Control design of robot arm process

7.1 Independent Joint Control

Independent joint control is the classical control approach, where a controller is designed for each individual joint (Fig. 7.1). The effect of other joints is considered as a disturbance that the controller has to reject.

Let us first rewrite model in terms of each individual joint i ([4]):

$$d_{ii}\ddot{q}_i + c_{ii}\dot{q}_i = \tau_i - r_i^2 w_i, \quad i = 1, 2, .., n \tag{7.1}$$

where d_{ii} includes only the constant diagonal terms of $D'(q)$, while the disturbance term w_i includes all other i^{th} terms of $D'(q)$, and i^{th} components of $C(q, \dot{q})$ and $G(q)$. It is important to notice that if the the Coriolis and centripetal terms are not very large, then the process dynamics can be approximated well by *n decoupled linear second-order systems*.

7.1.1 Control problem

Design a linear controller for each joint that ensure tracking (q tracks a reference signal q_d) and disturbance rejection (w_p).

Consider standard PD controllers

$$\tau_i = K_{di}\dot{e}_i + K_{pi}e_i, \tag{7.2}$$

where e_i represents the tracking error, defined as $e_i = q_{di} - q_i$, and $\dot{e}_i = \dot{q}_{di} - \dot{q}_i$ is the error of the derivatives.

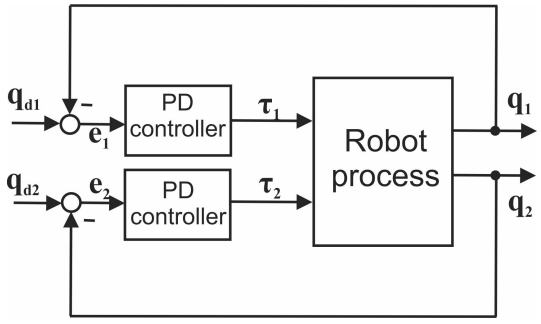

Figure 7.1: Independent joint control - PD controllers

If we consider the case of set-point tracking, that is $\dot{q}_{di} = 0$, and replace (7.2) in (7.1) we obtain:

$$d_{ii}\ddot{q}_i(t) + (c_{ii} + KK_{di})\dot{q}_i(t) + KK_{pi}q_i(t) = KK_{pi}q_{di}(t) - w_{pi}(t), \quad i = 1, 2, \tag{7.3}$$

By applying the Laplace transform we get the transfer function relating the outputs (q_i) to the reference and disturbance inputs (q_{di} and w_{pi}):

$$q_i(s) = \frac{KK_{pi}}{d_{ii}s^2 + (c_{ii} + KK_{di})s + KK_{pi}}q_{di}(s) - \frac{1}{d_{ii}s^2 + (c_{ii} + KK_{di})s + KK_{pi}}w_{pi}(s), \quad i = 1, 2, \tag{7.4}$$

The characteristic equations are

$$d_{ii}s^2 + (c_{ii} + KK_{di})s + KK_{pi} = 0,$$

which can also be written as

$$s^2 + \frac{c_{ii} + KK_{di}}{d_{ii}}s + \frac{KK_{pi}}{d_{ii}} = 0.$$

Because the standard second order equation is given by

$$s^2 + 2\zeta_i\omega_{ni}s + \omega_{ni}^2 = 0,$$

the controller parameters can be expressed in terms of damping ratio ζ and natural frequency ω_n:

$$K_{pi} = \frac{d_{ii}\omega_{ni}^2}{K}, \quad K_{di} = \frac{2\zeta_i\omega_{ni}d_{ii} - c_{ii}}{K}. \tag{7.5}$$

Usually ζ is set to 1 (critical damping), and ω_n is chosen as high as possible. One possible limitation in the value adopted for ω_n is the input torque τ_i saturation. The control strategy proves very efficient in practice. If we further want to force a very small or null steady state error e_{ss}, then we can either adopt PID type controllers

$$\tau_i = K_{di}\dot{e}_i + K_{pi}e_i + K_{ii}\int_0^t e\,dt, \tag{7.6}$$

or PD controllers with an additional gravity term [1]

$$\tau_i = K_{di}\dot{e}_i + K_{pi}e_i + G_i'(q). \tag{7.7}$$

[1] Note that in this case the control signals are not decoupled anymore.

