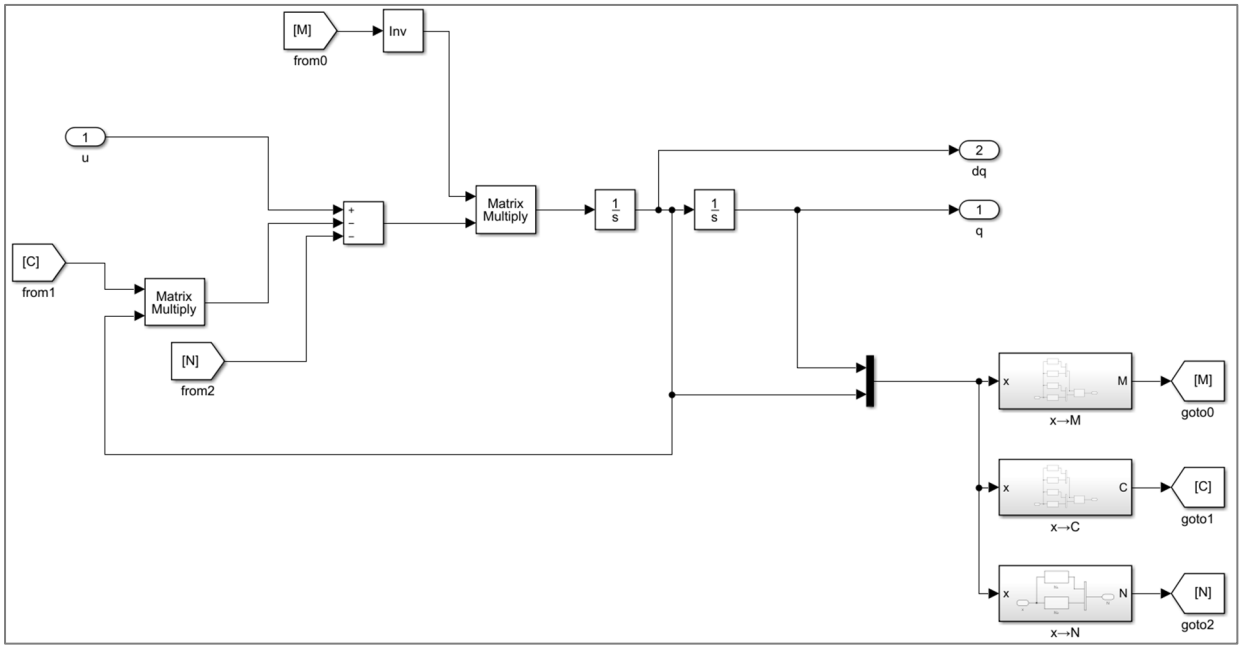
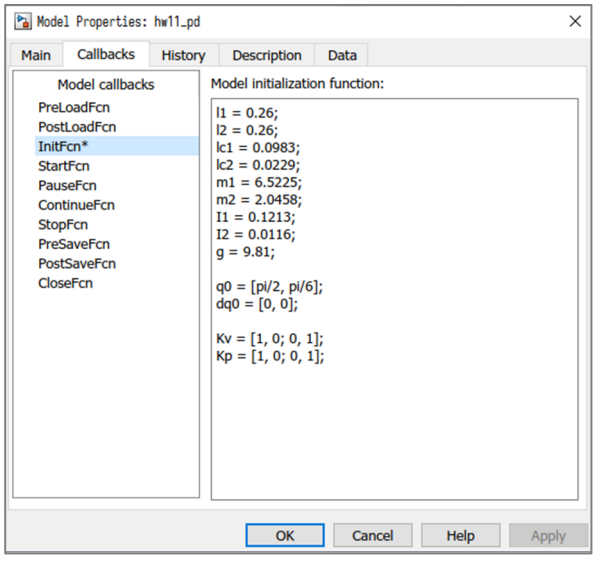
Robot Model

Refer to `robot_pd.slx` and `robot_ctc.slx` for the implementation in `Simulink`.

In both files, there is a Subsystem called `ROBOT` that contains the system dynamics for a robot manipulator model. Inside the subsystem is the following.



