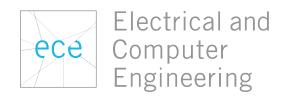
## ELEC 442 101

Introduction to Robotics

# Assignment 5

Zhi Chuen Tan Hubert Shim 65408361 ??????





### 1 Two-Link Manipulator Open-Loop Simulation

The Euler-Lagrange approach results in a generalized manipulator dynamics equation of the form:

$$\underbrace{D(q)}_{\substack{\text{manipulator} \\ \text{inertia matrix}}} \ddot{q} + \underbrace{C(q, \dot{q})\dot{q}}_{\substack{\text{accounts for} \\ \text{Coriolis & Centripetal} \\ \text{terms}}} + \underbrace{G(q)}_{\substack{\text{accounts for} \\ \text{gravitational & other} \\ \text{potential energy terms}}} = \underbrace{u}_{\substack{\text{generalized} \\ \text{motor} \\ \text{forces}}} + \underbrace{\underline{J_n^T} \left[ \underline{f_e} \right]}_{\underline{T_e}} \right]$$

As the robot does not interact with the environment, we can ignore the last term. Rearranging this equation then gives us:

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

where

$$q = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$u = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$$D(q) = \begin{bmatrix} (m_1 + m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2\cos\theta_2 & m_2l_2^2 + m_2l_1l_2\cos\theta_2 \\ m_2l_2^2 + m_2l_1l_2\cos\theta_2 & m_2l_2^2 \end{bmatrix}$$

$$C(q, \dot{q}) = \begin{bmatrix} -2m_2l_1l_2(\sin\theta_2)\dot{\theta_2} & -m_2l_1l_2(\sin\theta_2)\dot{\theta_2} \\ m_2l_1l_2(\sin\theta_2)\dot{\theta_1} & 0 \end{bmatrix}$$

$$G(q) = \begin{bmatrix} (m_1 + m_2)gl_1\cos\theta_1 + m_2gl_2\cos(\theta_1 + \theta_2) \\ m_2gl_2\cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$T = \frac{1}{2}\dot{q}^T D(q)\dot{q}$$

$$V = m_1gl_1\sin\theta_1 + m_2g[l_1\sin\theta_1 + l_2\sin(\theta_1 + \theta_2)]$$

