Assignment 4

ELEC 442 - Introduction to Robotics

Sondre Myrberg (81113433)   Ola Helbaek (68776772)

November 30, 2017

THE UNIVERSITY OF BRITISH COLUMBIA

I. Two-Link Manipulator Open Loop Simulation

Considering the two link manipulator described on page 87 in chapter 6 of the notes, the equations of motion are given by equation (209) to (233). In general we have the Euler-Lagrange equations of motion given as

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\frac{\partial L}{\partial \dot{q}_i}(q,\dot{q})\right] - \frac{\partial L}{\partial q_i}(q,\dot{q}) = \tau_i$$

where

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

and τ_i is the generalized force or torque associated with coordinate i. This can be written on standard form as

$$D(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + G(\boldsymbol{q}) = \boldsymbol{u} + \underline{J}_n^\top \begin{bmatrix} \underline{f}_e \\ \underline{\tau}_e \end{bmatrix}$$

where the last term can be ignored as we do not interact with the environment. This gives us the secod order system

$$\ddot{\boldsymbol{q}} = D^{-1}(\boldsymbol{q})\left[\boldsymbol{u} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - G(\boldsymbol{q})\right]$$

with $D(\boldsymbol{q})$ given by equation (222), $C(\boldsymbol{q},\dot{\boldsymbol{q}})$ given by (232) and $G(\boldsymbol{q})$ by (233), and $\boldsymbol{q} = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix}$. To generate these matrices in Simulink, we implement block functions shown in Figure 8, Figure 9 and Figure 10 in Appendix B. The complete mainpulator dynamics are implemented in Figure 11.

i).

With $\boldsymbol{x}(0) = \begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}^\top$ and $\tau_1 = \tau_2 = 0$ we get the response shown in Figure 1.

ii).

With $\boldsymbol{x}(0) = \begin{bmatrix}0 & \frac{\pi}{2} & 0 & 0\end{bmatrix}^\top$ and $\tau_2 = 5\,\mathrm{Nm}$ we get the response shown in Figure 2. Here the total energy of the system is increasing, which is seen in the plots for the angular velocities.

iii).

With $\boldsymbol{x}(0) = \begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}^\top$ and friction modeled as $\tau_1 = -0.5\dot{\theta}_1$ and $\tau_2 = -0.5\dot{\theta}_2$ we get the response shown in Figure 3. Here we see that the amplitude is decreasing, wich makes sense because the total energy of the system is decreasing when friction is added.