The model parameters are loaded as follows, with the values listed in the image. First, the parameters for the robot. Second, the initial conditions for $q(t)$ and $\dot{q}(t)$. Third, the values for K_p and K_v, depending on the controller.

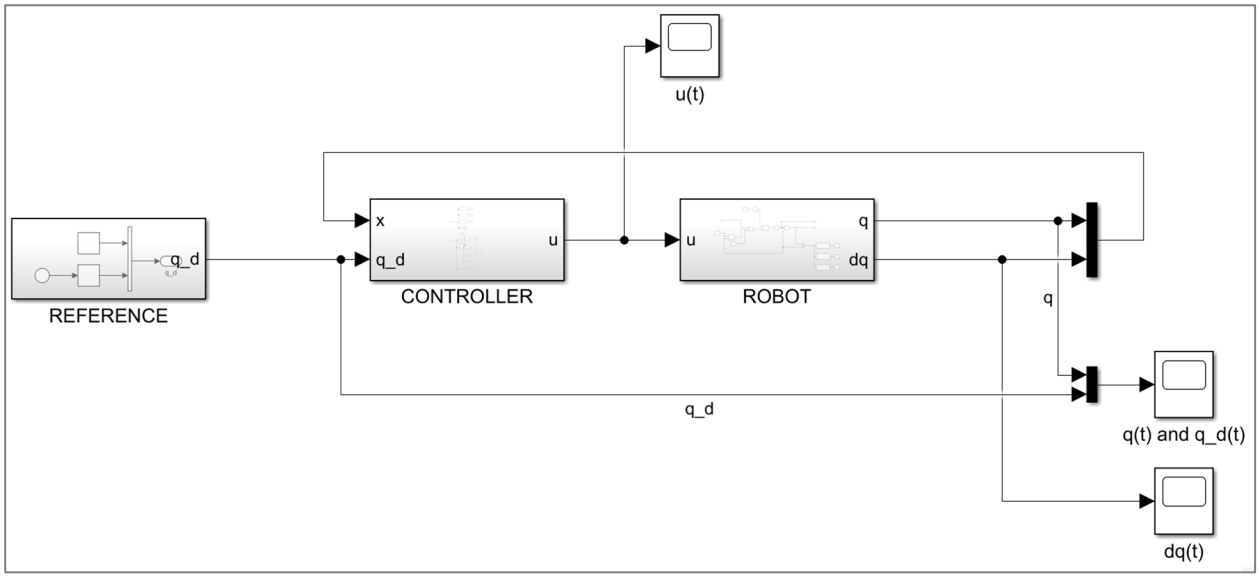


The initial conditions are properly set up in the `Integrator` blocks.