$$\Rightarrow E_T = T + V$$

#### 1.1 Simulation

#### 1.1.1 MATLAB code

The following code was used to simplify running the simulation, plotting the joint angles and velocities, and kinetic, potential, and total energies, and automatically saves the figures as .fig (and .eps or .pdf files for the IATEX typesetting, which makes these diagrams vector diagrams, i.e. they can be zoomed-in without significant aliasing):

```
function run_sim_asn5q1(x_0_in,tau_in)
       % Initialize constants
       x_0 = x_0 = in;
       tau = tau_in;
       11=1;12=1;m1=1;m2=1;g=9.81;
       % Run simulation
       simOut = sim('asn5q1');
       % Take output values from Simulink
       theta1=simOut.get('theta1');
11
       theta2=simOut.get('theta2');
12
       theta1_dot=simOut.get('theta1_dot');
13
       theta2_dot=simOut.get('theta2_dot');
14
15
       T=simOut.get('T');
       V=simOut.get('V');
16
       E_total=simOut.get('E_total');
17
18
19
       % Plots theta1
       figure;
20
       hold on;
21
       view(2);
22
       title('Plot for $\theta_{1}$', 'Interpreter', 'latex');
23
24
       xlabel('Time (seconds)', 'Interpreter', 'latex');
       ylabel('$\theta_{1}$', 'Interpreter', 'latex');
25
       plot(theta1, 'Color', '#A2142F');
       saveas(gcf, 'qla_tl.fig'); % saves figure as .fig
27
       saveas(gcf, 'q1a_t1', 'epsc'); % saves figure as .eps (for preparing text)
28
29
       % Plots theta2
30
31
       figure;
       hold on;
32
       view(2);
33
       title('Plot for $\theta_{2}$', 'Interpreter', 'latex');
34
       xlabel('Time (seconds)', 'Interpreter', 'latex');
35
```

```
ylabel('$\theta_{2}$', 'Interpreter', 'latex');
       plot(theta2, 'Color', '#A2142F');
37
       saveas(gcf, 'qla_t2.fig'); % saves figure as .fig
38
39
       saveas(gcf, 'q1a_t2', 'epsc'); % saves figure as .eps (for preparing text)
40
       % Plots theta1_dot
       figure;
42
       hold on;
43
44
       view(2);
       title('Plot for $\dot{\theta}_{1}$', 'Interpreter', 'latex');
45
       xlabel('Time (seconds)', 'Interpreter', 'latex');
46
       ylabel('$\dot{\theta}_{1}$', 'Interpreter', 'latex');
47
       plot(theta1_dot, 'Color', '#A2142F');
48
       saveas(gcf, 'qla_tld.fig'); % saves figure as .fig
       saveas(gcf, 'qla_tld', 'epsc'); % saves figure as .eps (for preparing text)
50
51
       % Plots theta2_dot
       figure;
53
       hold on;
       view(2);
55
       title('Plot for \dot{\theta}_{2}, 'Interpreter', 'latex');
56
       xlabel('Time (seconds)', 'Interpreter', 'latex');
       ylabel('$\dot{\theta}_{2}$', 'Interpreter', 'latex');
58
       plot(theta2_dot, 'Color', '#A2142F');
59
       saveas(gcf, 'qla_t2d.fig'); % saves figure as .fig
       saveas(gcf, 'qla.t2d', 'epsc'); % saves figure as .eps (for preparing text)
61
62
       % Plots kinetic energy
63
       figure;
64
       hold on;
       view(2);
66
67
       title('Plot for Kinetic Energy, $T$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
68
       ylabel('$T$', 'Interpreter', 'latex');
69
       plot(T, 'Color', '#A2142F');
       saveas(gcf, 'qla_T.fig'); % saves figure as .fig
71
72
       saveas(gcf, 'q1a_T', 'epsc'); % saves figure as .eps (for preparing text)
       % Plots potential energy
74
       figure;
75
       hold on;
77
       view(2);
       title('Plot for Potential Energy, $T$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
79
       ylabel('$V$', 'Interpreter', 'latex');
80
```

```
plot(V, 'Color', '#A2142F');
       saveas(gcf, 'qla_V.fig'); % saves figure as .fig
82
       saveas(gcf, 'qla_V', 'epsc'); % saves figure as .eps (for preparing text)
83
84
       % Plots total energy
85
       figure;
       hold on;
87
       view(2);
88
       title('Plot for Total Energy, $T+V$', 'Interpreter', 'latex');
89
       xlabel('Time (seconds)', 'Interpreter', 'latex');
90
       ylabel('$T+V$', 'Interpreter', 'latex');
91
       plot(E_total, 'Color', '#A2142F');
92
       saveas(gcf, 'qla_total.fig'); % saves figure as .fig
93
       saveas(gcf, 'qla_total', 'epsc'); % saves figure as .eps (for preparing text)
   end
95
```