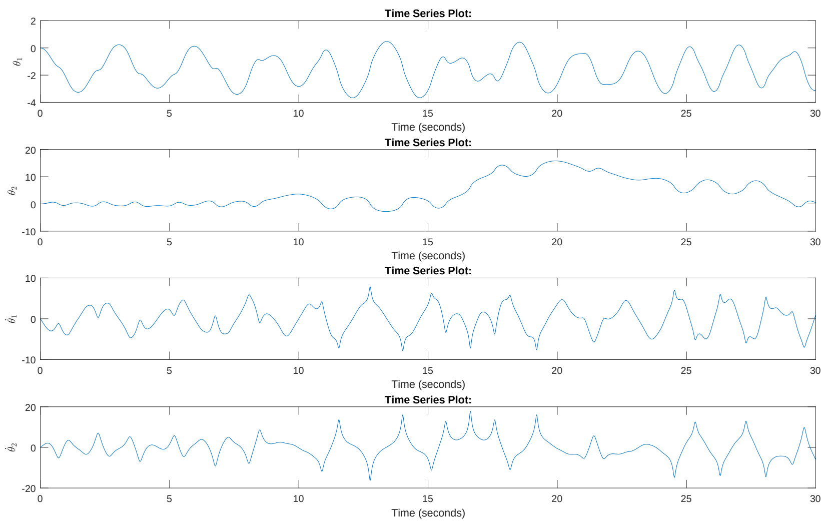

Figure 1: Response of all states with all states initialized to zero

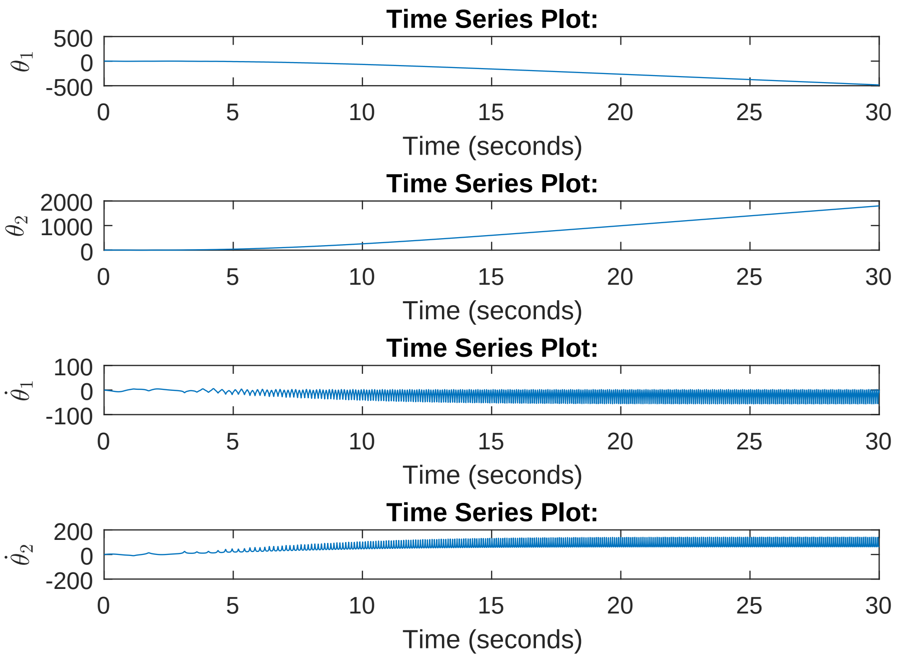

Figure 2: Response of all states with $\boldsymbol{x}(0) = \left[0\ \frac{\pi}{2}\ 0\ 0\right]^\top$ and $\tau_2 = 5\,\mathrm{Nm}$

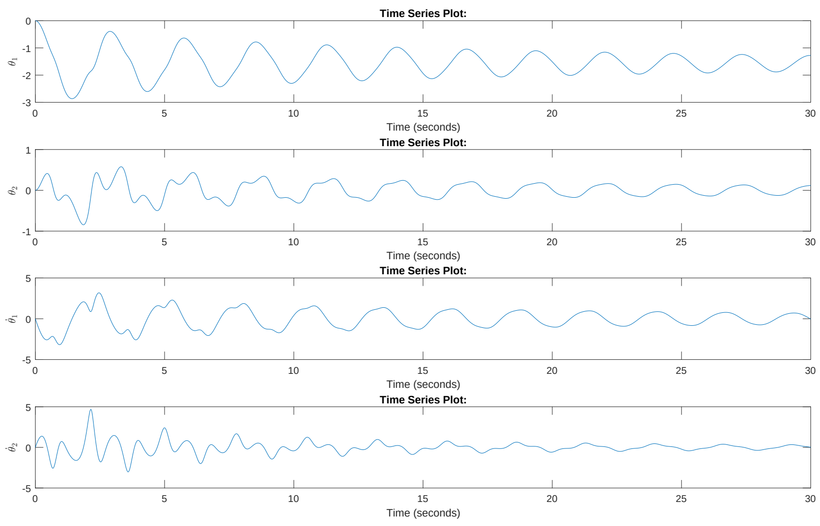

Figure 3: Response of all states with all states initialized to zero and friction modeled as $\tau_1 = -0.5\dot{\theta}_1$ and $\tau_2 = -0.5\dot{\theta}_2$

II. Controller Implementation

i). Closed loop joint-space control

With the PD + gravity set point controller we get the input vector

$$\boldsymbol{u} = \underbrace{G(\boldsymbol{q})}_{\text{Gravity terms}} + \underbrace{K_p(\boldsymbol{q}_d - \boldsymbol{q}) - K_v(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}})}_{\text{PD-controller}}$$

where we require

$$K_p, K_v \succ 0$$

which is implemented as the Simulink diagram shown in Figure 12. With this implementation and the initial conditions

$$\begin{aligned} \boldsymbol{x}(0) &= \begin{bmatrix} -\frac{\pi}{2} & 0 & 0 & 0 \end{bmatrix}^\top \\ \boldsymbol{q}_d &= \begin{bmatrix} 0 & \frac{\pi}{2} \end{bmatrix}^\top \\ \dot{\boldsymbol{q}}_d &= \begin{bmatrix} 0 & 0 \end{bmatrix}^\top \\ K_p &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ K_v &= \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \end{aligned}$$

we get the output shown in Figure 4.