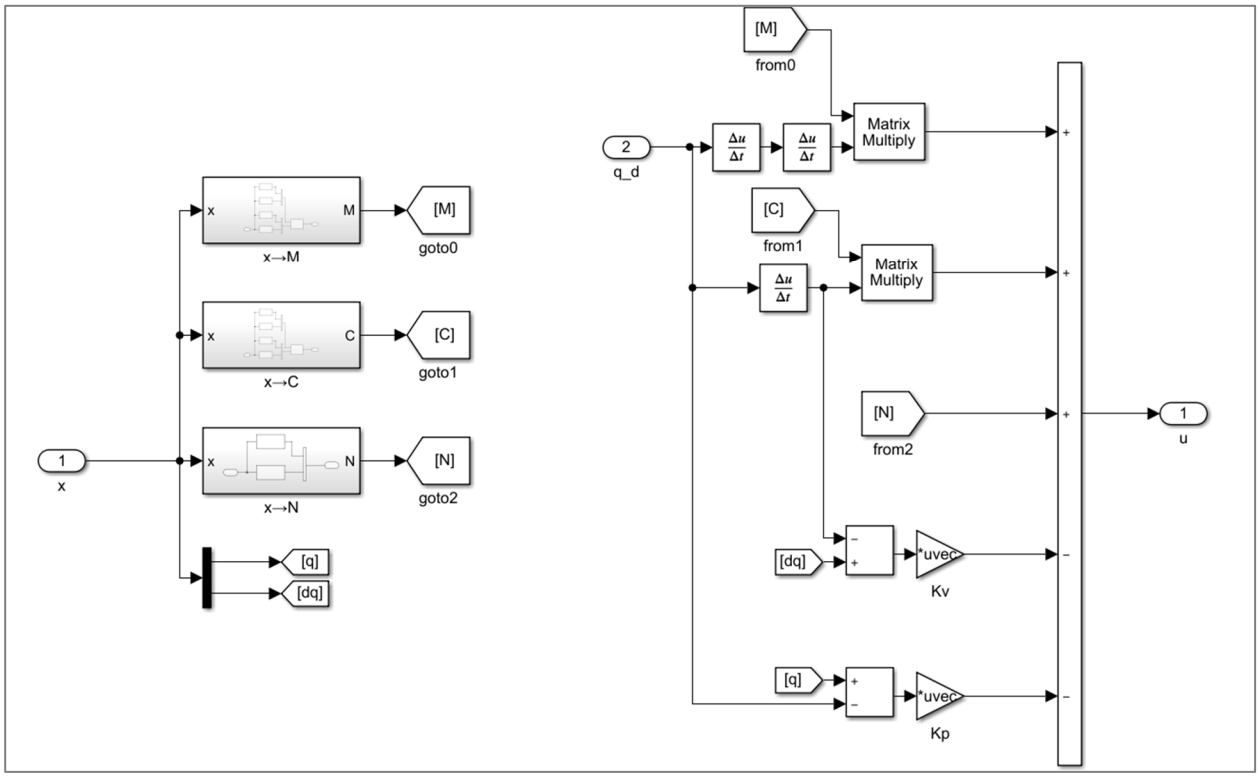
PD Control

Refer to *robot_pd.slx* for the implementation in *Simulink*.

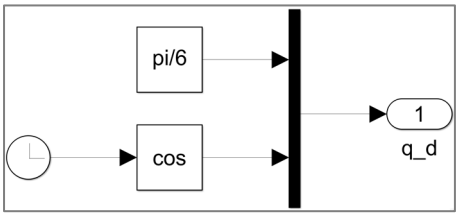
The overall control loop used in the simulation is as follows.



The Augmented PD Controller is programmed into the *CONTROLLER* block and implements the following control law: $\boxed{u(t) = M(q)\ddot{q}_d + C(q,\dot{q})\dot{q}_d + N(q,\dot{q}) - \mathrm{K_v}\dot{e} - \mathrm{K_p}e}$