Listing 1: MATLAB code used to simulate the system for questions 1(a-b)

```
function run_sim_asn5q1c(x_0_in)
       % Initialize constants
       x_0 = x_0 = in;
       11=1;12=1;m1=1;m2=1;q=9.81;
       % Run simulation
       simOut = sim('asn5q1c');
        % Take output values from Simulink
10
       theta1=simOut.get('theta1');
11
       theta2=simOut.get('theta2');
       theta1_dot=simOut.get('theta1_dot');
13
       theta2_dot=simOut.get('theta2_dot');
14
15
       T=simOut.get('T');
       V=simOut.get('V');
16
       E_total=simOut.get('E_total');
17
18
       % Plots thetal
19
       figure;
       hold on;
21
       view(2);
22
23
       title('Plot for $\theta_{1}$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
24
       ylabel('$\theta_{1}$', 'Interpreter', 'latex');
25
       plot(theta1, 'blue');
26
```

```
27
       saveas(gcf, 'qlc_tl.fig'); % saves figure as .fig
       saveas(gcf, 'qlc_tl', 'epsc'); % saves figure as .eps (for preparing text)
28
29
30
       % Plots theta2
       figure;
31
32
       hold on;
       view(2);
33
       title('Plot for $\theta_{2}$', 'Interpreter', 'latex');
34
       xlabel('Time (seconds)', 'Interpreter', 'latex');
35
       ylabel('$\theta_{2}$', 'Interpreter', 'latex');
36
       plot(theta2, 'blue');
37
       saveas(gcf, 'q1c_t2.fig'); % saves figure as .fig
38
       saveas(gcf, 'q1c_t2', 'epsc'); % saves figure as .eps (for preparing text)
39
       % Plots theta1_dot
41
       figure;
42
       hold on;
       view(2);
44
       title('Plot for $\dot{\theta}_{1}$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
46
       47
       plot(theta1_dot, 'blue');
       saveas(gcf, 'q1c_t1d.fig'); % saves figure as .fig
49
       saveas(gcf, 'q1c_t1d', 'epsc'); % saves figure as .eps (for preparing text)
50
       % Plots theta2_dot
52
53
       figure;
       hold on;
54
       view(2);
55
       title('Plot for $\dot{\theta}_{2}$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
57
58
       ylabel('\$\dot{\theta}_{2}\$', 'Interpreter', 'latex');
       plot(theta2_dot, 'blue');
59
       saveas(gcf, 'qlc_t2d.fig'); % saves figure as .fig
60
       saveas(gcf, 'q1c_t2d', 'epsc'); % saves figure as .eps (for preparing text)
62
       % Plots kinetic energy
63
       figure;
       hold on;
65
       view(2);
66
67
       title('Plot for Kinetic Energy, $T$', 'Interpreter', 'latex');
       xlabel('Time (seconds)', 'Interpreter', 'latex');
68
       ylabel('$T$', 'Interpreter', 'latex');
69
70
       plot(T, 'blue');
       saveas(gcf, 'q1c_T.fig'); % saves figure as .fig
71
```

```
saveas(gcf, 'q1c_T', 'epsc'); % saves figure as .eps (for preparing text)
73
        % Plots potential energy
74
75
        figure;
       hold on;
76
        view(2);
       title('Plot for Potential Energy, $T$', 'Interpreter', 'latex');
78
       xlabel('Time (seconds)', 'Interpreter', 'latex');
79
        ylabel('$V$', 'Interpreter', 'latex');
80
       plot(V, 'blue');
81
        saveas(gcf, 'q1c_V.fig'); % saves figure as .fig
82
        saveas(gcf, 'q1c_V', 'epsc'); % saves figure as .eps (for preparing text)
83
84
        % Plots total energy
        figure;
86
       hold on;
       view(2);
88
       title('Plot for Total Energy, $T+V$', 'Interpreter', 'latex');
89
        xlabel('Time (seconds)', 'Interpreter', 'latex');
91
       ylabel('$T+V$', 'Interpreter', 'latex');
        plot(E_total, 'blue');
92
        saveas(gcf, 'qlc_total.fig'); % saves figure as .fig
93
        saveas(qcf, 'qlc_total', 'epsc'); % saves figure as .eps (for preparing text)
94
95
96
97
98
99
  end
100
```

Listing 2: MATLAB code used to simulate the system for question 1(c)

#### 1.1.2 Simulation results

All of the diagrams below are vector diagrams and can be zoomed in without significant aliasing.

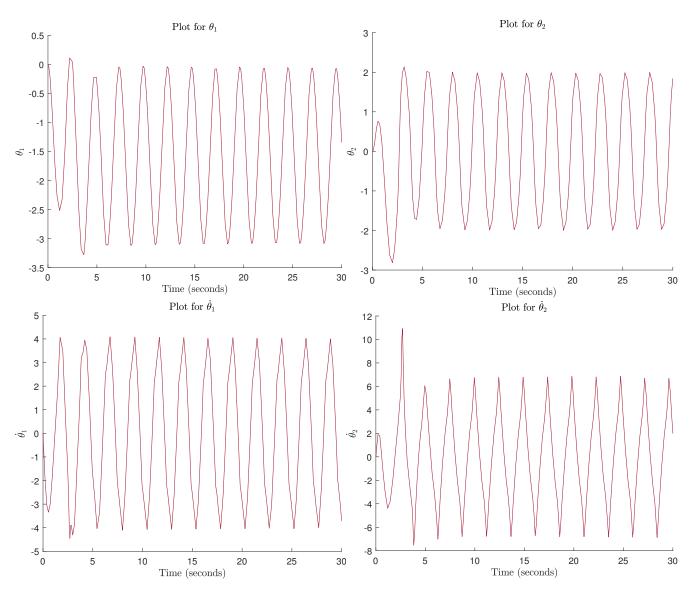


Figure 1: Joint angle and velocity plots for  $x(0)=\begin{bmatrix}0&0&0&0\end{bmatrix}^T$ ,  $\tau_1=\tau_2=0$ 

