Chapter 11
Position Control of a PM Brushed DC Motor

In this chapter, we continue with the control of a permanent magnet (PM) brushed DC motor, but now the position control problem is considered. To this aim, the motor model in (10.23) is considered after a power amplifier and a current loop have been included (see Sects. 10.2 and 10.3). This model is rewritten here for ease of reference:

$$\theta(s) = \frac{1}{s(s+a)}[kI^*(s) - \frac{1}{J}T_p(s)], \tag{11.1}$$
$$a = \frac{b}{J}, \quad k = \frac{nk_m}{J},$$

where $\theta(s)$ is the angular position to be controlled, $I^*(s)$ is the input or the control variable, and $T_p(s)$ is an external torque disturbance.

11.1 Identification

In this section, an experiment is designed that allows us to know the values of the constants k and a in the model (11.1). In the following, the way of performing this experiment is explained. Suppose that no external disturbance is applied, i.e., $T_p(s) = 0$; hence, the model (11.1) becomes:

$$\theta(s) = \frac{k}{s(s+a)}I^*(s). \tag{11.2}$$

Suppose that i^* is designed as a proportional position controller, i.e.,

$$i^* = k_p(\theta_d - \theta), \tag{11.3}$$

© Springer International Publishing AG, part of Springer Nature 2019

V. M. Hernández-Guzmán, R. Silva-Ortigoza, *Automatic Control with Experiments*,
Advanced Textbooks in Control and Signal Processing,
https://doi.org/10.1007/978-3-319-75804-6_11

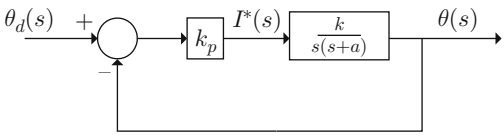

Fig. 11.1 Proportional position control

or, using the Laplace transform:

$$I^*(s) = k_p(\theta_d(s) - \theta(s)), \tag{11.4}$$

where k_p is a positive constant. A block diagram combining the expressions (11.2) and (11.4) is shown in Fig. 11.1. The closed-loop transfer function is:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{k_p k}{s^2 + as + k_p k} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \tag{11.5}$$

$$\omega_n = \sqrt{k_p k}, \quad \zeta = \frac{a}{2\sqrt{k_p k}}. \tag{11.6}$$

According to (11.5) and Sect. 3.3.1, if the desired position θ_d is a step the obtained time response $\theta(t)$ presents an overshoot and a rise time given as:

$$M_p(\%) = 100 \times e^{-\frac{\zeta}{\sqrt{1-\zeta^2}}\pi}, \tag{11.7}$$

$$t_r = \frac{1}{\omega_d}\left[\pi - \arctan\left(\frac{\sqrt{1-\zeta^2}}{\zeta}\right)\right], \tag{11.8}$$

if $0 \leq \zeta < 1$. This can always be achieved using a large enough value for k_p. From (11.7) and (11.8), the following is found:

$$\zeta = \sqrt{\frac{\ln^2\left(\frac{Mp(\%)}{100}\right)}{\ln^2\left(\frac{Mp(\%)}{100}\right) + \pi^2}}, \tag{11.9}$$

$$\omega_d = \frac{1}{t_r}\left[\pi - \arctan\left(\frac{\sqrt{1-\zeta^2}}{\zeta}\right)\right], \tag{11.10}$$

$$\omega_n = \frac{\omega_d}{\sqrt{1-\zeta^2}}. \tag{11.11}$$

This allows us to compute k and a using the following procedure:

- Implement the proportional controller shown in (11.4) or (11.3). Ensure that no external disturbance is applied to the motor. Propose some known positive value for k_p. This value must be large enough to produce a clear overshoot on the

position response θ when a step change is commanded in the desired position θ_d. Plot the position θ versus time when a step change is commanded in the desired position θ_d.
- Measure $M_p(\%)$ and t_r.
- Use (11.9)–(11.11) to compute ζ and ω_n.
- Using the expressions in (11.6) and k_p employed in the experiment, compute k and a.

The results obtained when performing the experiment described in the above procedure are shown in Fig. 11.2. $k_p = 0.5$ was employed. By direct measurement in Fig. 11.2, the following is found:

$$M_p(\%) = 78.2\%, \quad t_r = 0.09[\text{s}].$$

These measurements are simplified and rendered more exact if a computer program is employed. Following the above procedure, the following is found:

$$k = 675.4471, \quad a = 2.8681. \tag{11.12}$$

Finally, it is important to state that, as in the case of the velocity model, it is not necessary to compute the value of the factor $1/J$ appearing as coefficient of the disturbance $T_p(s)$ in (11.1). This is because T_p is also unknown and, despite this, its effect can be compensated for.

11.2 Position Control When External Disturbances Are Not Present ($T_p = 0$)

The controller design is intended to ensure that the closed-loop transient and steady-state response specifications is achieved. The transient response specifications refer to the desired rise time and overshoot. These specifications are satisfied by suitably assigning the closed-loop poles. On the other hand, the steady-state specifications are achieved by ensuring that the measured position θ reaches its desired value θ_d as time increases. The controller is a device designed in such a way that the desired transient and steady-state response specifications are satisfied.

Suppose that the desired position is a constant or a step, i.e., $\theta_d = A$ is a constant. As the plant in (11.1) has one pole at $s = 0$, then the system type is 1, which ensures that $\theta = \theta_d = A$ in a steady state, if a proportional position controller is used and no external disturbance is present, i.e., if $T_p = 0$. This means that the requirement to reach the desired position is naturally satisfied by the plant, i.e., the DC motor. Hence, the controller design is to be performed merely to satisfy the transient response specifications, i.e., to suitably assign the closed-loop poles. In the case of a DC motor, there are two simple controllers allowing the closed-loop poles to be suitably located: (i) Proportional position control with velocity feedback. (ii) A lead compensator.

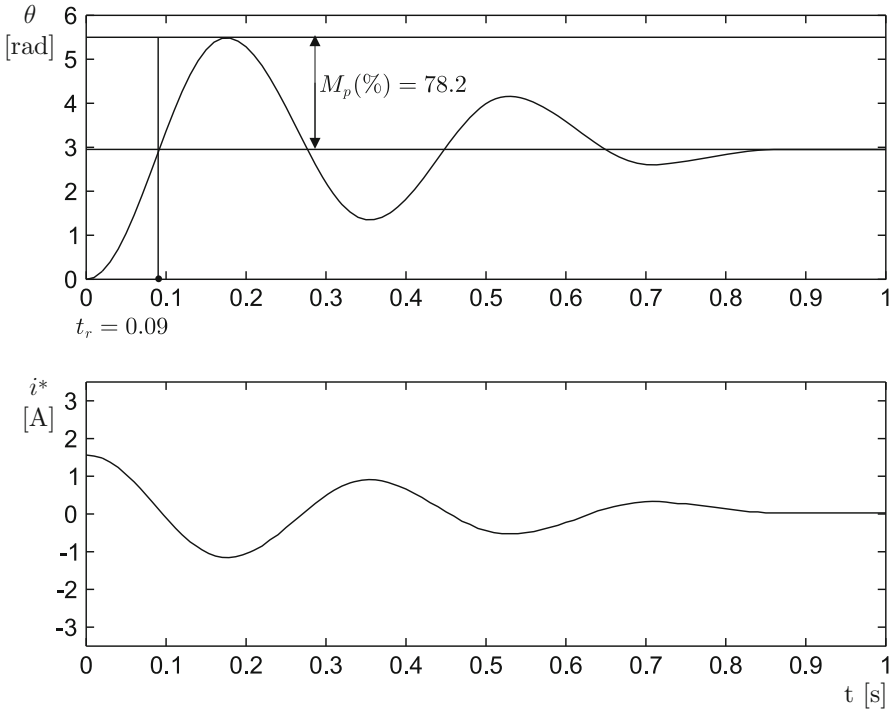

Fig. 11.2 Experimental identification

11.2.1 Proportional Position Control with Velocity Feedback

This controller is given as:

$$i^*(t) = k_p(\theta_d - \theta) - k_v\dot{\theta}, \tag{11.13}$$

where k_p and k_v are positive constants, or using the Laplace transform:

$$I^*(s) = k_p(\theta_d(s) - \theta(s)) - k_v s\theta(s). \tag{11.14}$$

The block diagram resulting from the combination of (11.14) and (11.2) is depicted in Fig. 11.3. The closed-loop transfer function is:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{k_p k}{s^2 + (a + k_v k)s + k_p k} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2},$$
$$\omega_n = \sqrt{k_p k}, \quad \zeta = \frac{a + k_v k}{2\sqrt{k_p k}}. \tag{11.15}$$

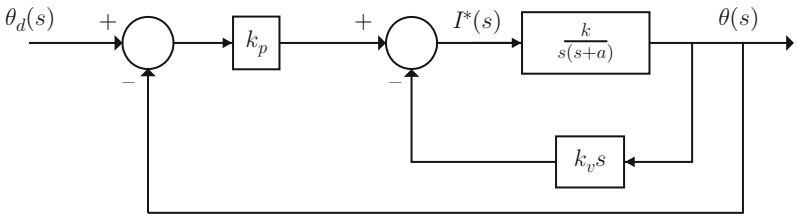

Fig. 11.3 Proportional position control with velocity feedback

Thus, it is always possible to find positive values for k_v and k_p such that the desired ζ and ω_n are obtained. This means that it is always possible to assign the closed-loop poles $(s+\zeta\omega_n+j\omega_d)(s+\zeta\omega_n-j\omega_d) = s^2+2\zeta\omega_n s+\omega_n^2$, at any point on the left half-plane, by a suitable selection of k_p and k_v. For instance, consider the numerical values in (11.12). Suppose that overshoot and rise time are desired to be:

$$M_p(\%) = 20\%, \quad t_r = 0.068[\text{s}]. \tag{11.16}$$

Using these data and (11.9)–(11.11), it is found that the closed-loop poles must be located at:

$$s = -\zeta\omega_n \pm j\omega_d = -15.4009 \pm j30.0623,$$

with $\zeta = 0.4559$ and $\omega_n = 33.7776$. Finally, using these values, (11.12), (11.15), and (11.16), it is found that the following controller gains are required to achieve the desired rise time and overshoot:

$$k_p = 1.6891, \quad k_v = 0.0414. \tag{11.17}$$

Some simulation results obtained when using these controller gains, the motor parameters in (11.12), and the control scheme in Fig. 11.3 are shown in Fig. 11.4. The desired position is $\theta_d = 1.5$[rad]. From these results, it is possible to verify that overshoot is 20% and rise time is 0.068[s], as desired. This simulation is performed using the following MATLAB code in an m-file:

```
k=675.4471;
a=2.8681;
kp=1.6891;
kv=0.0414;
gm=tf(k,[1 a 0]);
velfeedck=tf(kv*[1 0],1);
gc1=feedback(gm,velfeedck,-1);
M=feedback(kp*gc1,1,-1);
figure(1)
```

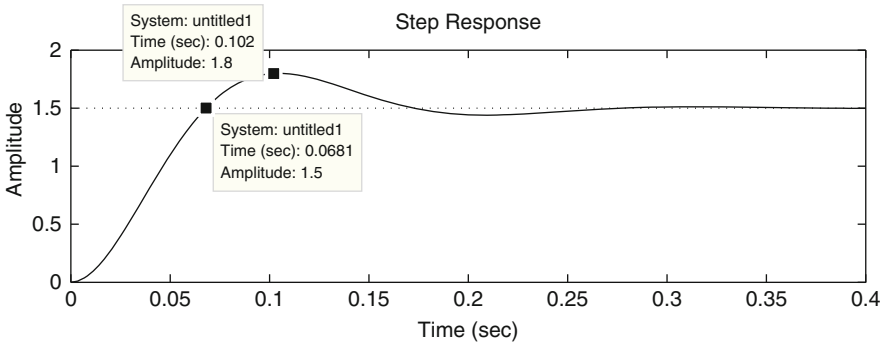

Fig. 11.4 Proportional position control with velocity feedback. Simulation results

```
subplot(2,1,1)
step(M*1.5)
```

The experimental results obtained with controller (11.13) and gains in (11.17), when θ_d is a step, are shown in Fig. 11.5. Overshoot and rise time measured in this experiment are:

$$M_p(\%) = 16\%, \quad t_r = 0.068[\text{s}],$$

which are close to the design specifications in (11.16).

