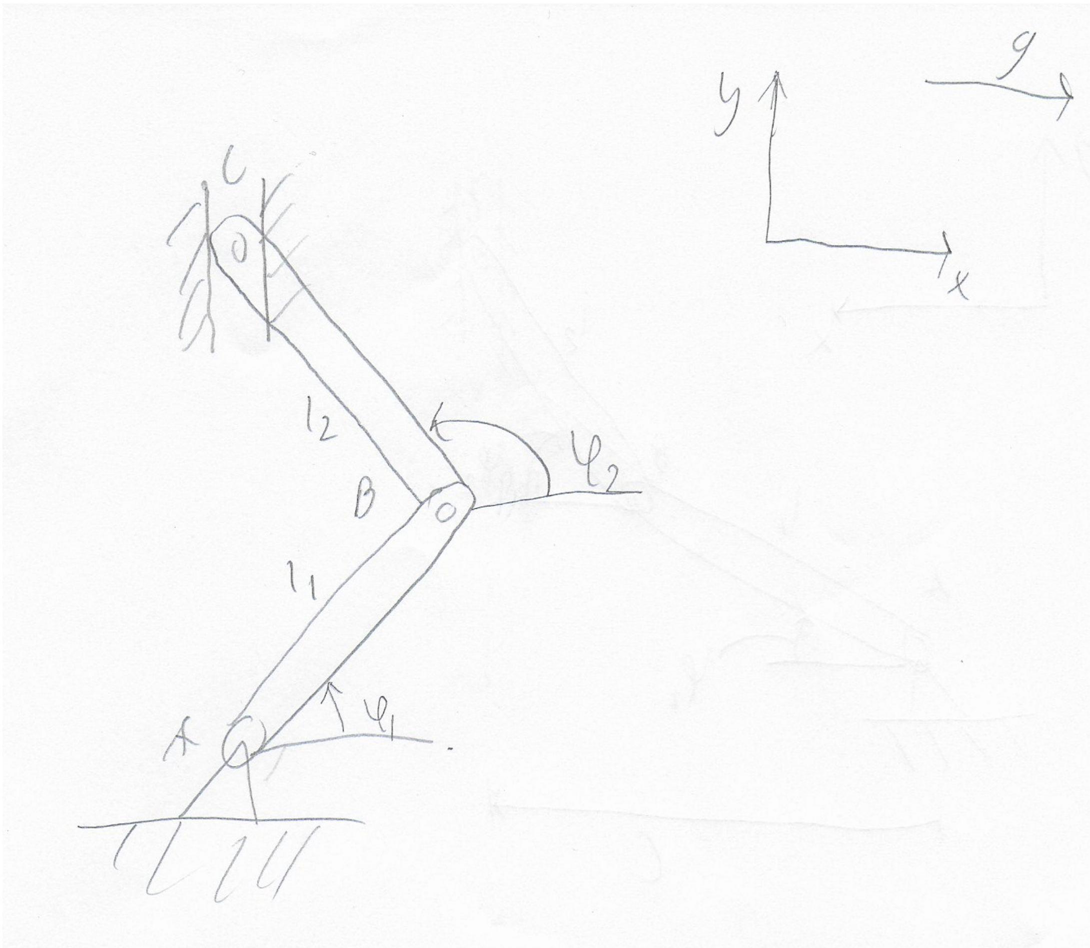
These results are the same as we got in Assignment 1, but with opposite signs, i.e. same values, opposite signs.
Now we see that the Lagrange multiplier have the same values but opposite signs of the reactive forces from Assignment 1. We can thus conclude that the Jacobian matrix $C_{k,i}$ represents the applied forces on the center of mass of the bodies for which the system is in equilibrium, i.e. the static equilibrium holds.

$$F_i = C_{k,i} \lambda_k$$

Note that we get opposite signs due to sign convention in describing the virtual power expression.

e)

Now our system has changed where there is a constraint at the right end of bar 2, point C, see figure below.



Now a new constraint has been introduced into the system, where only vertical sliding is allowed at point C. This constraint can be expressed in terms of CoM of body 2, i.e.