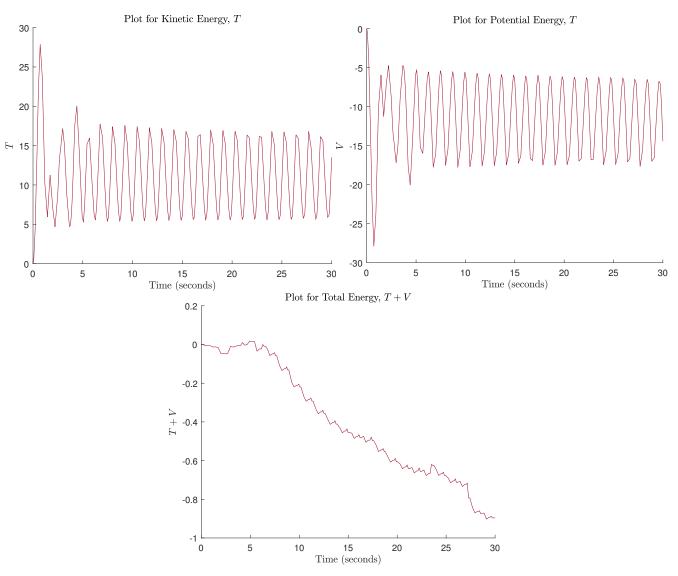


Figure 2: Energy plots for  $x(0)=\begin{bmatrix}0&0&0&0\end{bmatrix}^T$  ,  $\tau_1=\tau_2=0$ 

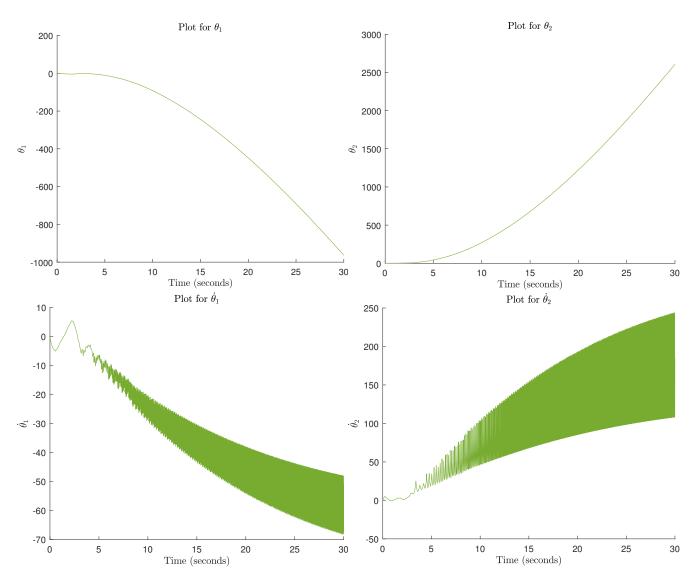


Figure 3: Joint angle and velocity plots for  $x(0) = \begin{bmatrix} 0 & \frac{\pi}{2} & 0 & 0 \end{bmatrix}^T$ ,  $\tau_1 = 0$ ,  $\tau_2 = 5$ 