It is worth stating that the use of a dedicated sensor to measure velocity implies several mechanical modifications that complicate the prototype construction and increase cost. Because of this, it is common to estimate velocity from the measured position in practice. In the experiments shown in Fig. 11.5, the velocity $\dot{\theta}$ has been estimated by numeric differentiation of the measured position (see Appendix F). However, this process has the effect of a *high-pass filter*: high-frequency signals, i.e., noise, are strongly amplified by the velocity feedback of the controller. The noise content may result in control system performance deterioration because motor vibration may occur, even at rest. Thus, the gain k_v must always be kept small. In the following section, a controller is proposed that needs neither velocity measurements nor position differentiation to suitably assign the closed-loop poles.

11.2.2 A Lead Compensator

This controller is defined by the transfer function:

$$\frac{I^*(s)}{E(s)} = \gamma \frac{s+d}{s+c}, \tag{11.18}$$

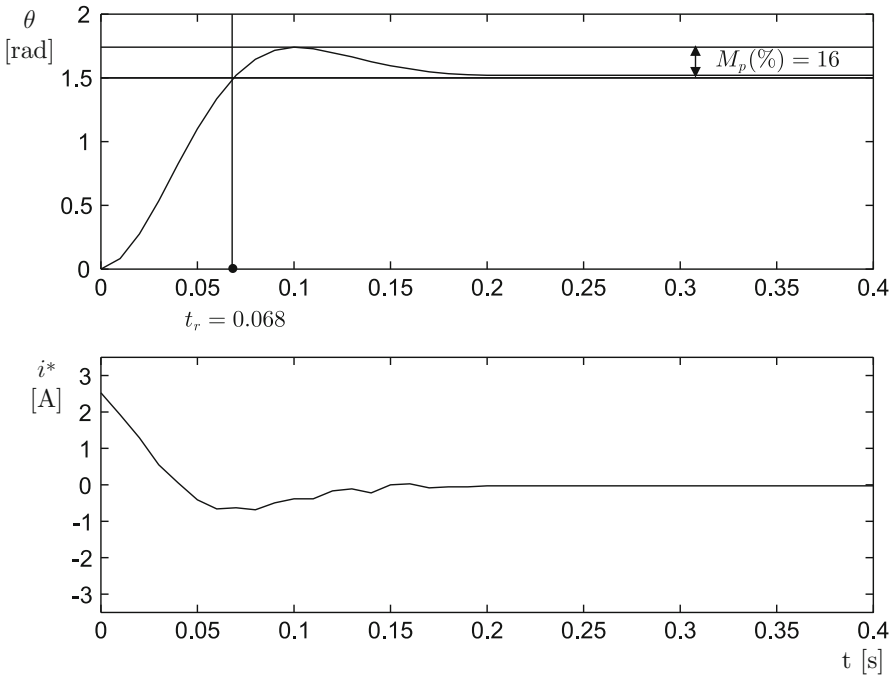

Fig. 11.5 Proportional position control with velocity feedback. Experimental results ($\theta_d = 1.5$[rad])

where $E(s)$ is the Laplace transform of the difference $e = \theta_d - \theta$, i.e., the system error, whereas γ, d and c are positive constants to be computed and they are constrained to satisfy $d < c$.

The block diagram resulting from the connection of (11.18) and (11.2) is shown in Fig. 11.6. Note that the open-loop transfer function:

$$G(s)H(s) = \frac{\gamma k(s+d)}{s(s+c)(s+a)}, \tag{11.19}$$

has three poles, i.e., the closed-loop transfer function also has three poles and one zero at $s = -d$. It is left as an exercise for the reader to find the closed-loop transfer function to verify that it has the form:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{\gamma k(s+d)}{(s-p_1)(s-p_2)(s-p_3)}. \tag{11.20}$$

In this case, besides assigning the complex conjugate poles p_1 and p_2 at the desired locations, it must be ensured that the third pole, at p_3, and the zero at $s = -d$ do not significantly modify the transient response. As discussed in Sect. 5.2.3, this

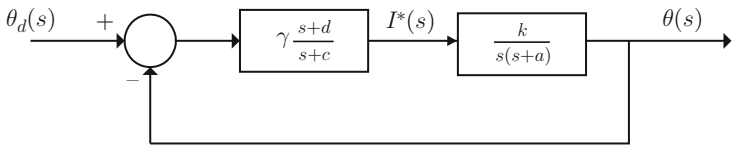

Fig. 11.6 Position control with a lead compensator

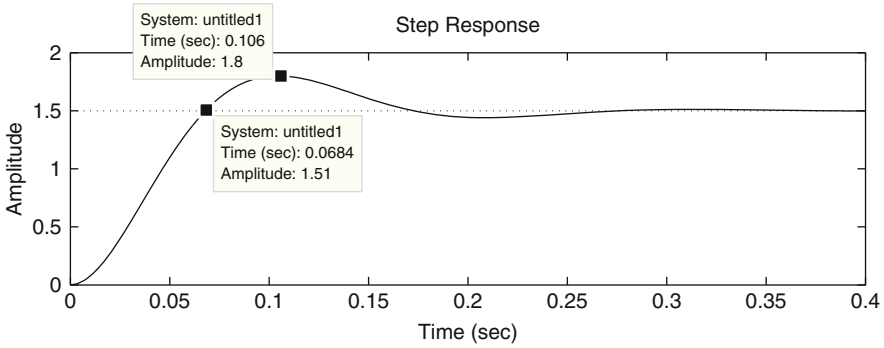

Fig. 11.7 Position control with a lead compensator. Simulation results ($\theta_d = 1.5$[rad])

is achieved by setting:

$$d = a, \quad c = 2\zeta\omega_n, \quad \gamma = \frac{\omega_n^2}{k}.$$

Using this tuning rule and the parameters in (11.12), it is found that use of the gains:

$$c = 30.8018, \qquad \gamma = 1.6891, \qquad d = 2.8681,$$

in the controller in (11.18) assigns two closed-loop poles at:

$$s = -\zeta\omega_n \pm j\omega_d = -15.4009 \pm j30.0623.$$

This specifies by design a 0.068[s] rise time and a 20% overshoot.

The simulation results with c, d, γ, as above, the motor parameters in (11.12), and the control scheme in 11.6 are presented in Fig. 11.7. The reader can verify that overshoot is 20% and rise time is 0.068[s], as desired. These simulations were performed by executing the following MATLAB code in an m-file:

```
k=675.4471;
a=2.8681;
gm=tf(k,[1 a 0]);
d=a;
z=0.4559;
```

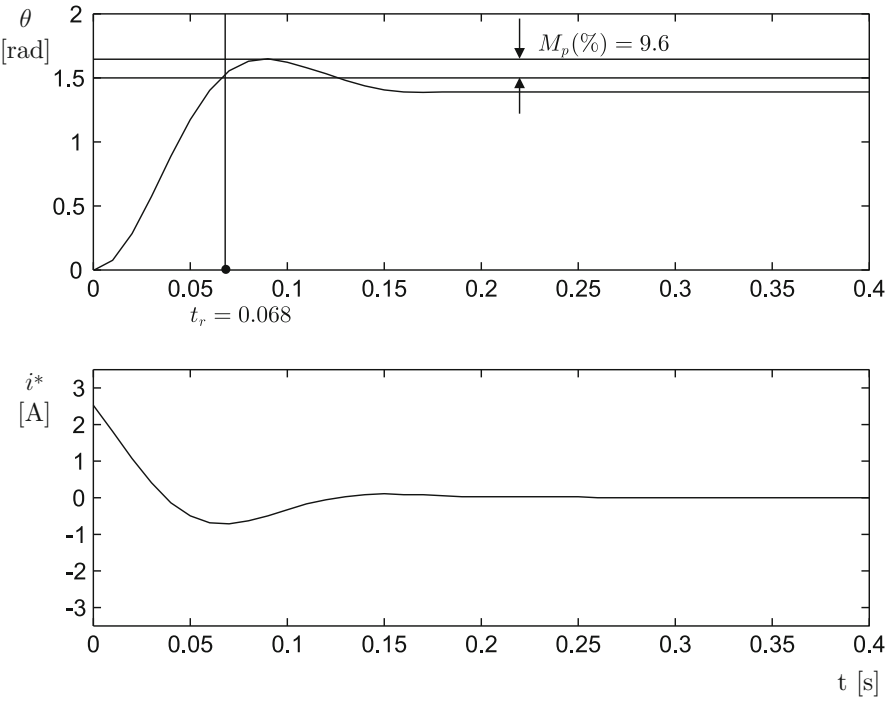

Fig. 11.8 Position control with a lead compensator. Experimental results ($\theta_d = 1.5$[rad])

```
wn=33.7776;
c=2*z*wn
gamma=wn^2/k
gc=tf(gamma*[1 d],[1 c]);
M=feedback(gc*gm,1,-1)
figure(1)
subplot(2,1,1)
step(M*1.5,0.4)
```

Some experimental results obtained when $\theta_d = 1.5$[rad] is a step are shown in Fig. 11.8. There, a $M_p(\%) = 9.6\%$ overshoot and a $t_r = 0.068$[s] rise time are measured. Note that these values are close to the design values: $t_r = 0.068$[s] and $M_p(\%) = 20\%$. Finally, it is important to stress that the use of the lead compensator (Fig. 11.8) also results in a small steady-state error. This steady-state error is due to the presence of nonlinear friction: static and Coulomb (see (2.28)).

11.3 Control Under the Effect of External Disturbances

In this section, a constant external disturbance is considered, i.e.,

$$T_p = t_d, \quad T_p(s) = \frac{t_d}{s},$$

where t_d is a constant different from zero. This means that the possibility that an external agent applies a torque at the motor shaft is considered. The controller design must be performed such that the effect of such a disturbance on the system output must vanish as fast as possible.

In this class of applications, it is common to employ a proportional–integral–derivative (PID) controller because its integral part is intended to cope with the deviations due to the constant disturbance. However, as stated in Sect. 5.2.5, a tuning rule is not at the disposal of the designer, allowing us to compute the PID controller gains to simultaneously achieve the desired transient response specifications (rise time and overshoot) and a satisfactory external disturbance rejection. Motivated by this situation, two new controllers are introduced in the following sections that are specially designed to simultaneously achieve these specifications.

11.3.1 A Modified PID Controller

The use of the inverse Laplace transform in (11.1) yields the following differential equation:

$$\ddot{\theta} + a\dot{\theta} = ki^* - \frac{1}{J}T_p. \tag{11.21}$$

It is shown next that the following controller allows us to find a tuning rule, solving the problem described above:

$$\begin{aligned} i^* = \kappa_p(\theta_d - \theta) - \kappa_d\dot{\theta} + k_1\kappa_p k \int_0^t (\theta_d - \theta(r))dr - k_1(\dot{\theta} - \dot{\theta}(0)), \\ -k_1(a + \kappa_d k)(\theta - \theta(0)), \end{aligned} \tag{11.22}$$

where θ_d is a constant standing for the desired position whereas $\theta(0)$ and $\dot{\theta}(0)$ represent the position and the velocity initial values respectively. Using the fact that:

$$\dot{\theta} - \dot{\theta}(0) = \int_0^t \ddot{\theta}(r)dr,$$

$$\theta - \theta(0) = \int_0^t \dot{\theta}(r)dr,$$

the above-mentioned controller can be written as:

$$\begin{aligned} i^* &= \kappa_p(\theta_d - \theta) - \kappa_d\dot{\theta} + k_1\kappa_p k\int_0^t (\theta_d - \theta(r))dr - k_1\int_0^t \ddot{\theta}(r)dr \\ &\quad -k_1(a+\kappa_d k)\int_0^t \dot{\theta}(r)dr, \\ &= \kappa_p(\theta_d - \theta) - \kappa_d\dot{\theta} - k_1\int_0^t \left[\ddot{\theta}(r) + (a+\kappa_d k)\dot{\theta}(r) + \kappa_p k(\theta(r) - \theta_d)\right]dr. \end{aligned}$$

Replacing this in (11.21), the following is found:

$$\begin{aligned} &\ddot{\theta} + (a+\kappa_d k)\dot{\theta} + \kappa_p k(\theta - \theta_d) \\ &+k_1 k\int_0^t \left[\ddot{\theta}(r) + (a+\kappa_d k)\dot{\theta}(r) + \kappa_p k(\theta(r) - \theta_d)\right]dr = -\frac{1}{J}T_p. \end{aligned}$$

This can be written as:

$$\dot{\xi} + k_1 k\xi = -\frac{1}{J}T_p, \tag{11.23}$$

if the following is defined:

$$\xi = \int_0^t \left[\ddot{\theta}(r) + (a+\kappa_d k)\dot{\theta}(r) + \kappa_p k(\theta(r) - \theta_d)\right]dr.$$