The desired reference for the position is programmed into the `REFERENCE` block, and is set to $q_d(t) = \begin{bmatrix} \pi/6 \\ \cos(t) \end{bmatrix}$ using the following blocks.



The scopes capture three plots of the four desired quantities:

- position $q_d(t)$ versus $q(t)$
- velocity $\dot{q}(t)$
- input $u(t)$

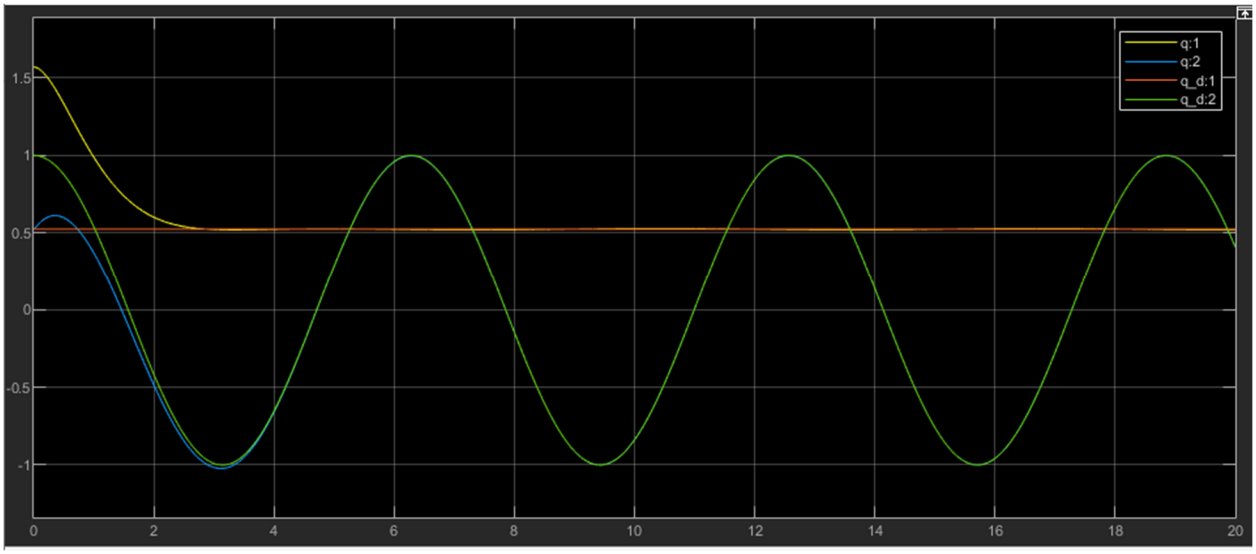
The plots are laid out in this order on the next page.