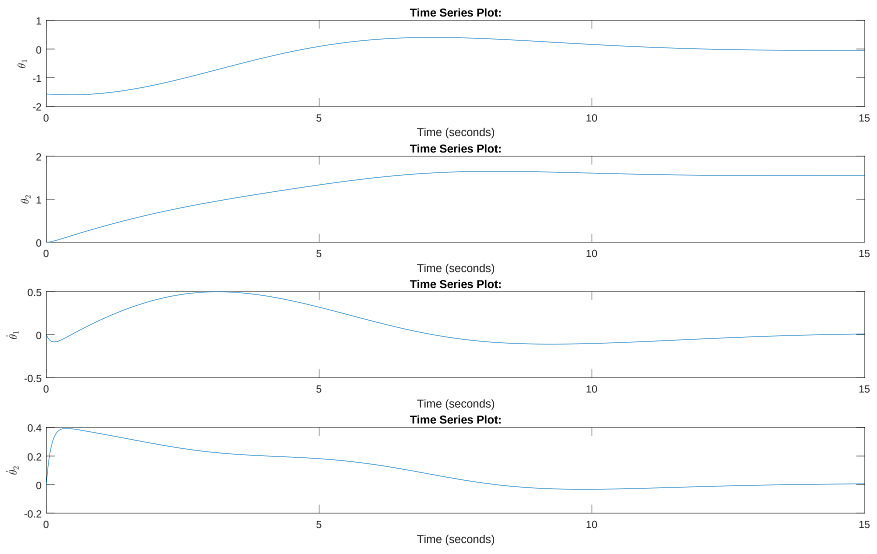

Figure 4: Output states with the PD + gravity controller

ii). Closed loop Cartesian-space control

When we consider the 2DOF double pendulum there are a lot fo terms we can ignore as they will cancel out anyway. The orientation error and angular velocities are not to be considered, and also the position along the $\boldsymbol{k}$-axis can be ignored. This results in the control law

$$\boldsymbol{u} = G(\boldsymbol{q}) + J^\top \begin{bmatrix} \underline{\boldsymbol{f}}_n \\ \underline{\boldsymbol{\tau}}_n \end{bmatrix}$$

$$\begin{bmatrix} \underline{\boldsymbol{f}}_n \\ \underline{\boldsymbol{\tau}}_n \end{bmatrix} = K_p(\boldsymbol{o}_d - \boldsymbol{o}_n) + K_v(\dot{\boldsymbol{o}}_d - \dot{\boldsymbol{o}}_n)$$

where J is given by the $\boldsymbol{i}$- and $\boldsymbol{j}$-components of

$$\begin{bmatrix} \boldsymbol{k}_0 \times (\boldsymbol{o}_n - \boldsymbol{o}_0) & \boldsymbol{k}_1 \times (\boldsymbol{o}_n - \boldsymbol{o}_1) \\ \boldsymbol{k}_0 & \boldsymbol{k}_1 \end{bmatrix}$$

and

$$\begin{aligned}
\boldsymbol{o}_0 &= \mathbf{0}^\top \\
\boldsymbol{o}_n &= \boldsymbol{o}_2 \\
\boldsymbol{o}_1 - \boldsymbol{o}_0 &= \begin{bmatrix} l_1 \cos\theta_1 \\ l_1 \sin\theta_1 \\ 0 \end{bmatrix} \\
\boldsymbol{o}_2 - \boldsymbol{o}_0 &= \begin{bmatrix} l_1 \cos\theta_1 + l_2\cos(\theta_1+\theta_2) \\ l_1 \sin\theta_1 + l_2\sin(\theta_1+\theta_2) \\ 0 \end{bmatrix} \\
\boldsymbol{o}_2 - \boldsymbol{o}_1 &= \begin{bmatrix} l_2\cos(\theta_1+\theta_2) \\ l_2\sin(\theta_1+\theta_2) \\ 0 \end{bmatrix}
\end{aligned}$$