Applying the Laplace transform to (11.23) and assuming all initial conditions to be zero, the following is obtained:

$$\xi(s) = \frac{-\frac{1}{J}}{s + k_1 k}T_p(s),$$

or:

$$s\xi(s) = \frac{-\frac{1}{J}s}{s + k_1 k}T_p(s). \tag{11.24}$$

If the disturbance is constant, i.e., $T_p(s) = \frac{t_d}{s}$, then the final value theorem can be used to find:

$$\begin{aligned} \lim_{t\to\infty}\dot{\xi}(t) &= \lim_{s\to 0} s[s\xi(s)], \\ &= \lim_{s\to 0} s\frac{-\frac{1}{J}s}{s + k_1 k}\frac{t_d}{s} = 0. \end{aligned}$$

Hence, if $k_1 k > 0$, the filter in (11.24) is stable and it is ensured that $\lim_{t\to\infty} \dot{\xi}(t) = 0$. Moreover, the larger is $k_1 k > 0$ the faster $\dot{\xi}(t)$ approaches zero. Note that $\lim_{t\to\infty} \dot{\xi}(t) = 0$ implies that:

$$\begin{aligned}&\frac{d}{dt}\int_0^t \left[\ddot{\theta}(r) + (a + \kappa_d k)\dot{\theta}(r) + \kappa_p k(\theta(r) - \theta_d)\right] dr,\\ &= \ddot{\theta} + (a + \kappa_d k)\dot{\theta} + \kappa_p k(\theta - \theta_d) \to 0.\end{aligned}$$

This means that the effect of a constant disturbance vanishes as time increases and the only thing that remains is the differential equation $\ddot{\theta} + (a + \kappa_d k)\dot{\theta} + \kappa_p k\theta = \kappa_p k\theta_d$. This differential equation is stable if $\kappa_p > 0$ and $\kappa_d > 0$, has a unit gain in a steady state, and the gains κ_p, κ_d, can be used to arbitrarily assign the poles of the system. Thus:

- If no disturbance is present ($T_p = 0$ and $\dot{\xi}(t) = 0$ for all $t \geq 0$), the response in the position is as the response of a second-order system with rise time and overshoot that can be arbitrarily assigned by suitably selecting κ_p and κ_d. The position θ reaches the desired position θ_d (constant) in a steady state.
- If a constant disturbance appears ($T_p = t_d \neq 0$), the deviation that it produces vanishes as time increases. Moreover, if a larger k_1 is chosen (such that the product $k_1 k > 0$ is larger), then the deviation due to the disturbance vanishes faster. If, once the deviation due to a disturbance vanishes (once $\theta = \theta_d$), a step change in the desired position θ_d is commanded (i.e., when the disturbance T_p is still present), then the position response is identical to that obtained when $T_p = 0$, i.e., the rise time and overshoot are identical as before and the position θ reaches its desired value θ_d in a steady state.

As all initial conditions are assumed to be zero in classical control, then $\theta(0) = 0$, $\dot{\theta}(0) = 0$, can be assumed and the controller in (11.22) becomes:

$$\begin{aligned}i^* &= \kappa_p(\theta_d - \theta) - \kappa_d\dot{\theta} + k_1\kappa_p k\int_0^t (\theta_d - \theta(r))dr - k_1\dot{\theta} - k_1(a + \kappa_d k)\theta,\\ &= [\kappa_p + k_1(a + \kappa_d k)](\theta_d - \theta) - (\kappa_d + k_1)\dot{\theta} + k_1\kappa_p k\int_0^t (\theta_d - \theta(r))dr\\ &\quad -k_1(a + \kappa_d k)\theta_d,\\ &= K_P(\theta_d - \theta) - K_D\dot{\theta} + K_I\int_0^t (\theta_d - \theta(r))dr - k_1(a + k_d k)\theta_d, \qquad (11.25)\\ K_P &= \kappa_p + k_1(a + \kappa_d k), \quad K_D = \kappa_d + k_1, \quad K_I = k_1\kappa_p k,\end{aligned}$$

which constitutes a simple PID controller with a constant feedforward term. Note that the derivative term does not contain a time derivative of the desired position. Finally, the tuning rule of the controller in (11.25) is summarized as follows:

- $\kappa_p > 0$ and $\kappa_d > 0$ are chosen to fix the desired rise time and overshoot by suitably assigning the roots of the characteristic polynomial $s^2 + (a + \kappa_d k)s + \kappa_p k$, i.e.,

$$a + \kappa_d k = 2\zeta\omega_n, \quad \kappa_p k = \omega_n^2. \tag{11.26}$$

- $k_1 > 0$ is chosen large for a fast disturbance rejection. This can be done by trial and error or it can be computed recalling that $\frac{1}{k_1 k}$ is the time constant of the filter in (11.24), which is responsible for the disturbance rejection.

Another, better known, way of designing a PID controller with the same advantages is by using two-degrees-of-freedom controllers. These is presented in the next section.

11.3.2 A Two-Degrees-of-Freedom Controller

The block diagram of a two-degrees-of-freedom controller is shown in Fig. 11.9. $G_p(s)$ stands for the plant whereas the controller is composed of two parts: $G_{c1}(s)$ and $G_{c2}(s)$. In Example 4.5, Sect. 4.1, it is shown that the closed-loop system output is given as:

$$\theta(s) = G_1(s)\theta_d(s) + G_2(s)D(s),$$

where:

$$\begin{aligned} G_1(s) &= \frac{G_{c1}(s)G_p(s)}{1 + (G_{c1}(s) + G_{c2}(s))G_p(s)}, \\ G_2(s) &= \frac{G_p(s)}{1 + (G_{c1}(s) + G_{c2}(s))G_p(s)}, \end{aligned} \tag{11.27}$$

are the transfer functions defined as:

$$\begin{aligned} G_1(s) &= \frac{\theta(s)}{\theta_d(s)}, \quad \text{when } D(s) = 0, \\ G_2(s) &= \frac{\theta(s)}{D(s)}, \quad \text{when } \theta_d(s) = 0. \end{aligned}$$

The reason why this control scheme is called *two-degrees-of-freedom-controller* is because the transfer functions defined in (11.27) can be tuned independently from each other using for that, as independent variables (or degrees-of-freedom), the two controller components $G_{c1}(s)$ and $G_{c2}(s)$. It is desired to design $G_{c1}(s)$ and $G_{c2}(s)$ such that: (i) The transient and steady-state response specifications to a position reference $\theta_d(s)$ are satisfied. (ii) The effect of an external disturbance, $D(s)$, is taken to zero as fast as desired.

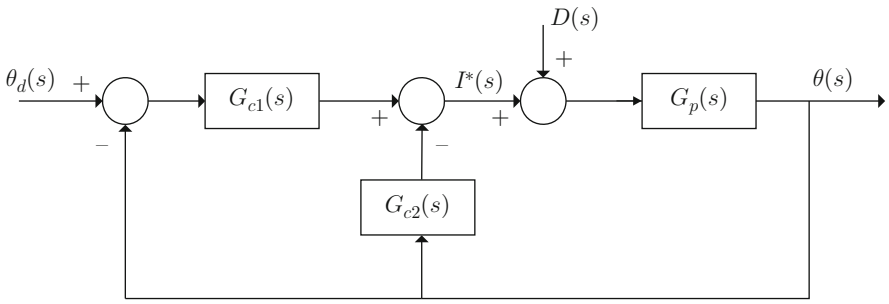

Fig. 11.9 Closed-loop system with a two-degrees-of-freedom controller

As the plant under study is a PM brushed DC motor, assume that:

$$G_p(s) = \frac{k}{s(s+a)}. \tag{11.28}$$

The controller design presented in this section assumes that $G_{c1}(s)$ and $G_{c2}(s)$ are PID controllers [5], chapter 10; hence:

$$\begin{aligned} G_{c1}(s) &= \frac{k_{d1}s^2 + k_{p1}s + k_{i1}}{s}, \\ G_{c2}(s) &= \frac{k_{d2}s^2 + k_{p2}s + k_{i2}}{s}, \\ G_c(s) = G_{c1}(s) + G_{c2}(s) &= \frac{k_d s^2 + k_p s + k_i}{s}, \\ &= \frac{k_d(s+\alpha)(s+\beta)}{s} = \frac{k_d(s^2 + (\alpha+\beta)s + \alpha\beta)}{s}, \\ k_d = k_{d1} + k_{d2}, \quad k_p &= k_{p1} + k_{p2}, \quad k_i = k_{i1} + k_{i2}, \end{aligned} \tag{11.29}$$

for some constants α and β. Thus:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{G_{c1}(s)G_p(s)}{1 + G_c(s)G_p(s)}.$$

Replacing (11.28) and (11.29) in the latter expression:

$$\begin{aligned} \frac{\theta(s)}{\theta_d(s)} &= \frac{G_{c1}(s)\frac{k}{s(s+a)}}{1 + \frac{k_d(s^2+(\alpha+\beta)s+\alpha\beta)}{s}\frac{k}{s(s+a)}}, \\ &= \frac{skG_{c1}(s)}{s^2(s+a) + k_d k(s^2 + (\alpha+\beta)s + \alpha\beta)}, \end{aligned}$$

$$= \frac{skG_{c1}(s)}{s^3 + (a + k_d k)s^2 + k_d k(\alpha + \beta)s + k_d k\alpha\beta}. \tag{11.30}$$

Suppose that the desired transient response is described by two dominant complex conjugate poles given as $s = -\sigma + j\omega_d$ and $s = -\sigma - j\omega_d$, $\sigma > 0$, $\omega_d > 0$. Then, the characteristic polynomial must satisfy:

$$\begin{aligned} &s^3 + (a + k_d k)s^2 + k_d k(\alpha + \beta)s + k_d k\alpha\beta \\ &= (s + \sigma + j\omega_d)(s + \sigma - j\omega_d)(s + f), \\ &= s^3 + (2\sigma + f)s^2 + (\sigma^2 + \omega_d^2 + 2\sigma f)s + (\sigma^2 + \omega_d^2)f, \end{aligned}$$

where the following must be satisfied:

$$a + k_d k = 2\sigma + f, \tag{11.31}$$

$$k_d k(\alpha + \beta) = \sigma^2 + \omega_d^2 + 2\sigma f, \tag{11.32}$$

$$k_d k\alpha\beta = (\sigma^2 + \omega_d^2)f. \tag{11.33}$$

To avoid the pole in $s = -f$, $f > 0$, to have an effect on the transient response of (11.30), a transfer function such as the following is proposed:

$$\frac{\theta(s)}{\theta_d(s)} = \frac{\rho(s + f)}{s^3 + (2\sigma + f)s^2 + (\sigma^2 + \omega_d^2 + 2\sigma f)s + (\sigma^2 + \omega_d^2)f}, \tag{11.34}$$

where, to obtain a transfer function with a unit gain in a steady state, the following must be satisfied:

$$\rho f = (\sigma^2 + \omega_d^2)f,$$

i.e.,

$$\rho = \sigma^2 + \omega_d^2. \tag{11.35}$$

On the other hand, to render the transfer functions in (11.30) and (11.34) equal, the following must be imposed:

$$skG_{c1}(s) = \rho(s + f),$$

where:

$$G_{c1}(s) = \frac{\rho s + \rho f}{sk} = k_{p1} + k_{i1}\frac{1}{s}, \tag{11.36}$$

$$k_{p1} = \frac{(\sigma^2 + \omega_d^2)}{k}, \quad k_{i1} = \frac{(\sigma^2 + \omega_d^2)f}{k}.$$

This means that $G_{c1}(s)$ must be a proportional–integral (PI) controller. The controller $G_{c2}(s)$ is obtained solving (11.29):

$$\begin{aligned} G_{c2}(s) &= G_c(s) - G_{c1}(s), \\ &= k_d s + k_d(\alpha+\beta) + k_d\alpha\beta\frac{1}{s} - \frac{(\sigma^2+\omega_d^2)}{k} - \frac{(\sigma^2+\omega_d^2)f}{k}\frac{1}{s}, \\ &= k_{d2}s + k_{p2}, \quad k_{p2} = \frac{2\sigma f}{k}, \quad k_{d2} = k_d = \frac{2\sigma + f - a}{k}, \end{aligned} \tag{11.37}$$

where (11.31), (11.32), (11.33), have been used. This means that $G_{c2}(s)$ is a PD controller. Thus, according to Fig. 11.9, the controller is given as:

$$\begin{aligned} I^*(s) &= G_{c1}(s)(\theta_d(s) - \theta(s)) - G_{c2}(s)\theta(s), \\ &= (k_{p1} + k_{i1}\frac{1}{s})(\theta_d(s) - \theta(s)) - (k_{d2}s + k_{p2})\theta(s), \\ &= (k_{p1}+k_{p2})(\theta_d(s) - \theta(s)) - k_{d2}s\theta(s) + k_{i1}\frac{1}{s}(\theta_d(s) - \theta(s)) - k_{p2}\theta_d(s). \end{aligned}$$

Applying the inverse Laplace transform:

$$i^* = (k_{p1}+k_{p2})(\theta_d - \theta) - k_{d2}\dot{\theta} + k_{i1}\int_0^t (\theta_d - \theta(r))dr - k_{p2}\theta_d. \tag{11.38}$$

As in the previous sections, it is desired that the response to a position step command has a rise time of 0.068[s] and a 20% overshoot, which corresponds to closed-loop poles at $s = -\sigma \pm j\omega_d$, where:

$$\sigma = 15.4009, \quad \omega_d = 30.0623. \tag{11.39}$$

On the other hand, the value of f does not affect the transient response to a desired reference because its effect is canceled (see (11.34)). What is then the effect of f and how to compute it? The key to an answer is the following:

- According to (11.31), once σ, a, k, are fixed, the derivative gain k_d is larger if f is larger. According to the previous sections, large derivative gains are not desirable because of noise amplification. Hence, a criterion is to select f as small as possible.
- The transfer function in (11.27) is computed as:

$$\frac{\theta(s)}{D(s)} = \frac{ks}{(s+\sigma+j\omega_d)(s+\sigma-j\omega_d)(s+f)}.$$

  Note that the pole at $s = -f$ cannot be canceled in this transfer function, i.e., its effect is very important in the transient response to an external disturbance.