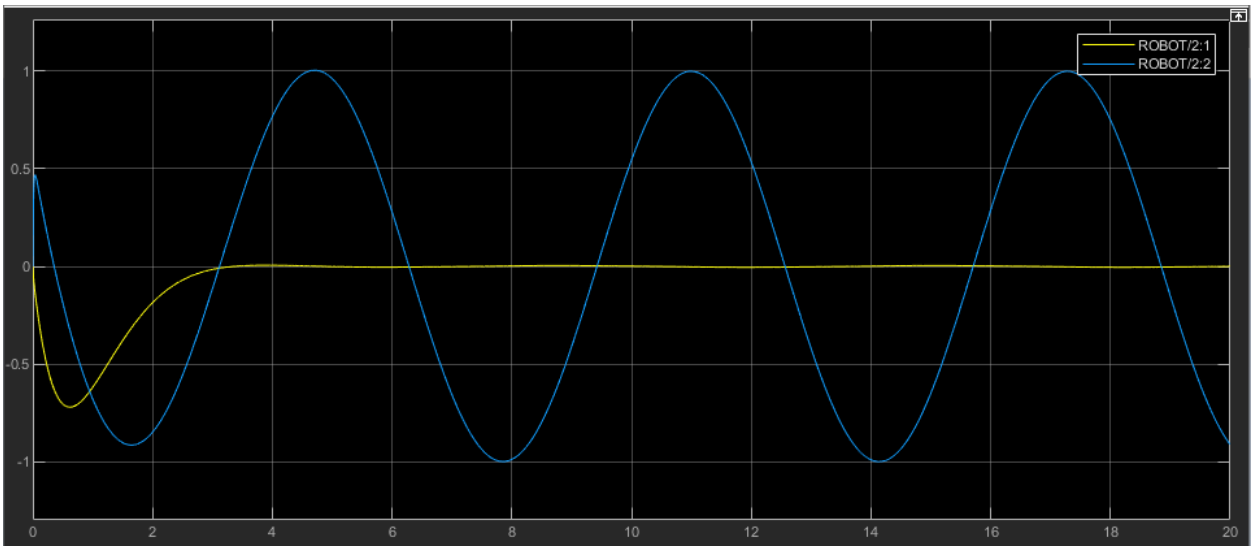
Remarks

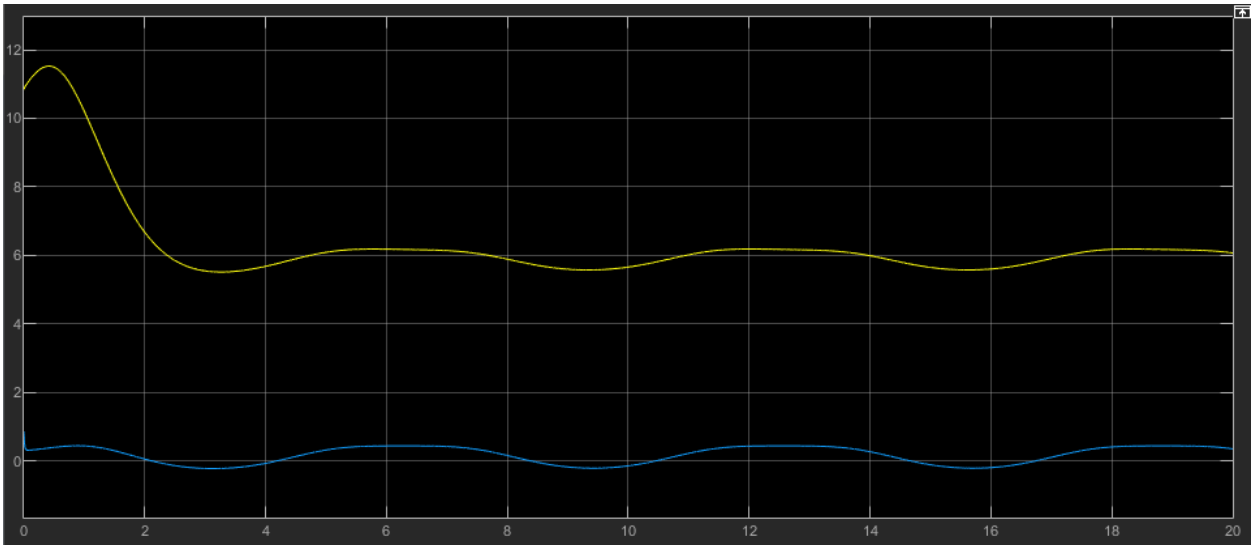
- The position $q(t)$ and velocity $\dot{q}(t)$ start from the correct initial conditions.
- $q_1(t)$ stabilizes and reaches its setpoint target of $q_{d,1}(t) = \pi/6$.
- $q_2(t)$ is correctly tracking its target of $q_{d,2}(t) = \cos(t)$.
- The input becomes periodic, which makes sense, because it is tracking a sinusoid.
- $\dot{q}_1(t)$ stabilizes and reaches zero, makes sense, because $q_1(t)$ becomes constant.
- $\dot{q}_2(t)$ becomes sinusoidal, makes sense, because $q_2(t)$ tracks a sinusoidal target.

For the Augmented PD Controller, the positive-definite gain matrices were set to:

```
Kv = [1, 0; 0, 1];
Kp = [1, 0; 0, 1];
```