This gives

$$\dot{\boldsymbol{o}}_n = \begin{bmatrix} -l_1 \sin\theta_1\dot\theta_1 - l_2\sin(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \\ l_1 \cos\theta_1\dot\theta_1 + l_2\cos(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \\ 0 \end{bmatrix}$$

and

$$J = \begin{bmatrix} -l_1 \sin\theta_1 - l_2\sin(\theta_1+\theta_2) & -l_2\sin(\theta_1+\theta_2) \\ l_1 \cos\theta_1 + l_2\cos(\theta_1+\theta_2) & l_2\cos(\theta_1+\theta_2) \\ 0 & 0 \\ \boldsymbol{k} & \boldsymbol{k} \end{bmatrix}$$

Again, we have a 2DOF manipulator, so we only have use the first two elements of our vectors, giving us

$$\begin{aligned}
\boldsymbol{u} = G(\boldsymbol{q}) + J^\top \Bigg[& K_p \left(\boldsymbol{o}_d - \begin{bmatrix} l_1 \cos\theta_1 + l_2\cos(\theta_1+\theta_2) \\ l_1 \sin\theta_1 + l_2\sin(\theta_1+\theta_2) \end{bmatrix} \right) \\
& + K_v \left(\dot{\boldsymbol{o}}_d - \begin{bmatrix} -l_1 \sin\theta_1\dot\theta_1 - l_2\sin(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \\ l_1 \cos\theta_1\dot\theta_1 + l_2\cos(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \end{bmatrix} \right) \Bigg] \\
J = & \begin{bmatrix} -l_1 \sin\theta_1 - l_2\sin(\theta_1+\theta_2) & -l_2\sin(\theta_1+\theta_2) \\ l_1 \cos\theta_1 + l_2\cos(\theta_1+\theta_2) & l_2\cos(\theta_1+\theta_2) \end{bmatrix}
\end{aligned}$$

The simulink diagram to generate the Jacobian is shown in Figure 13 and the simulink diagram for the stiffness controller is shown in Figure 14. To change between the different K_p's simply comment and uncomment in the MATLAB file as seen in Appendix A, as all constants are defined there. When changing controller, simply connect the deired controller in the simulink file shown in Figure 15 and run the associated script. With $K_p = \mathrm{diag}(1, 1)$ we get the response shown in Figure 5, with $K_p = \mathrm{diag}(0.2, 1)$ the response is shown in Figure 6 and with $K_p = \mathrm{diag}(1, 0.2)$ it's shown in Figure 7. From this we see that even though we have the same initial position and desired end effector location we get fundamentally different trajectories.