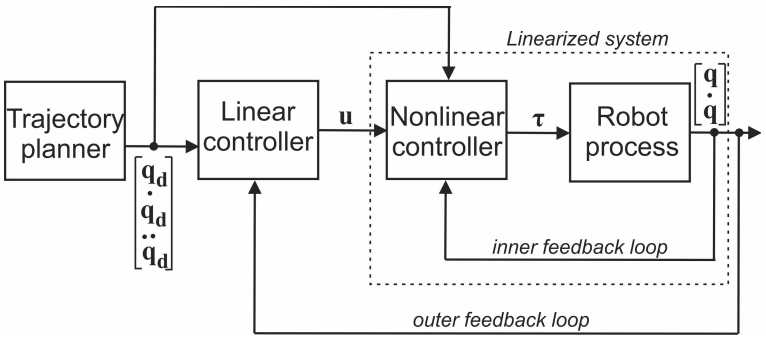

Figure 7.2: Computed torque control - feedback linearization - principal control structure

7.2 Computed Torque Control

Computed torque control (in some places called feedback linearization control) is a more sophisticated and modern control strategy that can be used to increase the control performances [2]. The control consists out of an inner feedback loop and an outer feedback loop - Figure 7.2. Although it is a nonlinear control approach, because the inner feedback loop achieves dynamic linearization, the outer feedback loop resume to a classical linear control design.

Consider the robot arm process model 7.1. By the change of notation $V(q,\dot{q}) = C(q,\dot{q})\dot{q} + G(q)$ we can rewrite the model more compactly as

$$D(q)\ddot{q} + V(q,\dot{q}) = \tau \tag{7.8}$$

Consider the following control law for the inner control loop:

$$\tau = D(q)(\ddot{q}_d - u) + V(q,\dot{q}) \tag{7.9}$$

We define the tracking error as

$$e = q_d - q. \tag{7.10}$$

Then $\dot{e} = \dot{q}_d - \dot{q} \quad and \quad \ddot{e} = \ddot{q}_d - \ddot{q}$.

By replacing the torque from 7.9 in model 7.8, and using the definition 7.10, we obtain:

$$\ddot{e} = u \tag{7.11}$$

Thus the inner feedback loop achieves dynamic linearization, in other words the outer loop "sees" a double integrator process. The double integrator model 7.11 can be written in state space form:

$$\dot{x} = Ax + Bu \tag{7.12}$$

[2]Independent joint control and Computed Torque control are the most widely encountered control techniques in the literature on Robot Control Systems ([4],[1], [5])

with

$$x = \begin{bmatrix} e \\ \dot{e} \end{bmatrix}, A = \begin{bmatrix} 0_2 & I_2 \\ 0_2 & 0_2 \end{bmatrix}, B = \begin{bmatrix} 0_2 \\ I_2 \end{bmatrix} \tag{7.13}$$

Now the outer loop can be designed by any classical linear control technique. We will consider here a state feedback control law with integrator component:

$$u = -Kx + K_i\epsilon \tag{7.14}$$

where ϵ is the output of the integrator ($\dot{\epsilon} = e$) - see Figure 7.3. We will first design the state feedback gain K using Ackerman's formula for a 4^{th} order system:

$$K = [0\,0\,0\,1]M_c^{-1}\Delta(A) \tag{7.15}$$

where Δ is the desired characteristic polynomial of the closed loop system

$$\Delta(s) = s^4 + a_3s^3 + a_2s^2 + a_1s^1 + a_0, \tag{7.16}$$

and M_c is the controllability matrix:

$$M_c = [B|AB|A^2B|A^3B]. \tag{7.17}$$

After the state feedback gain K is designed through pole placement, we will consider that the integrator gain can be designed through trial and error in simulations, in order the achieve the best tracking performances possible, while taking into account nonlinear effects like saturation, dead-zones or backlash.

Finally, the detailed control structure is shown in Figure 7.3.

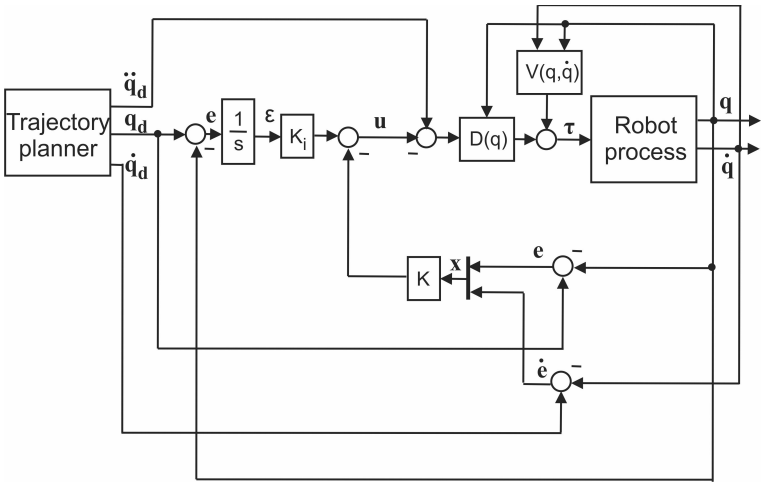

Figure 7.3: Computed torque control - feedback linearization - detailed control structure