Hence, the smaller f is, the slower the disturbance effect vanishes. In this respect, note that the final value theorem can be used to verify that the zero that this transfer function has at $s = 0$ ensures that the effect of a constant disturbance vanishes in a steady state.

Thus, f must be selected such that a trade-off between a fast disturbance rejection and noise attenuation is satisfied. Propose:

$$f = 40. \tag{11.40}$$

Using the plant parameters:

$$k = 675.4471, \quad a = 2.8681,$$

and (11.36), (11.37), (11.39), (11.40), the following is found:

$$G_{c1}(s) = 1.6891 + 67.5659\frac{1}{s}, \tag{11.41}$$

$$G_{c2}(s) = 0.1006s + 1.8241. \tag{11.42}$$

According to (11.36), (11.37) and (11.38), the controller is given as:

$$i^* = 3.5132(\theta_d - \theta) - 0.1006\dot{\theta} + 67.5659\int_0^t (\theta_d - \theta(r))dr - 1.8241\theta_d. \tag{11.43}$$

One last observation. The reader can verify that the controller in (11.38) is identical to the controller in (11.25) if (11.26) is satisfied and the following is established:

$$k_1 = \frac{f}{k}, \quad \sigma = \zeta\omega_n, \quad \sigma^2 + \omega_d^2 = \omega_n^2.$$

It is up to the reader to decide which of the approaches presented in Sects. 11.3.1 or 11.3.2 is preferred.

Some simulation results are presented in Fig. 11.10 when using the controller in (11.43) and the motor parameters $k = 675.4471$, $a = 2.8681$. There, a step change in the desired position is commanded and $t_r = 0.068$[s], $M_p(\%) = 19.6\%$, are measured, which are almost identical to the values specified above, i.e., $t_r = 0.068$[s] and $M_p(\%) = 20\%$. A constant disturbance is also considered $D(s) = -T_p(s)/(Jk) = -0.5/s$. This disturbance is introduced via software as a signal that adds to i^*, given in (11.43), from $t = 0.7$[s] on. Note that the steady-state error is zero. On the other hand, a very good response is observed when the external disturbance appears.

Some other simulation results are shown in Fig. 11.11 when using the controller (11.43). In this case, the following desired position is commanded:

$$\theta_d = \begin{cases} 1.5, & 0 \leq t < 1.5 \\ 3, & 1.5 \leq t < 4 \\ 1.5, & t \geq 4 \end{cases}. \tag{11.44}$$

The following external disturbance is also applied (via software):

$$d(t) = \begin{cases} 0, & 0 \leq t < 0.7 \\ 0.5, & 0.7 \leq t < 2.55 \\ 0, & 2.55 \leq t < 3.25 \\ 0.5, & 3.25 \leq t < 5.1 \\ 0, & 5.1 \leq t < 5.8 \\ 0.5, & t \geq 5.8 \end{cases}. \tag{11.45}$$

Comparing Figs. 11.10 and 11.11, it is clear that the desired transient and steady-state response specifications are satisfied. Moreover, note that the step changes in the reference commanded at $t = 1.5$[s] and $t = 4$[s] occur when the disturbance $d(t)$ is present and, despite this, the transient response specifications are still satisfied. Also note that the deviations due to the disturbances vanish very fast.

Some experimental results are presented in Fig. 11.12 under the same conditions as in Fig. 11.10. The following values are measured $t_r = 0.068$[s] and $M_p(\%) = 19.3\%$. Note that these experimental results are very similar to the simulation results in Fig. 11.10. It is stressed that the very good response observed when an external disturbance appears is thanks to $f = 40$, and that such a large value is possible because of the low noise content in the measured position θ.

Some other experimental results are shown in Fig. 11.13 that were performed under the same conditions as the simulations in Fig. 11.11. Aside from the measured position (continuous line) the response obtained in simulation (dashed line) is also shown in Fig. 11.13. The simulation response is obtained with the second-order system without disturbances $\ddot{\theta} + 30.8018\dot{\theta} + 1140.9\theta = 1140.9\theta_d$, which satisfies the design specifications $t_r = 0.068$[s] and $M_p(\%) = 20\%$, with poles located at $s = -15.4009 \pm j30.0623$. It is observed that both responses are almost identical and they cannot be distinguished from each other in Fig. 11.13. This shows that the desired transient and steady-state response specifications are satisfied in the experiment. These experimental results corroborate the observations stated just before (11.25).

The simulations in Figs. 11.10 and 11.11 were performed using the MATLAB/Simulink diagram shown in Fig. 11.14 and the following steps: (1) Execute the following MATLAB code in an m-file:

```
a=2.8681;
k=675.4471;
f=40;
Mp=20; %
tr=0.068; % sec
```

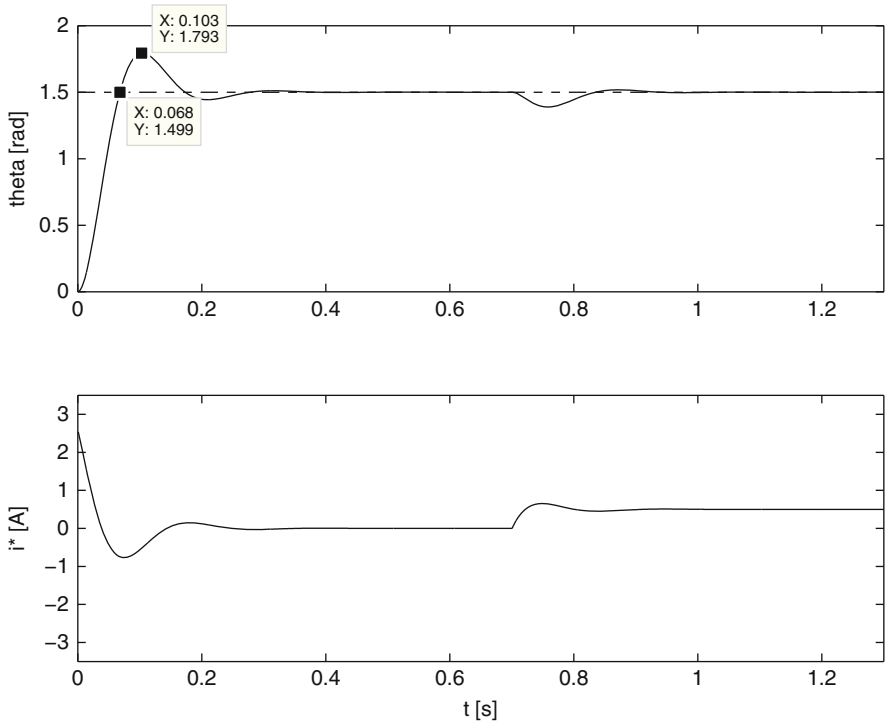

Fig. 11.10 Position control using a two-degrees-of-freedom controller. Simulation results (θ_d = 1.5[rad])

```
z=sqrt( log(Mp/100)^2/( log(Mp/100)^2 + pi^2) );
wn=1/(tr*sqrt(1-z^2))*(pi-atan(sqrt(1-z^2)/z));
sigma=z*wn;
wd=wn*sqrt(1-z^2);
kp1=(sigma^2+wd^2)/k;
ki1=(sigma^2+wd^2)*f/k;
kp2=2*sigma*f/k;
kd2=(2*sigma+f-a)/k;
kp=kp1+kp2
kd=kd2
ki=ki1
D=0.5;
```

(2) Run the simulation in Fig. 11.14. (3) Run the following MATLAB code in an m-file:

```
Ts=0.001;
t=0:Ts:6.5;
figure(1)
```

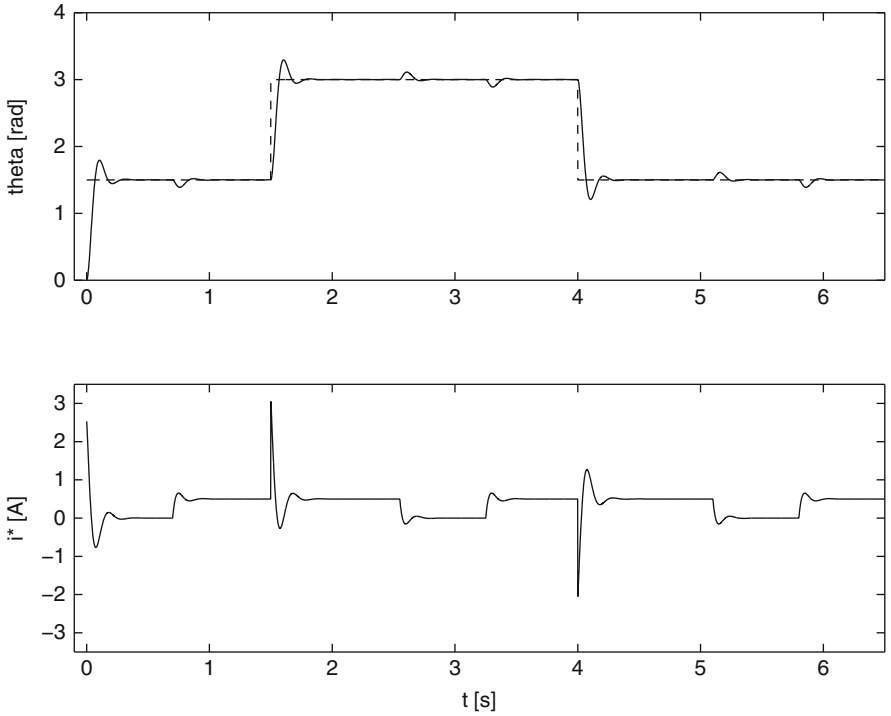

Fig. 11.11 Position control with a two-degrees-of-freedom controller. Simulation results

```
subplot(2,1,1)
plot(t,Datos(:,1),'b--');
hold on
plot(t,Datos(:,2),'b-');
hold off
axis([-0.1 6.5 0 4])
ylabel('theta [rad]')
subplot(2,1,2)
plot(t,Datos(:,3),'b-');
axis([-0.1 6.5 -3.5 3.5])
xlabel('t [s]')
ylabel('i* [A]')

figure(2)
subplot(2,1,1)
plot(t,Datos(:,1),'b--');
hold on
plot(t,Datos(:,2),'b-');
hold off
```