q:1
q:2
q_d:1
q_d:2
1.5
1
0.5
0
-0.5
-1
0
2
4
6
8
10
12
14
16
18
20

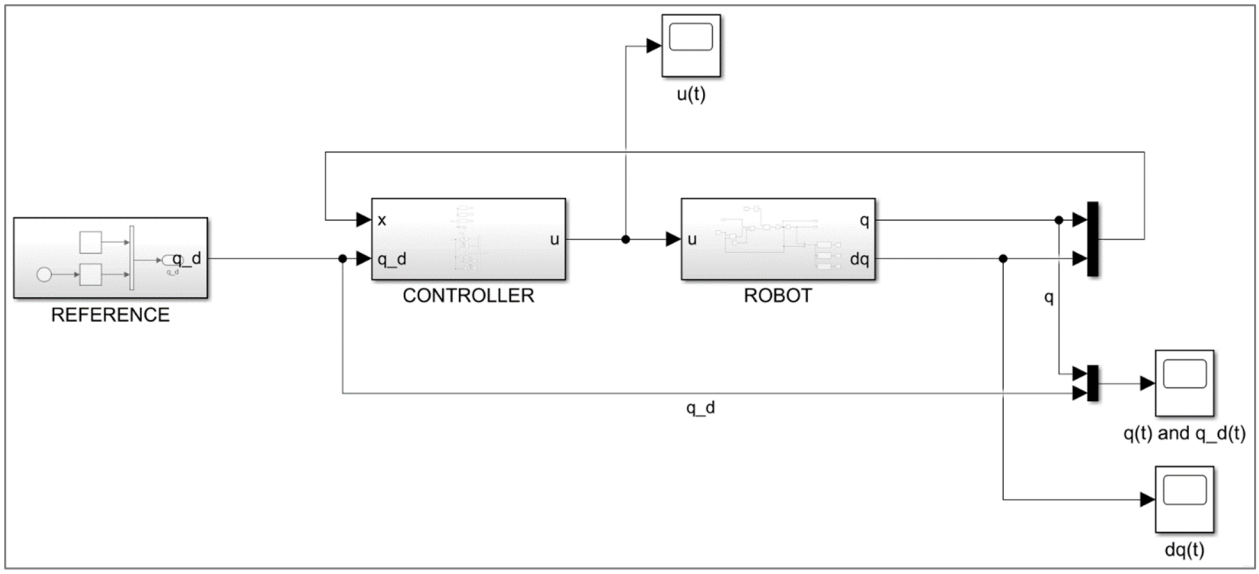

ROBOT/2:1
ROBOT/2:2
1
0.5
0
-0.5
-1
0
2
4
6
8
10
12
14
16
18
20


12
10
8
6
4
2
0
0
2
4
6
8
10
12
14
16
18
20

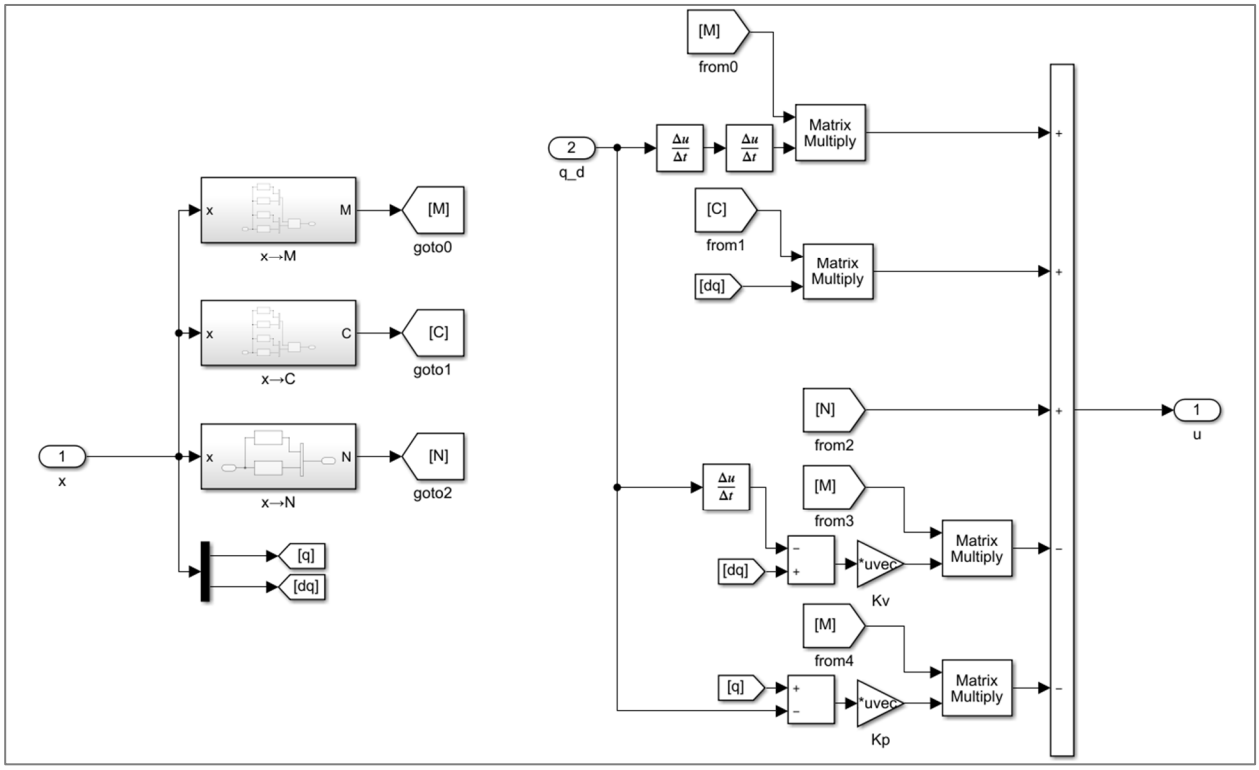
Computed–Torque Control

Refer to *robot_ctc.slx* for the implementation in *Simulink*.

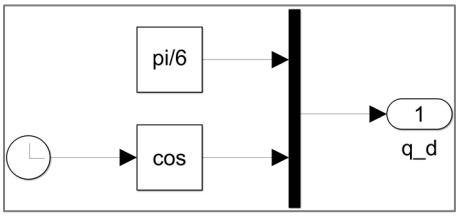
The overall control loop used in the simulation is as follows.