Figure 7.4: Real robot arm process - from [2]

7.3 Numerical results

7.3.1 Robot arm example

As an example of a 2DOF robot with the structure as in Figure 6.1, consider the robot from Figure 7.4 [3]. , that we model using 7.1.

The parameters are [4]:$L_1 = 0.095\ m$, $L_2 = 0.1\ m$, $m_1 = 0.095\ kg$, $m_2 = 0.37\ kg$, $g = 9.81\ m/s^2$, $I_{1x} = 2.27\ 10^{-2}\ kg\ m^2$, $I_{2y} = 2.27\ 10{-}2\ kg\ m^2$, $b_1 = 0.24$, $b_2 = 0.16$, $r = 1$. The torque control signal is limited to the range $[-1.18, 1.18]\ Nm$.

7.3.2 Independent Joint Control

Here we will design PD controller for independent joint controller

$$\tau_1 = K_{d1}\dot{e}_1 + K_{p1}e_1,$$

$$\tau_2 = K_{d2}\dot{e}_2 + K_{p2}e_2,$$

[3] This robot was designed and built by Zoltan Nagy - see [2]

[4] The parameters were either measured or estimated - see [2] for details.

for the robot process 7.1, with the parameters given in previous section. We will use the formula 7.5, with $B = 0, J_{p1} = 0.0263, J_{p2} = 0.0236$, $\zeta = 1$, $K = 1$. We started from a value for $\omega_n = \omega_{n1} = \omega_{n2}$ of 0.1, an increased it until the step response of the closed loop system is fast enough, and the control torques reach saturation for a small time interval. In the end, a value of 12 provided good enough results.

The implementation of the controllers and process was done in Matlab/Simulink. Figure 7.5 shows simulations results of the closed loop system with step reference signals, while Figure 7.6 shows the control torques. The results show that a small steady state is present. This can be eliminated by adding a gravity term as in 7.7. The results with PD+gravity control are shown in Figure 7.7. It can be noticed that now the steady state error is zero.[5] A more interesting and demanding tracking scenario is that when the reference signals are sinusoidal ($q_d(t) = sin(t)$), which is shown Figure 7.8 for the PD+gravity controllers. Although the steady state error is now exactly zero, because the reference continuously changes, the error is kept into very small limits.

[5]Note that in the case of PD+gravity controller, the controller is no longer joint independent, due to the gravity terms that contain expressions in both joint variables q_1 and q_2.