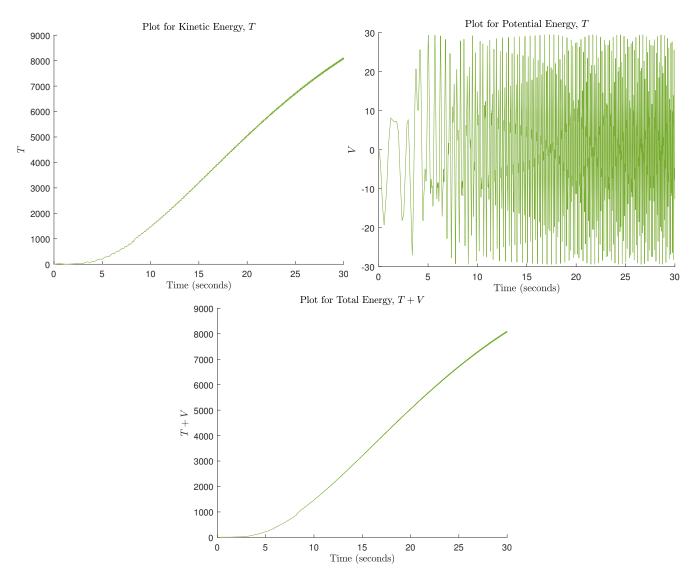


Figure 4: Energy plots for  $x(0)=\begin{bmatrix}0&\frac{\pi}{2}&0&0\end{bmatrix}^T$ ,  $\tau_1=0,\,\tau_2=5$ 

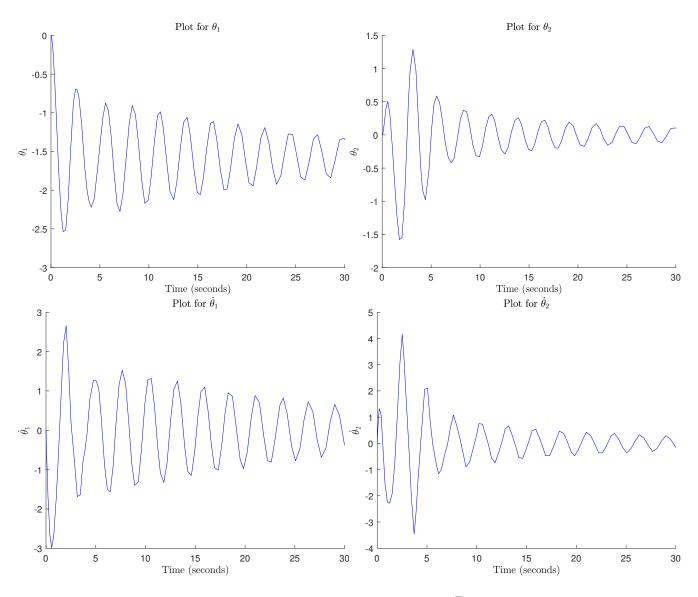


Figure 5: Joint angle and velocity plots for  $x(0) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ ,  $\tau_1 = -0.5\dot{\theta}_1$ ,  $\tau_2 = -0.5\dot{\theta}_2$ 

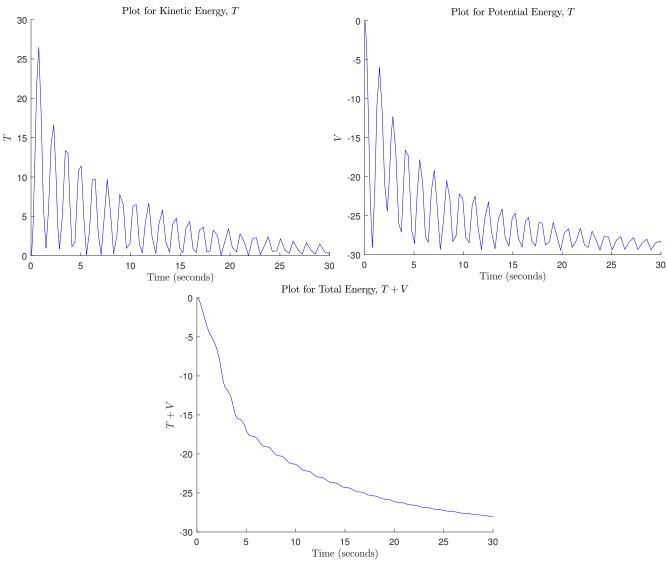


Figure 6: Energy plots for  $x(0) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ ,  $\tau_1 = -0.5\dot{\theta}_1$ ,  $\tau_2 = -0.5\dot{\theta}_2$ 