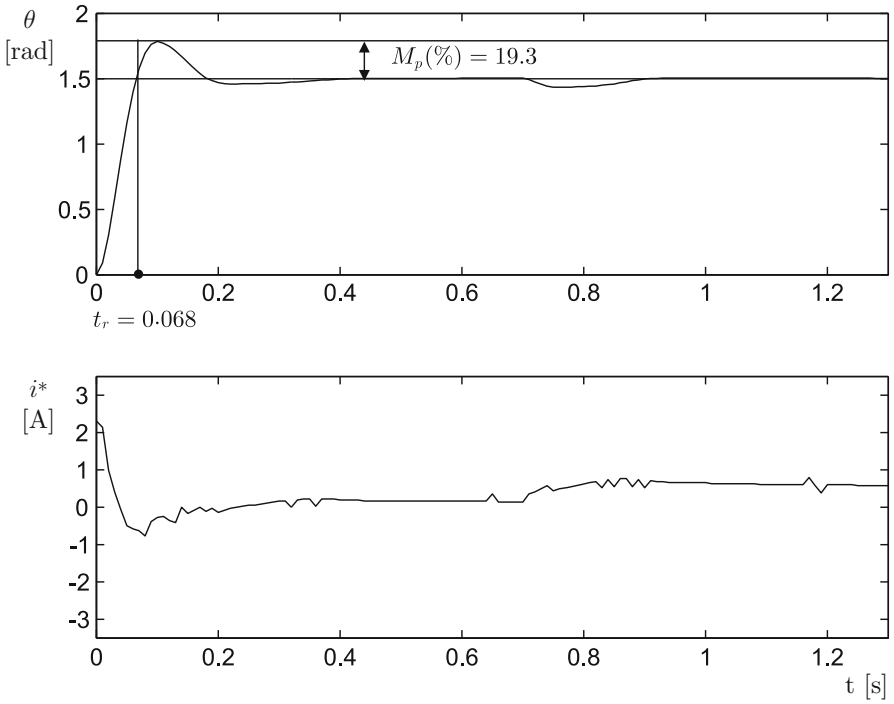

Fig. 11.12 Position control using a two-degrees-of-freedom controller. Experimental results ($\theta_d = 1.5$[rad])

```
axis([0 1.3 0 2])
ylabel('theta [rad]')
subplot(2,1,2)
plot(t,Datos(:,3),'b-');
axis([0 1.3 -3.5 3.5])
xlabel('t [s]')
ylabel('i* [A]')
```

The blocks in the MATLAB/Simulink diagram shown in Fig. 11.14 were programmed as follows. Three theta step blocks and five step D blocks were suitably programmed to generate the desired position in (11.44) and the disturbance in (11.45). The To workspace block saves data as an array with 0.001 as the sample time.

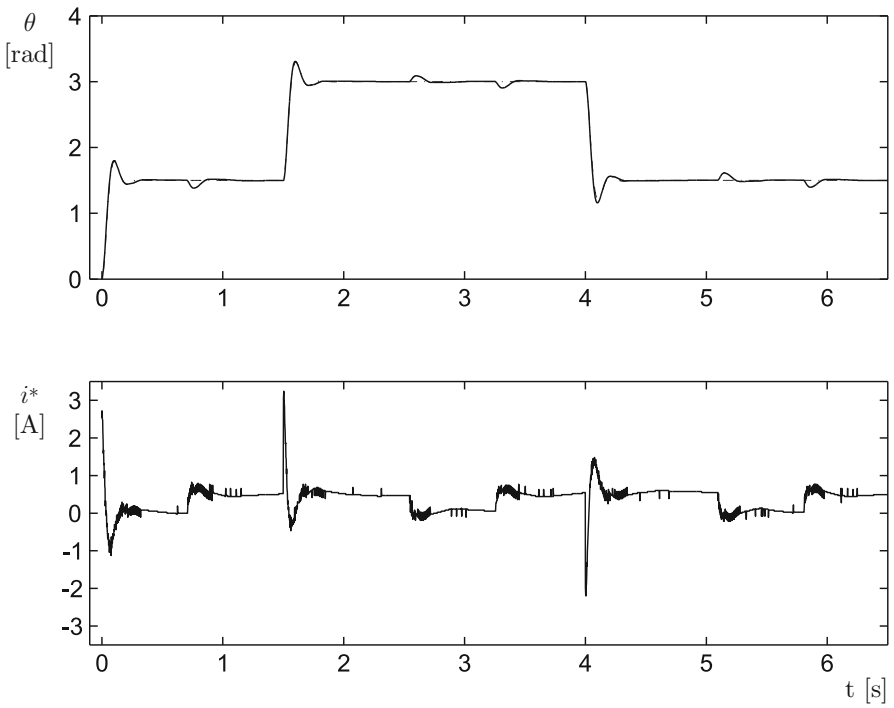

Fig. 11.13 Position control with a two-degrees-of-freedom controller. Experimental results

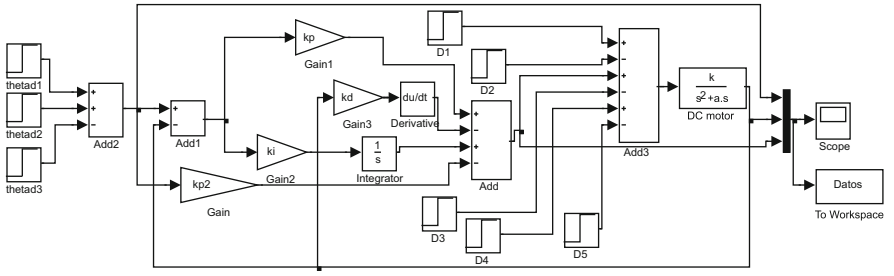

Fig. 11.14 MATLAB/Simulink diagram used for simulations in Figs. 11.10 and 11.11

11.3.3 A Classical PID Controller

To appreciate the advantages of the controller designed in Sects. 11.3.1 and 11.3.2, some experimental results are presented in Figs. 11.15 and 11.16, when using the following classical PID controller:

$$i^* = k_p(\theta_d - \theta) + k_d \frac{d}{dt}(\theta_d - \theta) + k_i \int_0^t (\theta_d - \theta(r))dr.$$

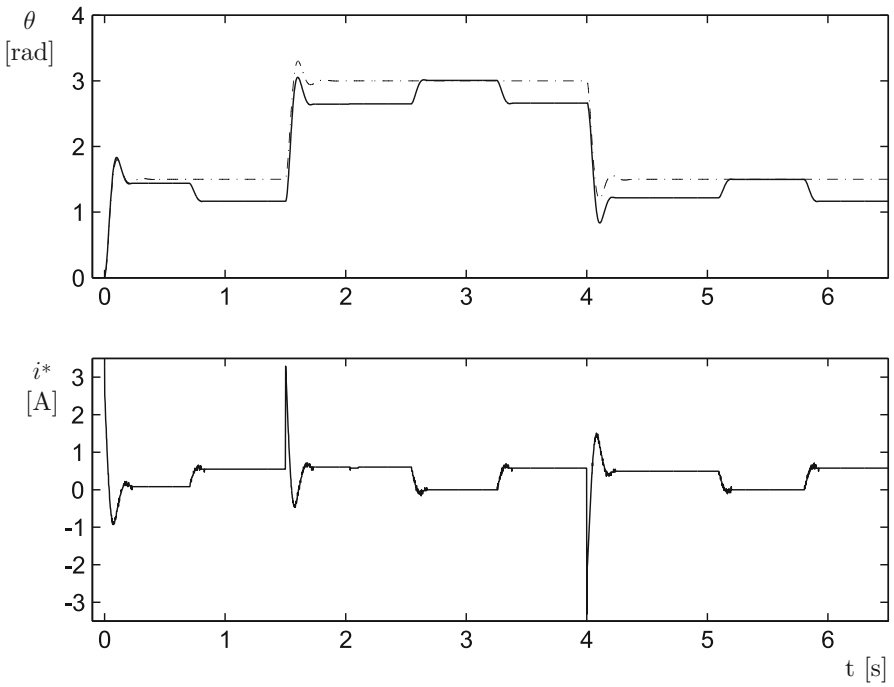

Fig. 11.15 Classical proportional–integral–derivative (PID) position control ($\alpha = 0.01$). Experimental results

The desired position and the external disturbance are the same used in Fig. 11.13, i.e., those defined in (11.44) and (11.45). The measured position (continuous line) is presented in addition to the response obtained in the simulation (dashed line). The simulation response is, again, obtained from the system without disturbances $\ddot{\theta} + 30.8018\dot{\theta} + 1140.9\theta = 1140.9\theta_d$, which achieves $t_r = 0.068$[s] and $M_p(\%) = 20\%$, with poles located at $s = -15.4009 \pm j30.0623$.

The classical PID controller design is performed according to Sect. 5.2.5. Using the plant parameters:

$$k = 675.4471, \quad a = 2.8681,$$

the desired closed-loop poles located at:

$$s = -15.4009 \pm j30.0623,$$

to achieve 0.068[s] as the rise time and 20% overshoot, the design parameter $\alpha = 0.01$, and (5.30), (5.32), the following controller gains are obtained:

$$k_d = 0.0414, \quad k_p = 1.6896, \quad k_i = 0.0169.$$

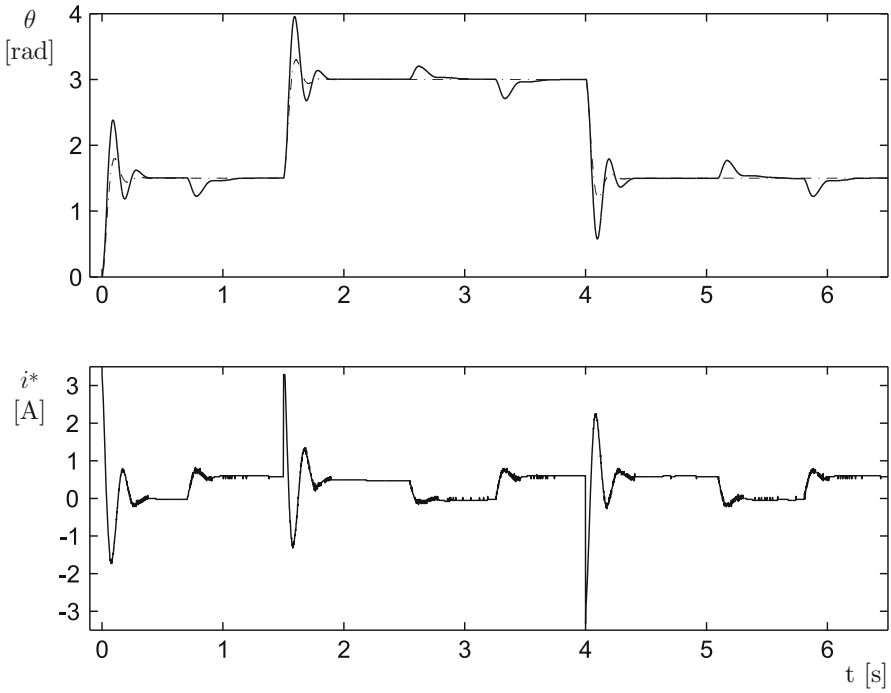

Fig. 11.16 Classical PID position control ($\alpha = 10$). Experimental results

It is important to stress that, according to the design criterion presented in Sect. 5.2.5, $\alpha = 0.01$ is chosen very close to zero to ensure that the transient response satisfies the desired specifications. As observed in Fig. 11.15, this is achieved for the first step change in the reference, i.e., before a disturbance appears. However, problems arise when an external disturbance is applied:

The use of a small α results in a very small integral gain, which produces such slow integral adjustments that the disturbance effect is not compensated for in Fig. 11.15. Moreover, although the measured position reaches the desired position at $t \approx 3$[s] and $t \approx 5.5$[s], this is because the external disturbance is not present at those points in time.

As explained in Sect. 5.2.5, larger integral gains can be obtained if α is chosen to be large. Thus, $\alpha = 10$ is proposed, which results in the following controller gains:

$$k_d = 0.0414, \quad k_p = 2.1027, \quad k_i = 16.8915.$$

The corresponding experimental results are shown in Fig. 11.16. Note that the deviation due to the disturbance varnishes very fast. However, the price to pay for this improvement is that the response to a step change in the desired position does not satisfy the desired specifications. This is the main drawback of classical PID control: there is no tuning rule at the disposal of the designer to compute controller

gains that simultaneously achieve the desired transient response specifications to a desired reference change and a satisfactory disturbance rejection. On the contrary, this problem is solved by the controllers designed in Sects. 11.3.1 and 11.3.2 whose experimental results are shown in Fig. 11.13.