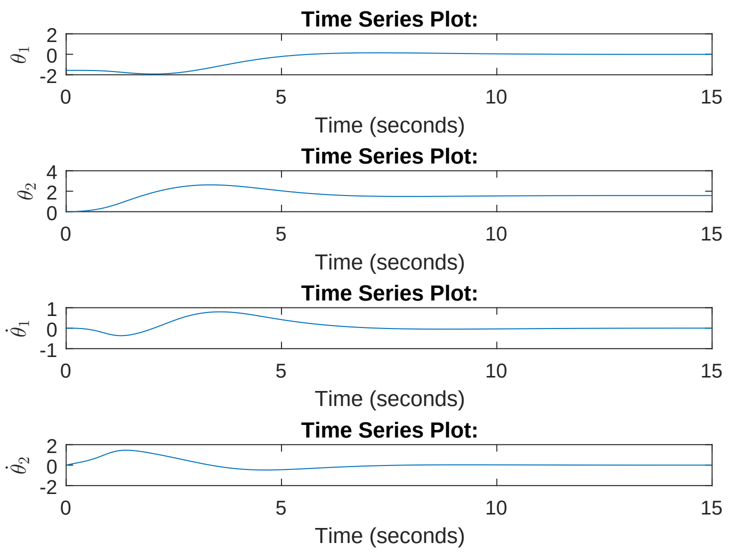

Figure 5: Response of the double pendulum with $K_p = \mathrm{diag}(1, 1)$

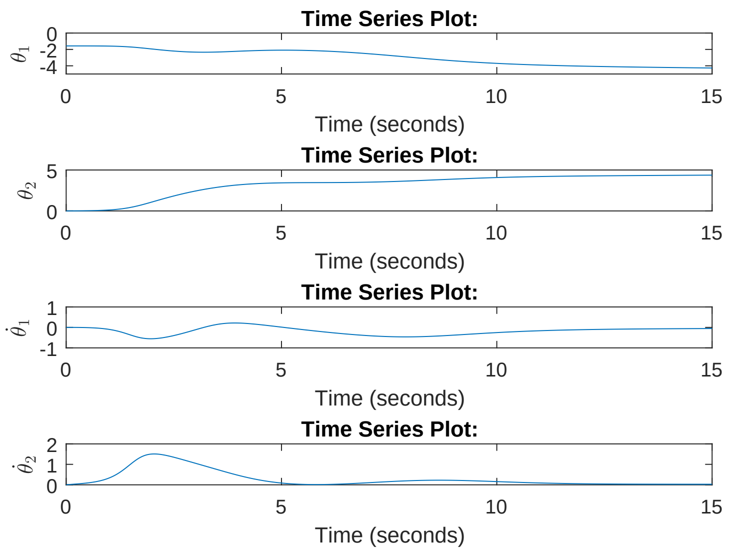

Figure 6: Response of the pwndulum with $K_p = \mathrm{diag}(0.2, 1)$

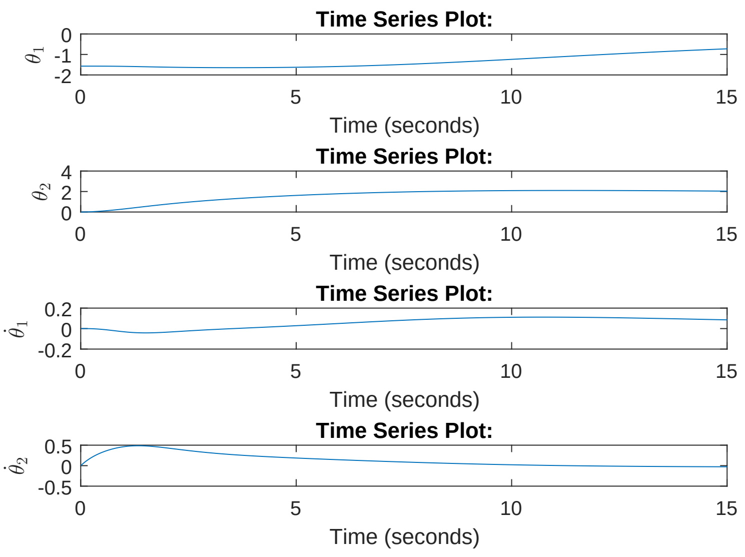

Figure 7: Response of the pwndulum with $K_p = \text{diag}(1, 0.2)$

Appendices

A. MATLAB code

```matlab
close all
clear variables

%% Variables
l1 = 1;
l2 = 1;
m1 = 1;
m2 = 1;
g = 9.81;
tau_1 = 0;
tau_2 = 0;

%% Joint space control
%Kp = diag([1 1]);
Kv = diag([2 2]);
x_init = [-pi/2;0;0;0];
q_d = [0; pi/2];
sim('assgt4');

%% Stiffness control
q_d = [0; pi/2]; %needed because of block structure but
   doesnt affect stiffness controller
% Kp = diag([1 1]);
% Kp = diag([0.2 1]);
Kp = diag([1 0.2]);
Kv = diag([2 2]);
x_init = [-pi/2;0;0;0];
o0 = [0 0]'; C0 = eye(2);
od = C0*[1 1]'; od_dot = [0 0]';
sim('assgt4');

%% Plot
figure
hold on;
subplot(4,1,1); plot(theta_1); ylabel('$\theta_1$', '
   interpreter', 'latex');
subplot(4,1,2); plot(theta_2); ylabel('$\theta_2$', '
   interpreter', 'latex');
```

```
subplot(4,1,3); plot(theta_dot_1); ylabel('$\dot{\theta}_1$'
    , 'interpreter', 'latex');
subplot(4,1,4); plot(theta_dot_2); ylabel('$\dot{\theta}_2$'
    , 'interpreter', 'latex');

figure
hold on;
plot(energy); plot(T, 'black'); plot(V, 'r');
```