### 1.1.3 Simulink block diagrams

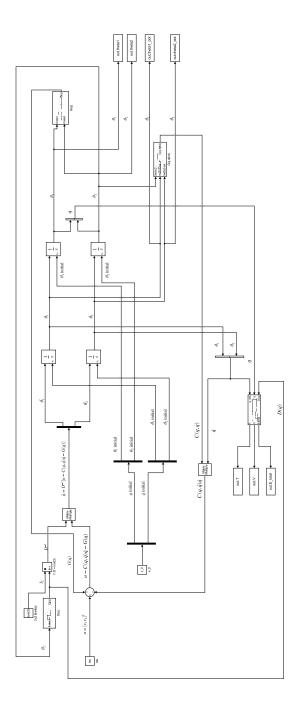


Figure 7: Simulink Block Diagram for the robot.

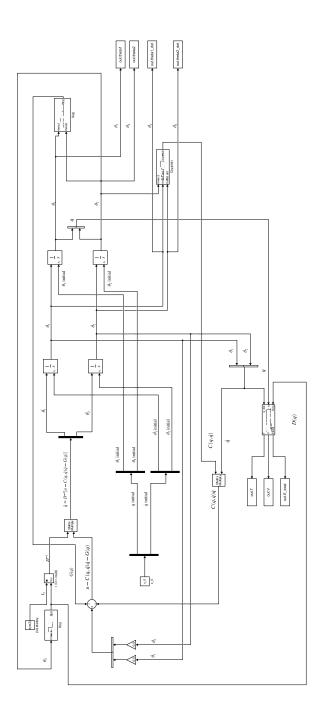


Figure 8: Simulink Block Diagram for the robot, used in part 1(c) to feed friction back into the system.)

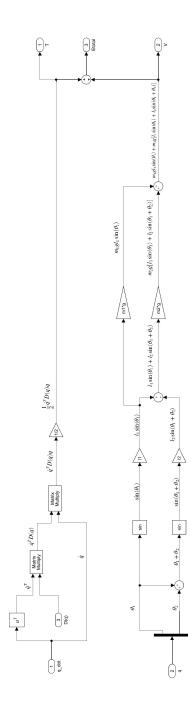


Figure 9: Simulink Block Diagram for the subsystem used to calculate (potetial, kinetic, and total) energy.

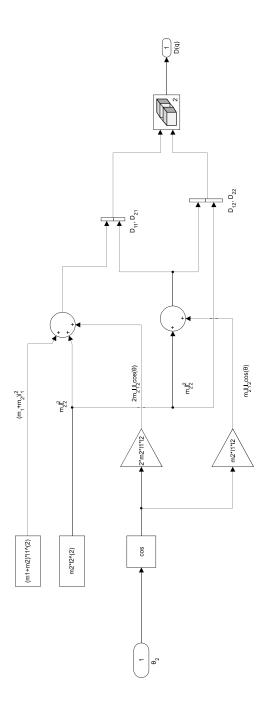


Figure 10: Simulink Block Diagram for the the subsystem used to calculate the D(q) matrix.

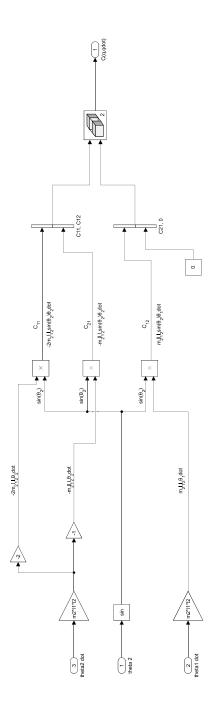


Figure 11: Simulink Block Diagram for the the subsystem used to calculate the  $C(\boldsymbol{q},\dot{\boldsymbol{q}})$  matrix.

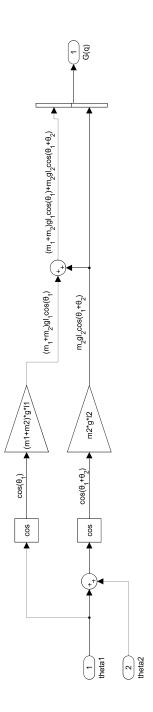


Figure 12: Simulink Block Diagram for the subsystem used to calculate the G(q) matrix.

## 2 Closed-Loop Controller Implementation

- 2.1 Joint Space Control
- 2.2 Task Space Control