$$C_5 = x_2 + 1/2 * l_2 * cos(\varphi_2) = 0$$

This means $C_k, C_{k,j}$ and $C_{k,ij}\dot{x}_i\dot{x}_j$ become

$$C_k = \begin{pmatrix} x_1 - l_1/2 * cos(\varphi_1) \\ y_1 - 1_1/2 * sin(\varphi_1) \\ (x_2 - l_2/2 * cos(\varphi_2)) - (x_1 + l_1/2 * cos(\varphi_1)) \\ (y_2 - l_2/2 * sin(\varphi_2)) - (y_1 + l_1/2 * sin(\varphi_1)) \\ x_2 + 1/2 * l_2 * cos(\varphi_2) \end{pmatrix} = \mathbf{0}$$

$$C_{k,j} = \begin{pmatrix} 1 & 0 & 1/2*l_1*sin(\varphi_1) & 0 & 0 & 0 \\ 0 & 1 & -1/2*l_1*cos(\varphi_1) & 0 & 0 & 0 \\ -1 & 0 & 1/2*l_1*sin(\varphi_1) & 1 & 0 & 1/2*l_2*sin(\varphi_2) \\ 0 & -1 & -1/2*l_1*cos(\varphi_1) & 0 & 1 & -1/2*l_2*cos(\varphi_2) \\ 0 & 0 & 0 & 1 & 0 & -1/2*l_2*sin(\varphi_2) \end{pmatrix}$$

$$C_{k,ij}\dot{x}_i\dot{x}_j = \begin{pmatrix} 1/2*l_1*\dot{\varphi_1}^2cos(\varphi_1) \\ 1/2*l_1*\dot{\varphi_1}^2sin(\varphi_1) \\ 1/2*l_1*\dot{\varphi_1}^2cos(\varphi_1)+1/2*l_2*\dot{\varphi_2}^2cos(\varphi_2) \\ 1/2*l_1*\dot{\varphi_1}^2sin(\varphi_1)+1/2*l_2*\dot{\varphi_2}^2sin(\varphi_2) \\ -1/2*l_2*\dot{\varphi_2}^2*cos(\varphi_2) \end{pmatrix}$$

Now for e)-g) we solve

$$\begin{pmatrix} \ddot{x}_j \\ \lambda_k \end{pmatrix} = \begin{pmatrix} M_{ij} & C_{k,i} \\ C_{k,j} & 0_{kk} \end{pmatrix}^{-1} \begin{pmatrix} F_i \\ -C_{k,ij}\dot{x}_i\dot{x}_j \end{pmatrix}$$

with the help of Matlab. We put our initial values as $\varphi_1 = \varphi_2 = \frac{\pi}{2}$ and $\dot{\varphi}_1 = \dot{\varphi}_2 = 0rad/s$. Matlab gives us:

$$\ddot{x}_j = \begin{pmatrix} 7.3575m/s^2 \\ 0m/s^2 \\ -24.5250rad/s^2 \\ 7.3575m/s^2 \\ 0m/s^2 \\ 24.5250rad/s^2 \end{pmatrix}$$

and

$$\lambda_k = \begin{pmatrix} 0.1479N \\ 0N \\ 0N \\ 0N \\ 0.1479N \end{pmatrix}$$

Now looking at the Lagrange multipliers we notice that an equal amount of reaction force is applied in the x direction at point A and point C, this is due to the gravitational force working on identical beams, who are constricted at point A and C, i.e. at this position the system is symmetric around point B. This also results in the same acceleration happening for both bar 1 and bar 2 in the x-direction. Furthermore, we see no reaction force on the joint in point B, meaning that joint is free. Additionally, there are zero vertical forces, due to gravitational force being horizontal. And there is no acceleration in the y direction. Lastly, the acceleration is the same for both bars in the x-direction.

f)

We put our initial values as $\varphi_1 = \varphi_2 = pi/2$ and $\dot{\varphi}_1 = pi$ and $\dot{\varphi}_2 = 0rad/s$. Matlab gives us:

$$\ddot{x}_j = \begin{pmatrix} 7.3575m/s^2 \\ 2.9609m/s^2 \\ -24.5250rad/s^2 \\ 7.3575m/s^2 \\ 8.8826m/s^2 \\ 24.5250rad/s^2 \end{pmatrix}$$

and

$$\lambda_k = \begin{pmatrix} 0.1479N \\ 0.5356N \\ 0N \\ 0.3571N \\ 0.1479N \end{pmatrix}$$

We have the same accelerations as in **e)** except now we have centrifugal accelerations as well in the y direction, for both bar 1 and bar 2. We also notice that the acceleration in the y-direction is 3 times larger for bar 2 than bar 1, because centrifugal force is F_c. Again we notice that the system is symmetric through point B regarding the x-direction, i.e. same acceleration, same forces etc. The joint in point B is again unobstructed and can move freely in the x-direction.

g)

Here we get no solution because there are infinite solutions, i.e. the $\ddot{x}_j$ and λ_k are singular. Furthermore, when looking at the DAEs matrix we see that it is rank deficit, since it's rank is 10 and not 11, since it is a $11x11$ matrix. This means all 11 unknowns can not be determined, we are essentially missing an equation to be able to solve this. This can be further demonstrated if we take the nullspace of the DAEs matrix, which shows that we have a lot of variables that can have infinite solutions, i.e. Matlab gives that all variables in $\ddot{x}_j$ can have infinite solutions as well as the vertical forces in λ_k.