The Computed Torque Controller is programmed into the *CONTROLLER* block and implements the control law: $\boxed{u(t) = M(q)\ddot{q}_d + C(q,\dot{q})\dot{q} + N(q,\dot{q}) - M(q)(\mathrm{K_v}\dot{e} + \mathrm{K_p}e)}$



The desired reference for the position is programmed into the *REFERENCE* block, and is set to $q_d(t) = \begin{bmatrix} \pi/6 \\ \cos(t) \end{bmatrix}$ using the following blocks.



The scopes capture three plots of the four desired quantities:

- position $q_d(t)$ versus $q(t)$
- velocity $\dot{q}(t)$
- input $u(t)$

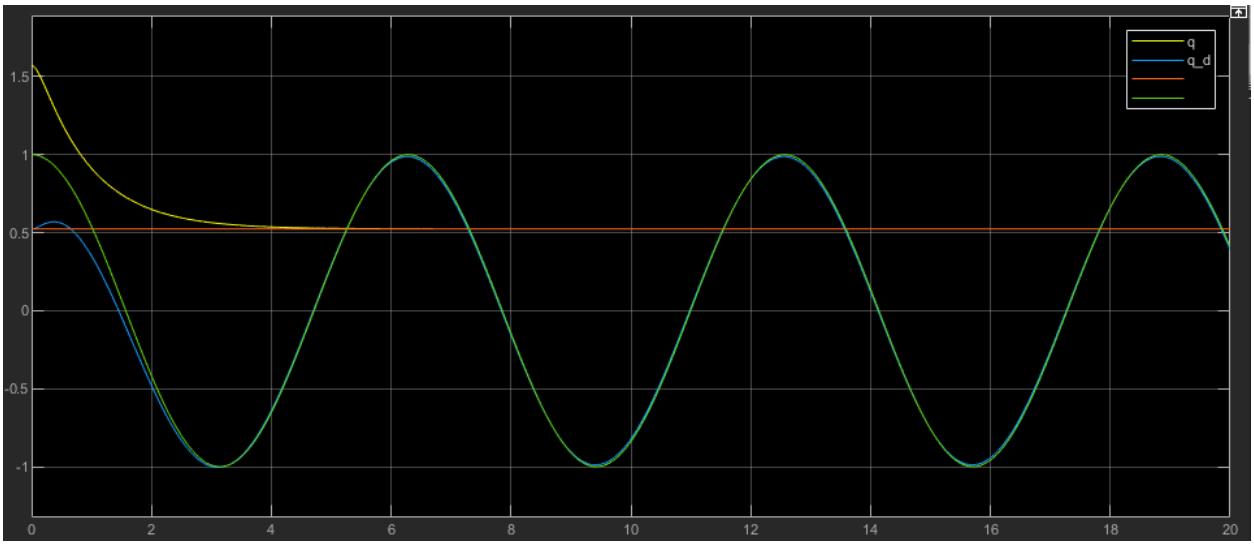
The plots are laid out in this order on the next page.