Listing 1: MATLAB code to initialize variables and simulate. When the PD+Gravity controller is connected run that part, and the opposite when the stiffness controller is connected

B. Simulink diagrams

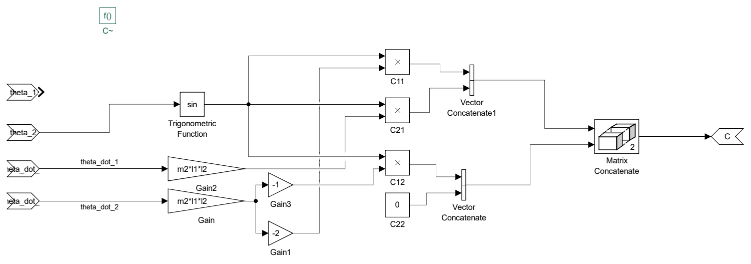

Figure 8: Simulink function block to generate $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$

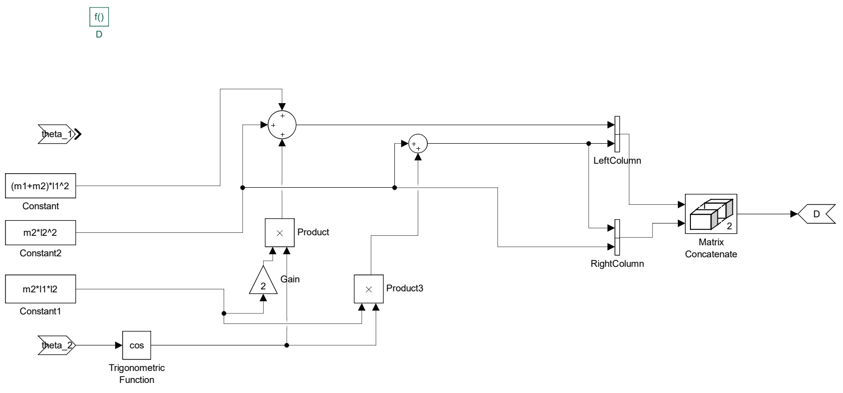

Figure 9: Simulink function block to generate $D(\boldsymbol{q})$