Appendix A

Matlab code

```
%% Find derivatives of constraints a)
% Set up variables
syms x1 y1 phi1 x2 y2 phi2
syms xd1 yd1 phid1 xd2 yd2 phid2 % for dx/dt I use xd etc
syms l1 l2

% put the cm coordinates in a column vector x
x = [x1; y1; phi1; x2; y2; phi2]
% and the time derivaties
xd = [xd1; yd1; phid1; xd2; yd2; phid2]

% the constraints
dxA = x1-l1/2*cos(phi1);
dyA = y1-l1/2*sin(phi1);
dxB = (x2-l2/2*cos(phi2))-(x1+l1/2*cos(phi1));
dyB = (y2-l2/2*sin(phi2))-(y1+l1/2*sin(phi1));
dC = x2 + 1/2*l2*cos(phi2);  %add this for the point C constraint
% put all constraints in one vector C
C = [dxA; dyA; dxB; dyB; dC];

% Cx is the jacobian dC/dx
Cx = jacobian(C,x);
Cx = simplify(Cx)

% and C2 are the convective terms C,xx*xd*xd
% which are by definition d(dC/dt)/dx*xd
% first determine dC/dt
Cd = Cx*xd; % this is dC/dt=dC/dx*xd
% and next the convective terms d(dC/dt)/dx*xd
C2 = jacobian(Cd,x)*xd;
C2 = simplify(C2)

matlabFunction(Cx,'File','Cxfunction')
matlabFunction(C2,'File','C2function')
```

```
%% Initialize variables
rho=670;
l1=0.6;
l2=l1;
V=l1*0.05*0.003;
m1=rho*V;
m2=m1;
I1=1/12*m1*l1^2;
I2=1/12*m2*l2^2;
g=9.81;
%% Initial position and velocities
phi1=0;  % b) pi/2, c) 0, d) 0, e) pi/2, f) pi/2, g) 0
phi2=0;  % b) pi/2, c) 0, d) 0, e) pi/2, f)pi/2, g) 0
phid1=0;    % b) 0, c) 0, d) pi, e) 0, f) -pi, g) 0
```

```
phid2=0;    % b) 0, c) 0, d) pi, e) 0, f) 0, g) 0
%% Set up DAEs  b), c) and d)
m=[m1 m1 I1 m2 m2 I2];
M=diag(m);
C_kj=[1 0 (l1*sin(phi1))/2 0 0 0;
      0 1 -(l1*cos(phi1))/2 0 0 0;
      -1 0 (l1*sin(phi1))/2 1 0 (l2*sin(phi2))/2;
      0 -1 -(l1*cos(phi1))/2 0 1 -(l2*cos(phi2))/2];
C_ki=C_kj';
C_kij=[(l1*phid1^2*cos(phi1))/2;
       (l1*phid1^2*sin(phi1))/2;
        (l1*cos(phi1)*phid1^2)/2 + (l2*cos(phi2)*phid2^2)/2;
        (l1*sin(phi1)*phid1^2)/2 + (l2*sin(phi2)*phid2^2)/2];
F_i=[m1*g 0 0 m2*g 0 0]';
Mat=[M C_ki; C_kj zeros(4)];
Y=[F_i; -C_kij];
Sol=inv(Mat)*Y
xddot=[Sol(1:6)]
lambda=[Sol(7:10)]

%% Set up DAEs e), f) and g)
m=[m1 m1 I1 m2 m2 I2];
M=diag(m);

C_kj=[1 0 (l1*sin(phi1))/2 0 0 0;
      0 1 -(l1*cos(phi1))/2 0 0 0;
      -1 0 (l1*sin(phi1))/2 1 0 (l2*sin(phi2))/2;
      0 -1 -(l1*cos(phi1))/2 0 1 -(l2*cos(phi2))/2;
      0 0 0 1 0 -(l2*sin(phi2))/2];

C_ki=C_kj';

C_kij=[(l1*phid1^2*cos(phi1))/2;
       (l1*phid1^2*sin(phi1))/2;
        (l1*cos(phi1)*phid1^2)/2 + (l2*cos(phi2)*phid2^2)/2;
        (l1*sin(phi1)*phid1^2)/2 + (l2*sin(phi2)*phid2^2)/2;
        -(l2*phid2^2*cos(phi2))/2];

F_i=[m1*g 0 0 m2*g 0 0]';    % For g) we put F_i=[m1*g 10 10*l1/2 m2*g 0 0]'
Mat=[M C_ki; C_kj zeros(5)];
Y=[F_i; -C_kij];
Sol=inv(Mat)*Y
xddot=[Sol(1:6)]
lambda=[Sol(7:11)]
```