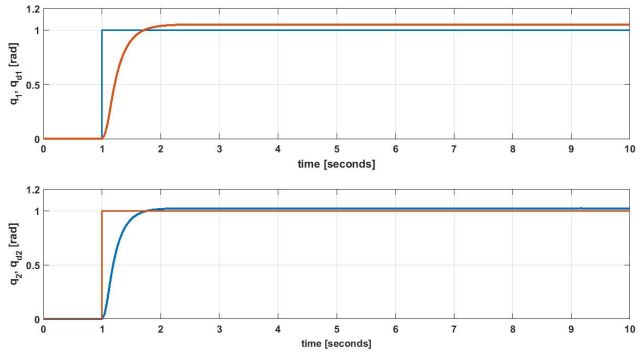

Figure 7.5: Simulations with PD independent joint control - step response

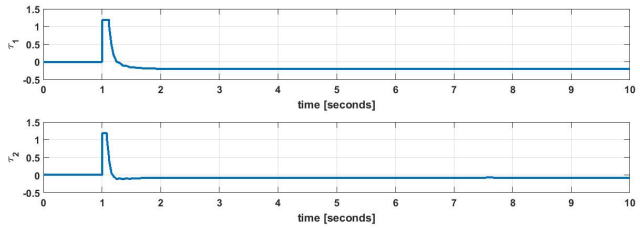

Figure 7.6: Control torques for PD independent joint control

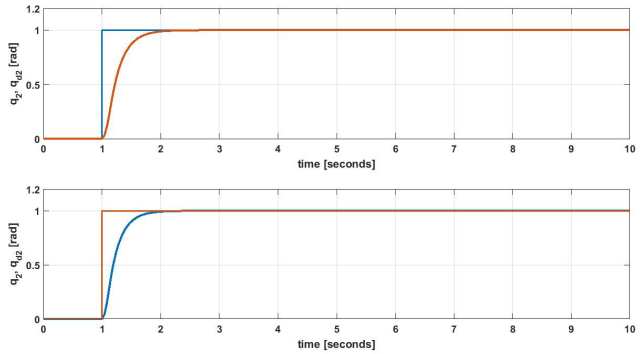

Figure 7.7: Simulations with PD+Gravity joint control - step response

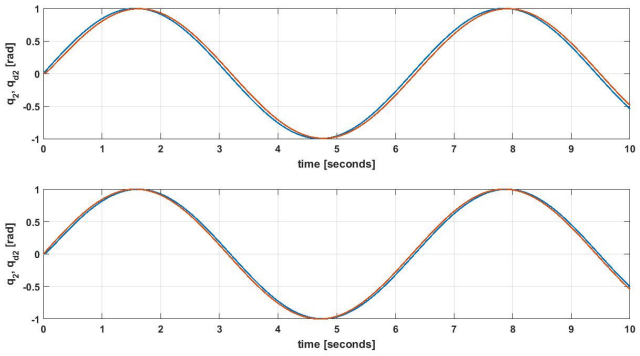

Figure 7.8: Simulations with PD+Gravity joint control - sinusoidal response

7.3.3 Computed Torque Control

Consider the inner feedback loop with control law 7.9, and the outer feedback loop with control law 7.14 (Figure 7.2).

We will design the state feedback gain K for the double integrator process 7.11, through pole placement. Consider the following closed loop pole configuration [-4 -4 -9 -9] (desired eigenvalues for $A - BK$). The poles where chosen such that we get an overdamped and fast enough response, while avoiding saturation as much as possible [6]. We obtained the gain values

$$K = \begin{bmatrix} 36 & 0 & 13 & 0 \\ 0 & 36 & 0 & 13 \end{bmatrix},$$

using the *place* function in Matlab.

The integrator gain K_i is found through trial and error in simulations: we increase the gain starting from an initial value of 0.1 - until the maximum steady state error starts to increase. We keep the previous iterated value. Thus, we finally arrive at the value 0.6 (that is $K_i = [0.6 \quad 0.6]$).

For a better comparison between the performances of the PD+gravity control approach versus the computed torque control, we reconsider the sinusoidal reference input from the previous section, but now we will look directly at the tracking errors defined as $e_1 = q_{d1} - q_1$, $e_2 = q_{d2} - q_2$. Figure 7.9 shows the tracking errors for the two joint positions in the case of PD+gravity control. The maximum error is about 0.05 rad (2.8 deg). For the Computed Torque control the results are shown in Figure 7.10. Notice that the maximum error is now much smaller 0.01 rad (0.5 deg). Of course that the cost is an increase complexity for the controller - the implementation requires a considerable increase in computational power.

[6] As the poles are moved farther to the left in the complex plane, the response becomes faster, but the control effort increases.