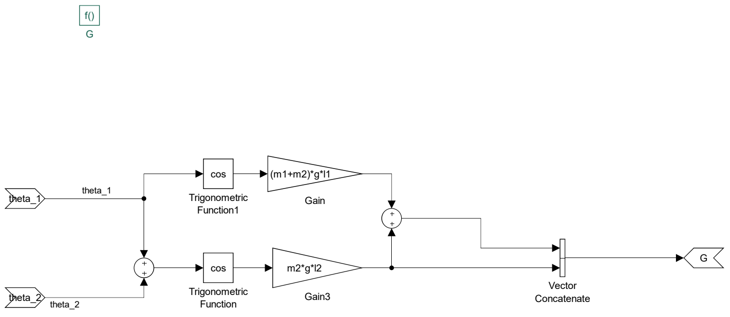

Figure 10: Simulink function block to generate $G(\boldsymbol{q})$

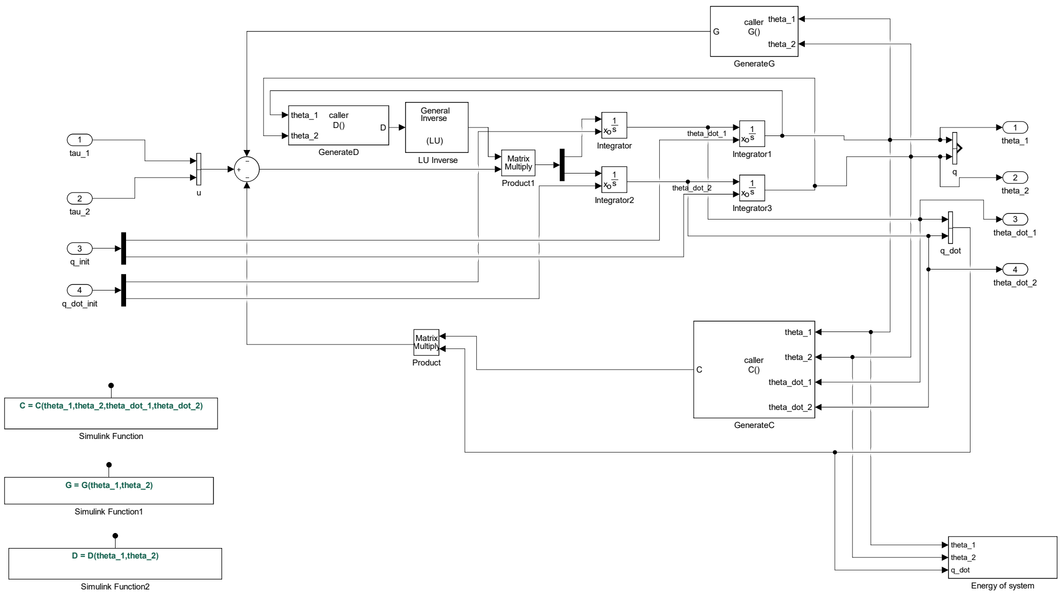

Figure 11: Simulink block to the complete manipulator dynamics

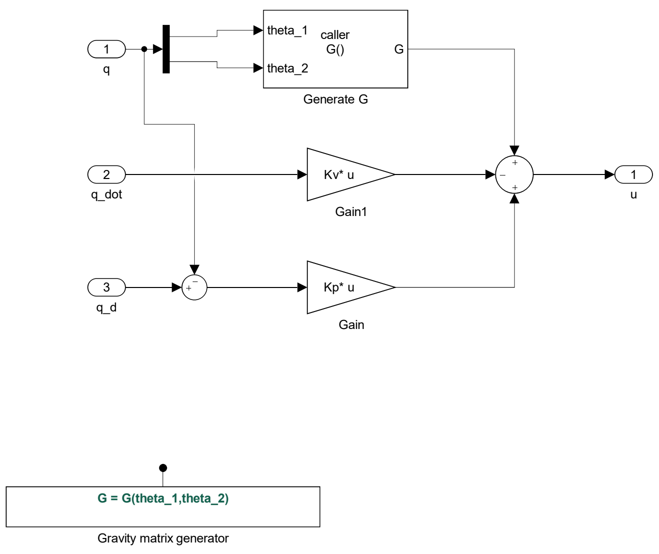

Figure 12: Simulink block for the PD + gravity controller

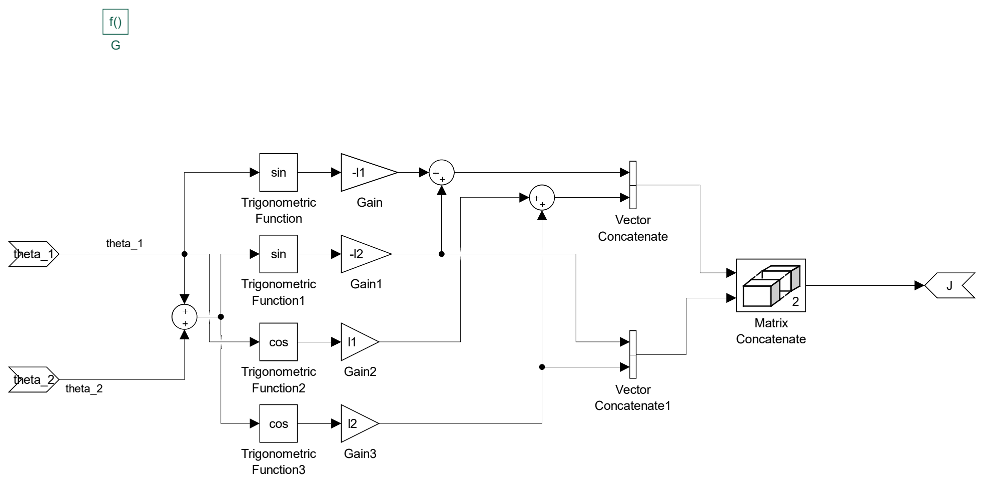

Figure 13: Simulink diagram for generating reduced Jacobian matrix