11.4 Trajectory Tracking

In this section, a controller is designed that allows the motor position to reach a time-varying desired position, i.e., $\theta_d(t)$ is no longer constant, but changes over time. It is assumed that no external disturbance exists, i.e., $T_p(s) = 0$, and the first and second time derivatives of $\theta_d(t)$ can be computed, i.e., $\dot{\theta}_d(t)$ and $\ddot{\theta}_d(t)$ are known and they are continuous. Consider the motor model in (11.1) once the inverse Laplace transform is applied with $T_p(s) = 0$, i.e.,

$$\ddot{\theta} + a\dot{\theta} = ki^*. \tag{11.46}$$

Also consider the following controller, where θ_d, $\dot{\theta}_d$ and $\ddot{\theta}_d$ are known:

$$i^*(t) = \frac{1}{k}\left[\frac{d^2\theta_d(t)}{dt^2} - k_d\left(\frac{d\theta(t)}{dt} - \frac{d\theta_d(t)}{dt}\right) - k_p\left(\theta(t) - \theta_d(t)\right) + a\frac{d\theta(t)}{dt}\right]. \tag{11.47}$$

Note that $\frac{d\theta(t)}{dt}$ stands for the measured motor velocity. Replacing (11.47) in (11.46), the following is found:

$$\frac{d^2\theta(t)}{dt^2} - \frac{d^2\theta_d(t)}{dt^2} + k_d\left(\frac{d\theta(t)}{dt} - \frac{d\theta_d(t)}{dt}\right) + k_p\left(\theta(t) - \theta_d(t)\right) = 0. \tag{11.48}$$

Define the tracking error and its Laplace transform as $\varepsilon(t) = \theta(t) - \theta_d(t)$ and $E(s)$ respectively. Then, (11.48) can be written as:

$$s^2E(s) - s\varepsilon(0) - \frac{d\varepsilon(0)}{dt} + k_d sE(s) - k_d\varepsilon(0) + k_pE(s) = 0, \tag{11.49}$$

and finally:

$$E(s) = \frac{(k_d + s)\varepsilon(0) + \frac{d\varepsilon(0)}{dt}}{s^2 + k_d s + k_p}. \tag{11.50}$$

According to Chap. 3, the solution of (11.50) has the form:

$$\varepsilon(t) = \varepsilon_n(t) + \varepsilon_f(t)$$

where the forced response is zero $\varepsilon_f(t) = 0$ because the differential equation in (11.49) is not excited. Hence, $\varepsilon(t) = \varepsilon_n(t)$, which tends toward zero as time grows if the roots of the characteristic polynomial $s^2 + k_d s + k_p$ have a negative real part. This situation is desirable because if $\varepsilon(t)$ tends toward zero, then $\theta(t)$ tends toward $\theta_d(t)$ as time grows. On the other hand, as the characteristic polynomial $s^2 + k_d s + k_p$ is second-degree, according to Sect. 4.2, its two roots have a negative real part if, and only if, all the coefficients of this polynomial have the same sign, i.e., if, and only if, $k_p > 0$ and $k_d > 0$. This is the criterion for selecting the gains for this controller. However, it is important to stress that the controller gains achieving a good performance in practice must be selected within this set of gains by trial and error. Some experimental results are shown in Fig. 11.17 when using a motor with $k = 25.26$, $a = 3.5201$, together with the controller in (11.47), the controller gains:

$$k_p = 363.88, \quad k_d = 22,$$

and the desired trajectory:

$$\theta_d(t) = 0.8 \sin(5t) + 0.5 \cos(3t) + 3.0. \tag{11.51}$$

Note that θ reaches the desired position $\theta_d(t)$ despite these variables having different initial values. It is important to state that, as a mathematical expression is known for θ_d, which is given in (11.51), the corresponding expressions for $\dot{\theta}_d$ and $\ddot{\theta}_d$ are computed analytically by differentiation. This avoids problems due to noise amplification.

11.5 Prototype Construction

The electrical diagram of the complete control system is presented in Fig. 11.18. The main components are the following:

- PM brushed DC motor. Nominal voltage 20[V], nominal current 3[A]. Provided with an optical encoder with 400 ppr, which requires +5 DC voltage.
- A portable computer with USB port.
- USB-series adapter.
- Microchip PIC16F877A Microcontroller.
- DAC0800LCN Digital/analog converter.
- TL084 quadruple operational amplifier.
- TL081 operational amplifier.
- MAX232 driver.
- TIP141 and TIP145 transistors.

Microcontroller PIC16F877A computes the control algorithm. Once the desired position θ_d and the measured position θ are known, the microcontroller computes the control signal i^*. This value is assigned to the variable "iast" according to program listed in Sect. 11.6.

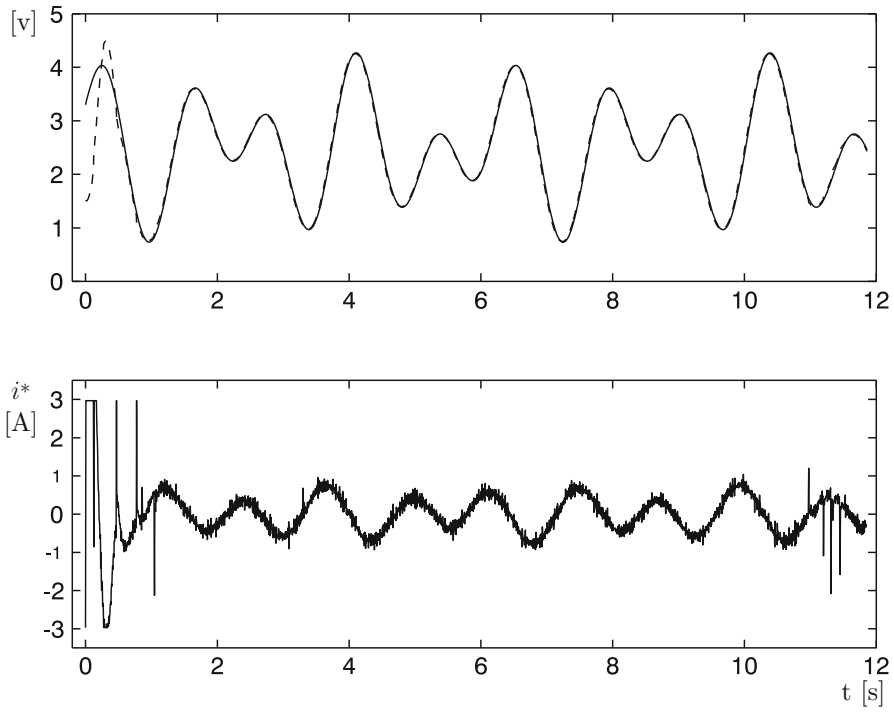

Fig. 11.17 Trajectory tracking. Motor measured position θ is represented by the dashed line. The continuous line represents $\theta_d(t)$

As indicated in Fig. 11.18, $-i^*$ must appear at the input of the operational amplifier TL081 inside the block labeled "current control." This is accomplished as follows. First, the program listed in Sect. 11.6 assigns iast=-iast to change the sign of i^*. The microcontroller must deliver an 8-bit digital code to the digital/analog converter DAC0800LCN. With this aim, the microcontroller computes:

$$\text{iastd} = 36.4286 * \text{iast} + 127,$$

which corresponds to the relation shown in Fig. 11.19. The digital/analog converter works together with an operational amplifier TL081. These devices are connected according to the manufacturer's specifications [1]. This ensures that, at the operational amplifier output, a voltage appears whose numerical value corresponds to $-i^*$. This value is received by an operational amplifier TL081, which evaluates the electric current controller [6, 7]:

$$u_i = 100(i^* - i).$$

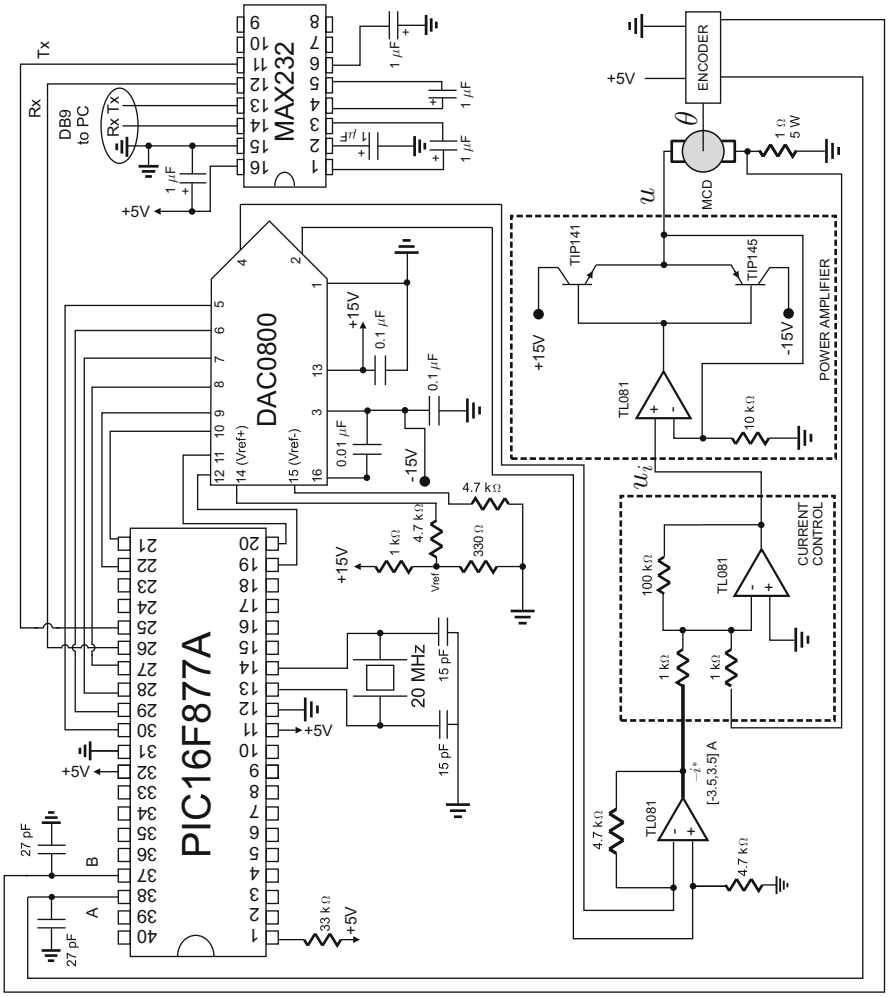

Fig. 11.18 Electrical diagram of the complete position control system based on a PIC16F877A microcontroller

Finally, a power amplifier with a unit gain is implemented with a third operational amplifier TL081 and two power transistors TIP141 and TIP145 in a complementary symmetry connection.

The driver MAX232 sends the measured position θ and the control signal i^* to a portable computer (through an USB series adapter) whose unique task is to plot these variables. In the following, it is described how the microcontroller PIC16F877A performs the controller task. It is stressed that the following exposition is not intended to present deep explanations of how the microcontroller works. We are merely presenting a general description of the microcontroller components involved in the specific task to be performed. The program employed by the microcontroller is listed in the next section.

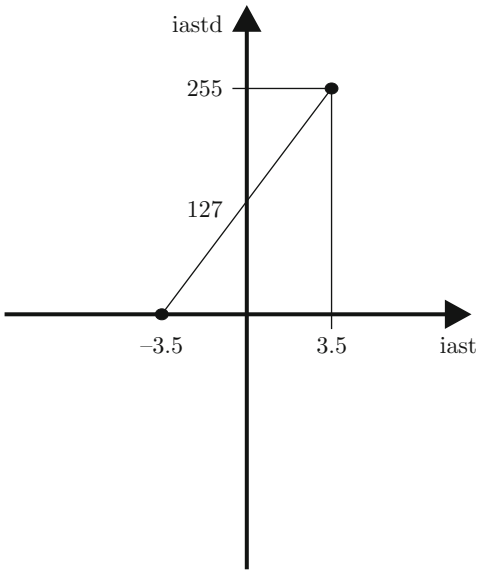

Fig. 11.19 Conditioning of signal i^* that must be sent to the digital/analog converter

The microcontroller PIC16F877A [2] has a maximal clock frequency of 20 MHz, it has five input–output configurable ports and three timers. Timer TMR0 is employed in this application to fix the sampling period: $T = 0.001$[s] (20 counts of timer 0) for experiments in Sects. 11.1, 11.2.1, and 11.2.2, whereas $T = 0.002$[seg] (40 counts of timer 0) for the experiment in Sect. 11.3.2.

The motor position θ is measured from the encoder using a subroutine and it is stored in the 16-bit variable "cuenta2." The position in radians is obtained by computing pos=cuenta2*0.0039 where 0.0039=3.1416 rad/(2*400 cuentas). This is because the encoder has 400 ppr, but 4×400 counts per revolution are really obtained (each 2π radians), i.e., 2×400 counts each π radians. Once the control signal "iast" is computed, it is delivered to the digital/analog converter through port D (PORTD=iastd). Finally, either the motor position (through the variable "cuenta") or the control signal (through the variable "iastd") is sent to the portable computer through the instructions: putc(0xAA); putc(cuentaH); putc(cuentaL). The electric current control and the power amplifier are implemented identically as described in Sects. 10.6.1 and 10.6.2 respectively.