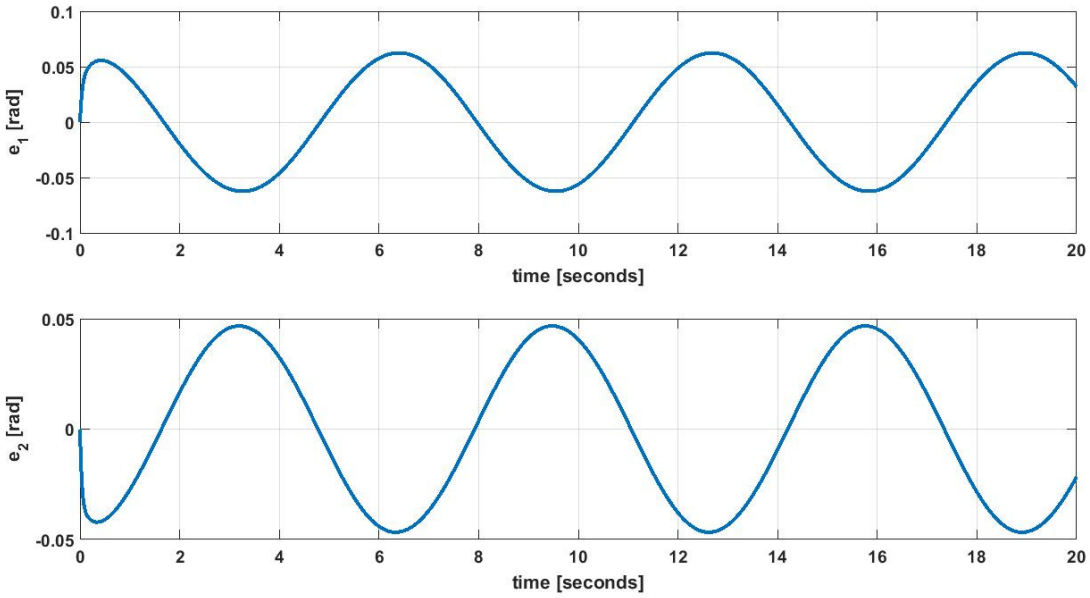

Figure 7.9: Tracking error for PD+grav control

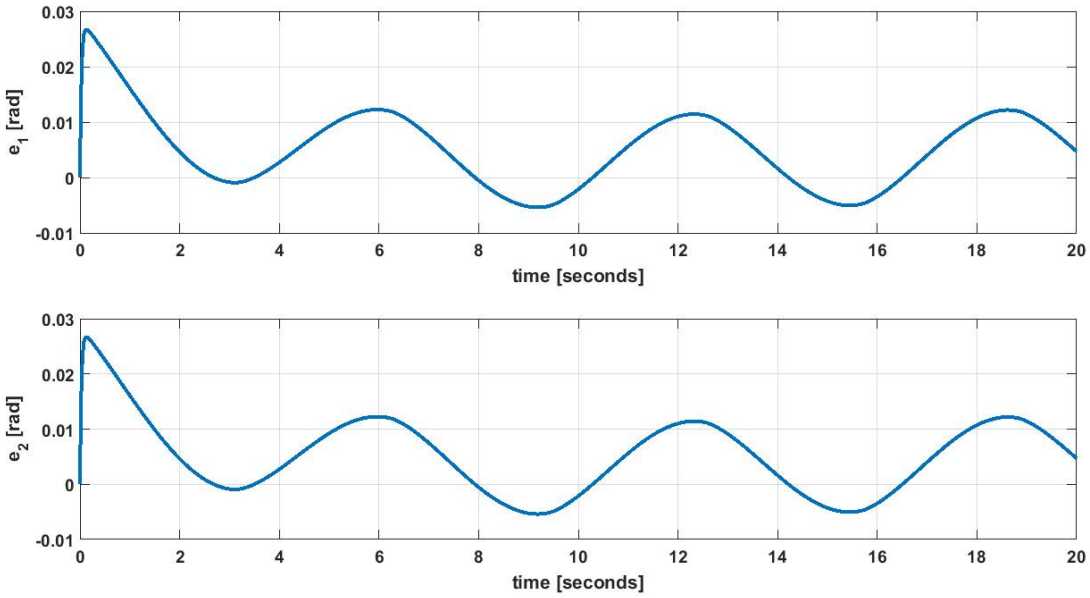

Figure 7.10: Tracking error for Computer Torque Control

7.4 Proposed problems

1. Consider the 2DOF robotic structure from laboratory 6. Using the Robot_control.mdl in the laboratory folder:

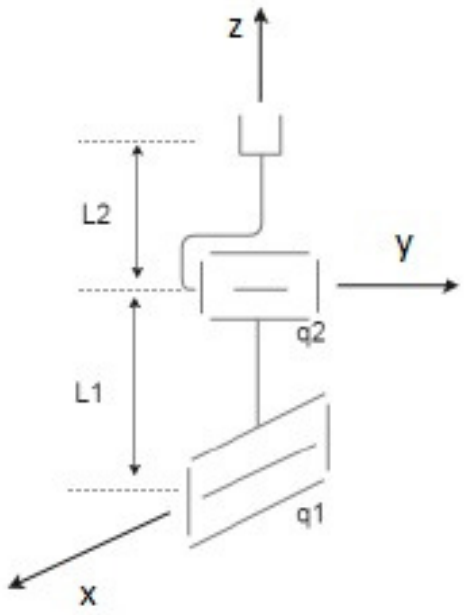

Figure 7.11: Schematic representation of a 2DOF robot arm

a) implement in Matlab Simulink an Independent Joint Control with PD controllers.

- Use sinusoidals of different frequencies as input joint trajectories.
- Add a saturation of -1.18, 1.18 for the input torque.
- Adapt the natural frequencies and observe their influence on the movement and position of the robot.
- Add an integrator and tune its gain manually.

b) Implement Simulink a PI Computed Torque Controller for the same robot. Use the same trajectories as in the previous example.