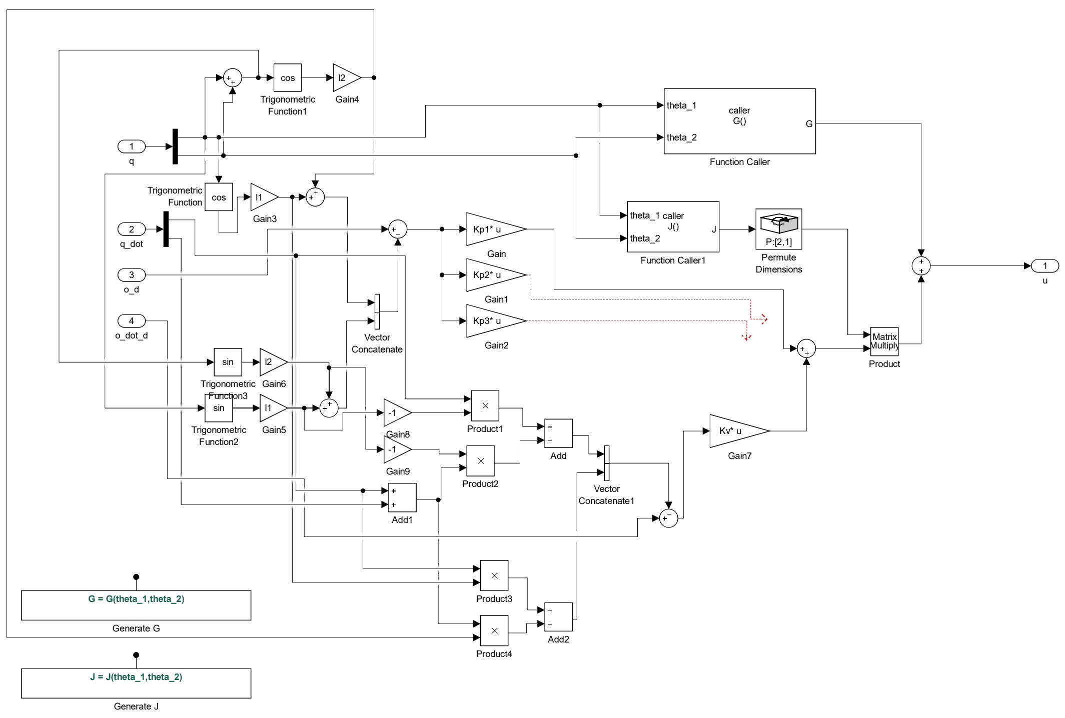

Figure 14: (Messy) simulink diagram of the stiffness controller. My apologies for the messiness here, but the output should be the same as the equations in section II subsection ii)

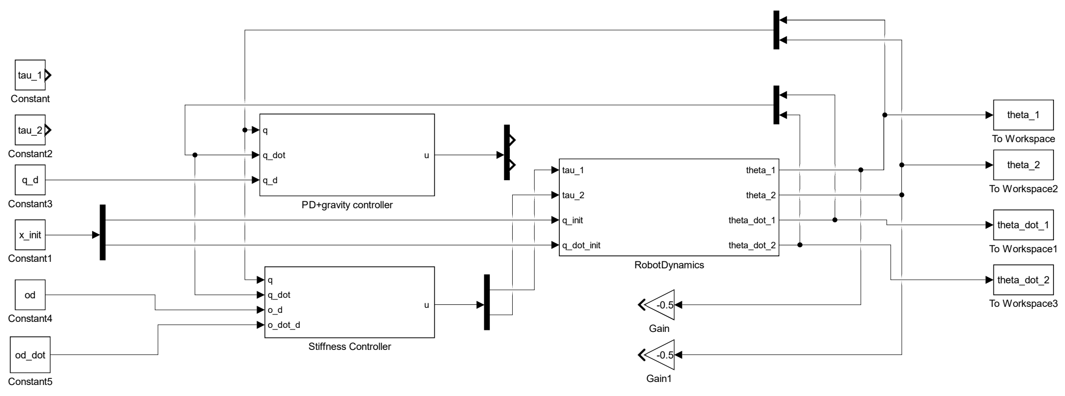

Figure 15: Overview of the system with corresponding blocks. The different controllers can easily be switched by changing wich has input into the Robot Dynamics block