11.6 Microcontroller PIC16F877A Programming

The flow diagram for the programming of microcontroller PIC16F877A when controlling the position in a PM brushed DC motor is shown in Fig. 11.20. The complete code is listed below. The reader is referred to [3] for a complete explanation of each of the instructions appearing in the following program.

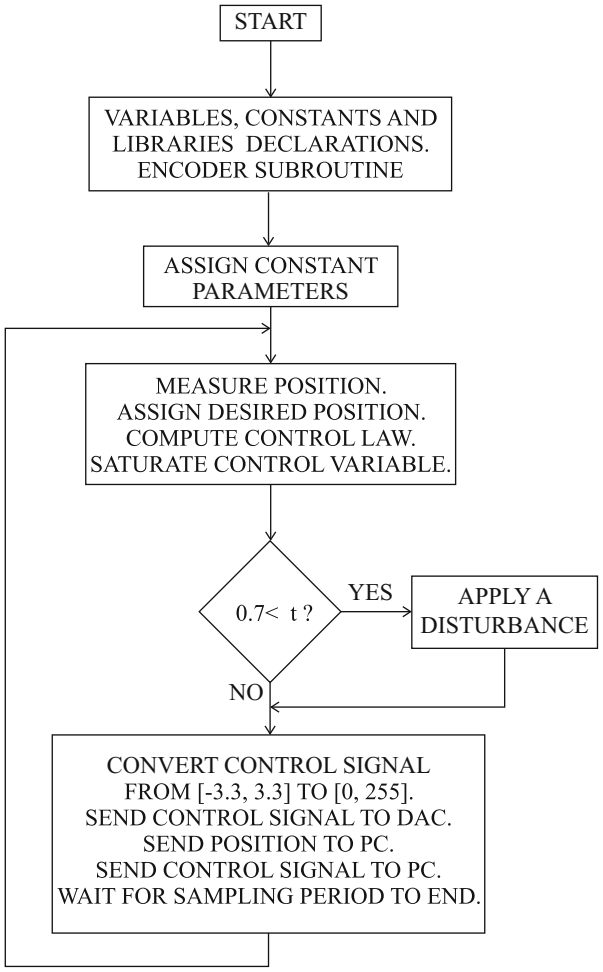

Fig. 11.20 Flow diagram for the programming of microcontroller PIC16F877A when controlling the position in a permanent magnet (PM) brushed DC motor

```
// PM brushed DC motor position control program
#include<16f877A.h>
#device adc=10 //ADC, 10 bits
#include<stdlib.h>
#include<math.h>

#fuses HS,NOWDT,PUT,NOBROWNOUT,NOLVP,NOWRT,NOPROTECT,NOCPD
#usedelay(clock=20000000) //time basis

#use
rs232(baud=115200,XMIT=PIN_C6,RCV=PIN_C7,BITS=8,PARITY=N)
//Config.
```

```
//serial port
//ports and registers addresses
#byte OPTION= 0x81
#byte TMR0 = 0x01
#byte PORTA = 0x05
#byte PORTB = 0x06
#byte PORTC = 0x07
#byte PORTD = 0x08
#byte PORTE = 0x09
#bit PC0 = 0x07.0
#bit PC1 = 0x07.1
//------------Variables declaration------------//
int16 inter,cuenta;
int8 cuentaH,cuentaL,puerto,AB,AB_1,aux,cont,iastd,cont2,
cont3,cont4;
float pos,error,iast,cuenta2,posm1,kp,kv,c,delta,v,kp1,ki1,
kp2,kd2,ie;
unsigned int u;
//int1 ban;
//------------Encoder interrupt subroutine------------//
#int_rb void rb_isr() {
puerto=PORTB;
AB=((puerto)&(0x30))>>4;
aux=AB^AB_1;
if(aux!=0)
if(aux!=3)
if(((AB_1<<1)^AB)&(0x02))
cuenta--;
else
cuenta++;
AB_1=AB;
}
//------------Main program------------//
void main(void) {
setup_adc(ADC_CLOCK_INTERNAL ); //ADC (internal clock)
set_tris_a(0b11111111);
set_tris_b(0b11111111);
set_tris_c(0b10000000); //serial communication
set_tris_d(0b00000000);
set_tris_e(0b11111111);
OPTION=0x07; //pre_scaler timer0, 1:256
PORTC=0;
TMR0=0;
cuenta=0;
AB=0;
AB_1=0;
enable_interrupts(global);
enable_interrupts(int_rb);
cont=4;
cont2=0;
cont3=0;
cont4=0;
PORTD=127; // iast=0 is set
kp=1.6891;
```

```
kv=0.0414;
c=30.8018;
delta=1.6891;
v=0.0;
kp1=1.6891;
ki1=67.5659;
kp2=1.8241;
kd2=0.1006;
posm1=0.0;
ie=0.0;
while(cont2<3) // a 10-s waiting period is introduced
{
while(cont3<255)
{
TMR0=0;
while(TMR0<255)
{
}
cont3++;
}
cont2++;
}
TMR0=0;
while(TRUE) // control loop
{
cont++;
cuenta2=(signed int16)cuenta;
pos=cuenta2*0.0039; // 3.1416 rad/(2*400 cuentas)
error=1.5-pos;
/* Identification */
// iast=0.5*error;
/* _____________________________________________ */

/* Proportional control plus velocity feedback */
// iast=kp*error-kv*(pos-posm1)/0.001;
/* _____________________________________________ */

/* Lead compensator */
// iast=delta*(error+v);
// v=(-c*v+(2.8-c)*error)*0.001+v;
/* _____________________________________________ */

/* 2-degrees-of-freedom controller */
iast=kp1*error+ki1*ie-kp2*pos-kd2*(pos-posm1)/0.002;
ie=ie+0.002*error;
/* _____________________________________________ */

if(iast<-3.3) // iast is constrained to [-3.3,+3.3] amperes
iast=-3.3; // (DAC delivers within the range [-3.5,+3.5])
if(iast>3.3)
iast=3.3;
iast=-iast; //-i*, TL081 at ''current control''
if(cont4>70) //time>0.7 s, disturbance (-0.5)
iast=iast+0.5;
```

```
iastd=(unsigned int)(36.4286*iast+127);
PORTD=iastd;
posm1=pos;
/* // uncomment if it is desired to send position to computer
// every 10 ms
if(cont==5)
{
cont=0;
inter=(cuenta)&(0xFF00);
cuentaH=inter>>8;
cuentaL=(cuenta)&(0x00FF);
putc(0xAA); //recognition serial port
putc(cuentaH); //sending to serial port
putc(cuentaL);
cont4++;
if(cont4==255)
cont4=0;
}
*/
/* // uncomment if it is desired to send control signal to
// computer every 10 ms
iast=-iast; // plot +i*
if(cont4>70) // plot control signal without disturbance
iast=iast+0.5;
iastd=(unsigned int)(36.4286*iast+127);
if(cont==5)
{
cont=0;
inter=(cuenta)&(0xFF00);
cuentaH=0x00; //inter>>8;
cuentaL=(iastd)&(0xFF);
putc(0xAA); //recognition serial port
putc(cuentaH); //sending to serial port
putc(cuentaL);
cont4++;
if(cont4==255)
cont4=0;
}
*/
PC1=1; //waiting for sampling time
while(TMR0<40) //each cuenta=(4/FXtal)*256 s,
// (40=2 ms, sampling period)
{
}
PC1=0; //sampling time has arrived
TMR0=0;
} //closing the infinite while
} //closing main
```

11.7 Personal Computer-Based Controller Implementation

The controller presented in Sect. 11.4, for trajectory tracking, was experimentally implemented using a personal computer and a data acquisition board. The reason for this is that the sine and cosine trigonometric functions involved in the controller are time-consuming for the microcontroller PIC16F877A. This problem disappears when a personal computer is employed. The following are the main control system components.

- Personal computer Pentium II, 233 MHz
- Data acquisition board Advantech PCL-812PG [4]. Fifteen analog/digital input channels with a unique 12-bit successive approximations analog/digital converter, with an analog range $[-5, +5]$[V]. Two digital/analog output channels, each based on a 12-bit digital/analog converter with analog range $[0, +5]$[V]. It is also provided with two programmable timers employed to fix the sampling period. In Fig. 11.17, the sampling period was set to de 0.005[s].
- PM brushed DC motor. Nominal voltage 24[V], nominal current 2.3[A].
- Position sensor. 1[KOhm] Precision potentiometer, 10 turns.

The block diagram of the complete control system is shown in Fig. 11.21.

11.7.1 Input Interface

The potentiometer used as a position sensor, Fig. 11.21, delivers a voltage θ in the range $[0, +5]$[V]. This is very convenient as the analog/digital converter has an analog range $[-5, +5]$[V]. From the voltage at the potentiometer, the analog/digital converter delivers a code, "code$_i$," within the range 0 to $2^n = 4096$ because it is a 12-bit converter. From this code, the variable u_a is retrieved, which must be numerically identical to the voltage at the potentiometer θ. According to Fig. 11.22, this can be performed by computing:

$$u_a = \frac{10.0}{4096.0}\,\text{code}_i - 5.0.$$

This must be performed by the computer.

11.7.2 Output Interface

According to (11.1) the control signal is the current command $-i^*$, which, depending on the motor specifications, may take values in the range -3[A] to +3[A]. $-i^*$ is sent by the computer to the data acquisition board, which must deliver $-i^*$ as a voltage signal.

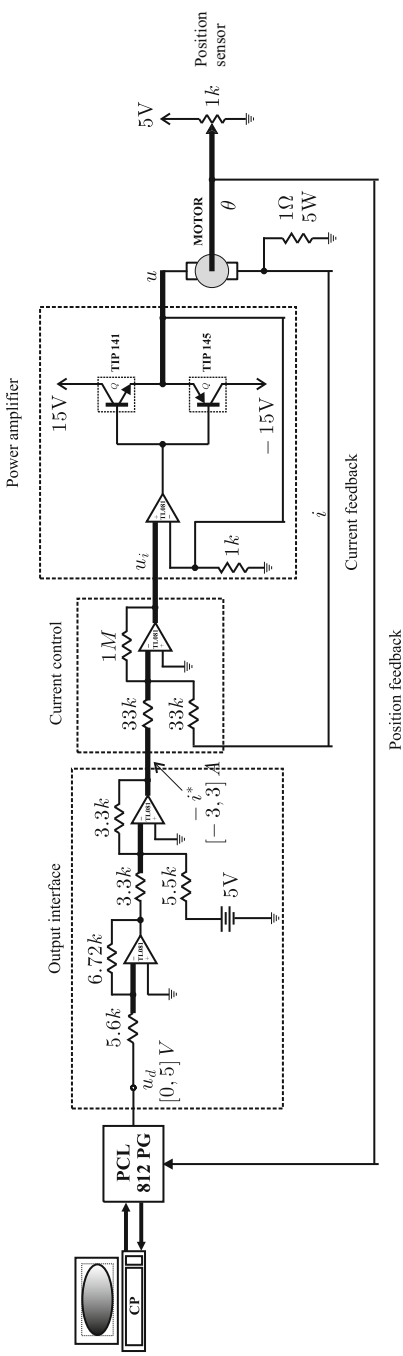

Fig. 11.21 Block diagram and interfaces of the personal computer-based position control system

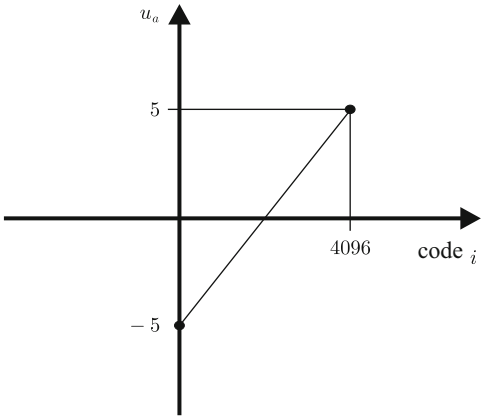

Fig. 11.22 Conditioning of the position signal delivered by the analog/digital converter