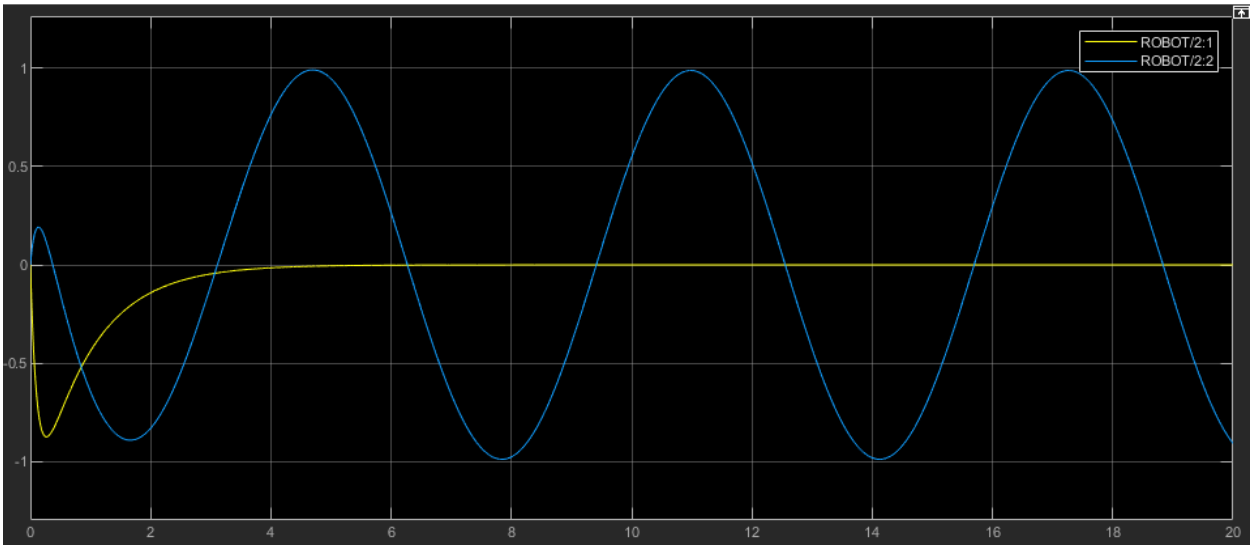
Remarks

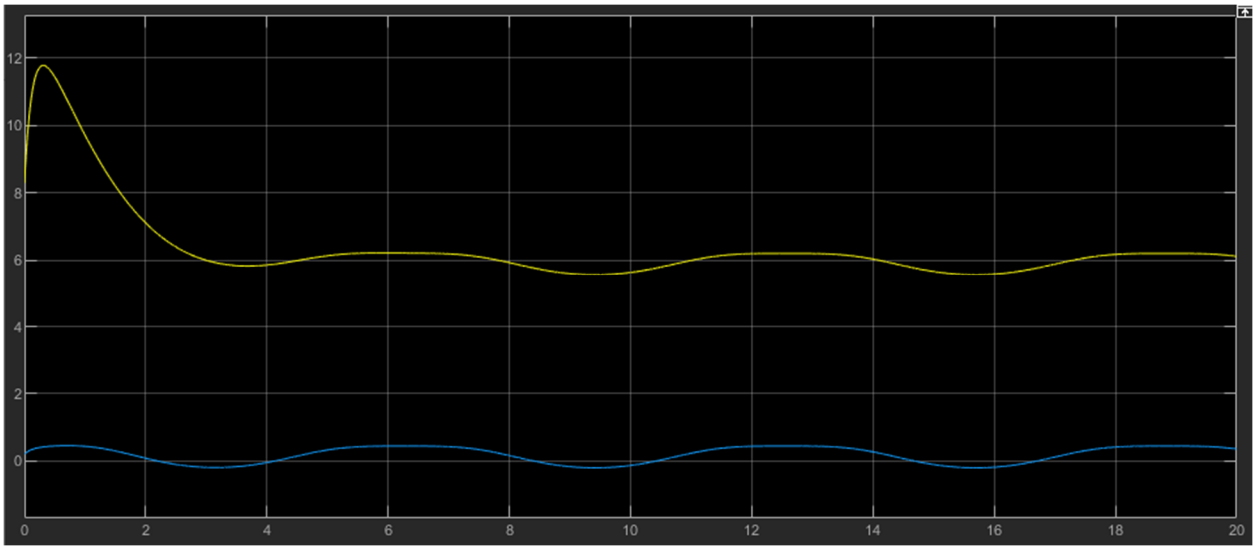
- The position $q(t)$ and velocity $\dot{q}(t)$ start from the correct initial conditions.
- $q_1(t)$ stabilizes and reaches its setpoint target of $q_{d,1}(t) = \pi/6$.
- $q_2(t)$ is correctly tracking its target of $q_{d,2}(t) = \cos(t)$.
- The input becomes periodic, which makes sense, because it is tracking a sinusoid.
- $\dot{q}_1(t)$ stabilizes and reaches zero, makes sense, because $q_1(t)$ becomes constant.
- $\dot{q}_2(t)$ becomes sinusoidal, makes sense, because $q_2(t)$ tracks a sinusoidal target.

For the Computed Torque Controller, the positive-definite gain matrices were set to:

```
Kv = [10, 0; 0, 10];
Kp = [10, 0; 0, 10];
```


q
q_d
1.5
1
0.5
0
-0.5
-1
0
2
4
6
8
10
12
14
16
18
20


ROBOT/2:1
ROBOT/2:2
1
0.5
0
-0.5
-1
0
2
4
6
8
10
12
14
16
18
20


12
10
8
6
4
2
0
0
2
4
6
8
10
12
14
16
18
20