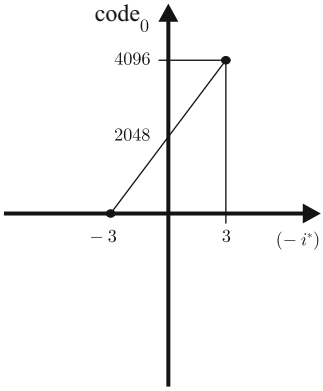

Fig. 11.23 Conditioning of the signal $-i^*$, which is to be sent to the digital/analog converter

Once the computer obtains $-i^*$, as a result of the corresponding control algorithm evaluation, a set of instructions constrains $-i^*$ to the range $[-3, +3]$[V]. After that, taking into account that the digital/analog converter is 12-bit, Fig. 11.23 is drawn to realize that the following datum:

$$\text{code}_o = \frac{4096}{6}(-i^*) + 2048,$$

has to be sent to the digital/analog converter. Hence, the digital/analog converter delivers a voltage u_d in the range 0[V] to +5[V]. Thus, an analog electronic circuit must be employed to convert this voltage in the range 0[V] to +5[V] into the range -3[V] to +3[V], which corresponds to the values taken by $-i^*$. According to Fig. 11.24 this is accomplished by an operational amplifier-based analog circuit performing the following mathematical operation:

$$-i^* = \frac{6}{5}u_d - 3.$$

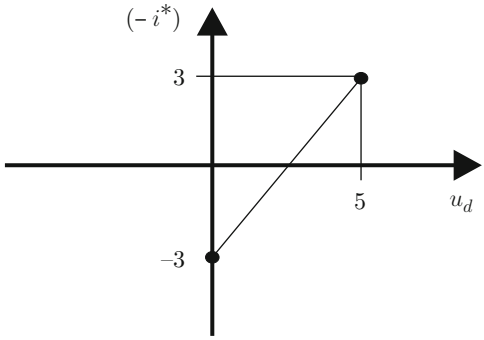

Fig. 11.24 Design of the analog interface that must be placed at the output of the digital/analog converter

This function is performed by the block labeled "output interface" in Fig. 11.21.[1]

The operational amplifier shown in the block labeled "current control" in Fig. 11.21 performs the mathematical operation indicated in (10.14) with $K = 30$. The power amplifier is identical to that presented in Sect. 10.6.2.

11.7.3 Computer Program

The flow diagram for the PC programming when controlling the position in a PM brushed DC motor is presented in Fig. 11.25. All the control algorithms were programmed in Borland C++. The following is the code that was employed.

```
#include <stdio.h>
#include <conio.h>
#include <math.h>

#include <stdlib.h>
#include <string.h>
#include <iostream.h>

int alto,bajo,dato,salida,i;

volatile float Ts=0.005;//sampling period

/*------------------encoder variables----------*/
//volatile int alto_dig=0,bajo_dig=0,dato_dig=0;
volatile float y,ir,ym1,v,dv,nv,e,ninte;

volatile float inte,gc1,gc2,tiempo,perturbacion;
/*-------------------------------------------------*/
```

[1] All operational amplifiers used in Fig. 11.21 are TL081. The first three amplifiers have their terminals "+" to ground whereas the operational amplifier connected to the transistor bases has its terminal "+" connected to u_i.

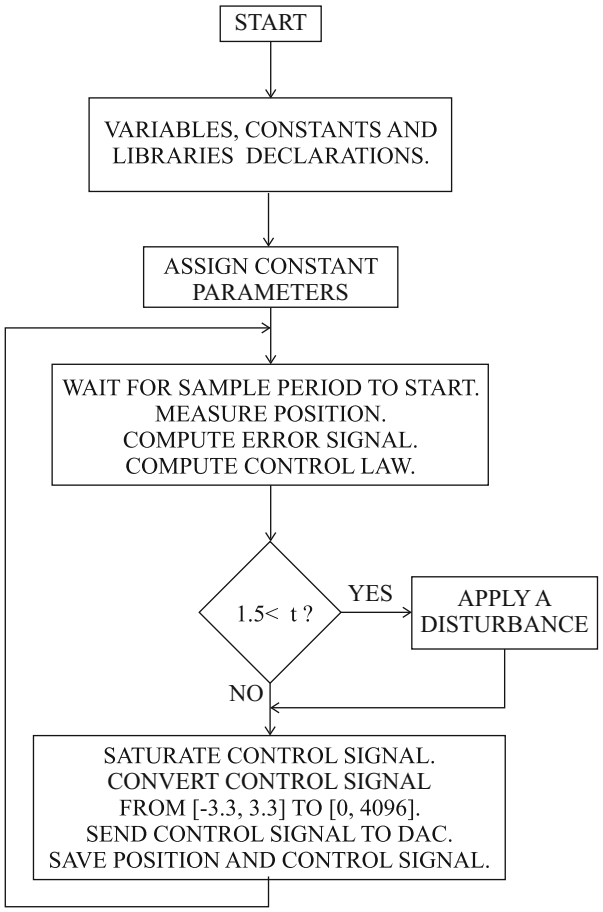

Fig. 11.25 Flow diagram for the PC programming when the controlling the position in a PM brushed DC motor

```
volatile float r_des=1.0,Kp=1.9434,Kv=0.1777;
//volatile float gama=2.3853,c=12.2734,b=4.8935;
volatile float gama=2.0465,c=10.3672,b=3.8935;

volatile float kp1=1.9434,ki1=9.7170;

volatile float kp2=1.3878,kd2=0.3195;
volatile float k=25.26,a=3.5201;
volatile float kp=363.88,kd=22;
volatile float ddthetad,dthetad,thetad,dy;
volatile long double y0;
/*-Base address and sampling period for acquisition board--*/
```

```
int base=0x210;//Acquisition board base address

int lsb1=0x00;//sampling period

int msb1=0x01;//sampling period

int lsb2=0x00;//sampling period

int msb2=0x01;//sampling period

volatile float yy[3000];
volatile float uu[3000];

volatile float ref[3000];
FILE *fp;
char Direccion[30];

void Inicializa(void);

void EscribeArchivo(float datos1,floatdatos2,float datos3);

void Cierra(void); void main()
{
clrscr();

/*---------Acquisition board programming--------*/

/*---See PCL 812-PG Advantech manual------*/

outportb(base+11,6);
//CAD starts by clock and datum software transfer
outportb(base+9,0);// gain 1

outportb(base+10,15);// Channel AD No. 15
outportb(base+3,0x77);//Programming sampling period
outportb(base+1,lsb1);//Programming sampling period
outportb(base+1,msb1);//Programming sampling period
outportb(base+3,0xb7);//Programming sampling period
outportb(base+2,lsb2);//Programming sampling period
outportb(base+2,msb2);//Programming sampling period
i=1;
do
// control cycle begins
{
/*-------------Conversion AD----------------------------*/
do
{
} while (inportb(base+5)>15);
alto=inportb(base+5);
bajo=inportb(base+4);
dato=256*alto+bajo;
y=(10.0/4096.0)*dato-5.0;//Position theta
tiempo=(i-1)*Ts;
/*--------encoder reading begins ---------------------*/
```

```
// for the digital inputs configuration
/*
alto_dig=inport(base+7);
alto_dig+=0xff+1;
bajo_dig=inport(base+6);
bajo_dig=(bajo_dig)&(0x00ff);
dato_dig=256*alto_dig+bajo_dig;
po_rad=((6.283185307*dato_dig)/1600);
// 1600 is no. of pulses per revolution.
// gotoxy(10,10); printf("Posicion en radianes: %f",po_rad);
y=po_rad;
*/
/*-------encoder reading ends-------------------------*/
if(i==1)
{ //initial conditions
y0=y;
ym1=0.0;
v=0.0;
inte=0.0;
}
y=y-y0;//output is fixed to zero
e=r_des-y;
/*------ Proportional control plus velocity feedback------*/
/*
ir=Kp*(r_des-y)-Kv*(y-ym1)/Ts;
*/
/*----------- Lead compensator --------------*/
/*
dv=(-c*v+(b-c)*e )*Ts;
nv=v+dv;
ir=gama*(e+v);
*/
/*---------- 2-degrees-of-freedom controller----------*/
/* gc1=kp1*e+inte;
gc2=kp2*y+kd2*(y-ym1)/Ts;
ir=gc1-gc2;
ninte=inte+ki1*e*Ts; */
/*---------- Trajectory tracking controller----------*/
dy=(y-ym1)/Ts;
thetad=0.8*sin(5.0*tiempo)+0.5*cos(3.0*tiempo)+3.0;
dthetad=4.0*cos(5.0*tiempo)-1.5*sin(3.0*tiempo);
ddthetad=(-1)*20.0*sin(5.0*tiempo)-4.5*cos(3.0*tiempo);
ir=1/k*(ddthetad-kd*(dy-dthetad)-kp*(y-thetad)+a*dy);
perturbacion=0.0;
if(tiempo>1.5)
{perturbacion=0.5;}
ir=-ir+perturbacion;
// Saturation of ir [-3,3][A]
if(ir>2.9 )
{ ir=2.9; }
if(ir<-2.9)
{ir=-2.9;}
salida=floor(682.66*ir+2048.0);
bajo=salida & 0xff;
```

```
alto=salida & 0x0f00;
alto=floor(alto/256.0);
outportb(base+4,bajo); //send code_o to CDA
outportb(base+5,alto);
yy[i]=y;
uu[i]=ir-perturbacion;
ref[i]=r_des;
ym1=y;
v=nv;
inte=ninte;
i++;
}while (!kbhit());// control cycle ends
// while (i<=3000);
ir=0.0;// motor is stopped
salida=floor(682.66*ir+2048.0);
bajo=salida & 0xff;

alto=salida & 0x0f00;
alto=floor(alto/256.0);
outportb(base+4,bajo);
outportb(base+5,alto);
// Save data to hard disk
cout<<i<<endl;
Inicializa();
i-=1;
for(int jj=1;jj<i;jj++)
{
EscribeArchivo(yy[jj],uu[jj],ref[jj]);
}
Cierra();
}

void Inicializa(void) {
strcpy(Direccion,"c:\\motorcd\\sigue.txt");
if( (fp=fopen(Direccion,"w+")) == NULL)
{
cout<<"File cannot be created";
exit(1);
}
}

void EscribeArchivo(float datos1,float datos2,float datos3) {
fprintf(fp,"%f %f %f \n",datos1,datos2,datos3);
}

void Cierra(void) {
fclose(fp);
}
```

11.8 Frequency Response-Based Design

11.8.1 Model Identification

Another experimental prototype for position control in a PM brushed DC motor has been built. This prototype is the same used in Sect. 10.10, Chap. 10. The identification of the corresponding model has been performed using a frequency response experiment. The details on the steps that have been followed to perform this task are given in the following.

1. The following voltage signal has been applied at the motor armature terminals $u = A\sin(\omega t)$[V] and it has been observed that the measured position of the motor has the following form $\theta = B\sin(\omega t + \phi)$[rad].
2. This experiment has been performed for several values of the frequency ω, and the results are presented in Table 11.1. Note that the phase ϕ is not considered.
3. In Fig. 11.26, the Bode diagram obtained from Table 11.1 is presented. One asymptote with a slope of -20[dB/dec] and one asymptote with a slope of -40[dB/dec] fit these experimental data well. This means that the motor model must have the following structure:

$$\frac{\theta(s)}{U(s)} = \frac{k}{s(s+a)}. \tag{11.52}$$

4. From the intersection of these asymptotes we conclude that:

$$a = 3.653 \tag{11.53}$$

5. The parameter k is found by trial and error until the data obtained from the model in (11.52) fit the experimental data well:

$$k = 62. \tag{11.54}$$

11.8.2 Proportional–Integral–Derivative Control Design

Although the motor model in 11.52 has one pole at $s = 0$, a controller with integral action is desirable to compensate for constant torque disturbances, e.g., the friction effects. Hence, given the second-order structure of the motor model, the design of a PID controller is well suited (see Fig. 11.27).

This means that the open loop transfer function is given as:

$$G(s)H(s) = \frac{k_d s^2 + k_p s + k_i}{s} \frac{k}{s(s+a)}.$$

Table 11.1 Experimental data for model identification of a PM brushed DC motor

ω[rad/s]	$2A$ [V]	$2B$ [rad]
1.5	6	64.97
2	6	46.535
3	6	27.335
4	6	17.305
5	6	12.045
6	6	8.585
7	6	6.29
8	6	5.065
9	6	4.2
10	6	3.355
20	6	0.86
30	6	0.415
40	6	0.235
50	6	0.16
60	6	0.115
70	6	0.08
80	6	0.06
90	6	0.045
100	6	0.04

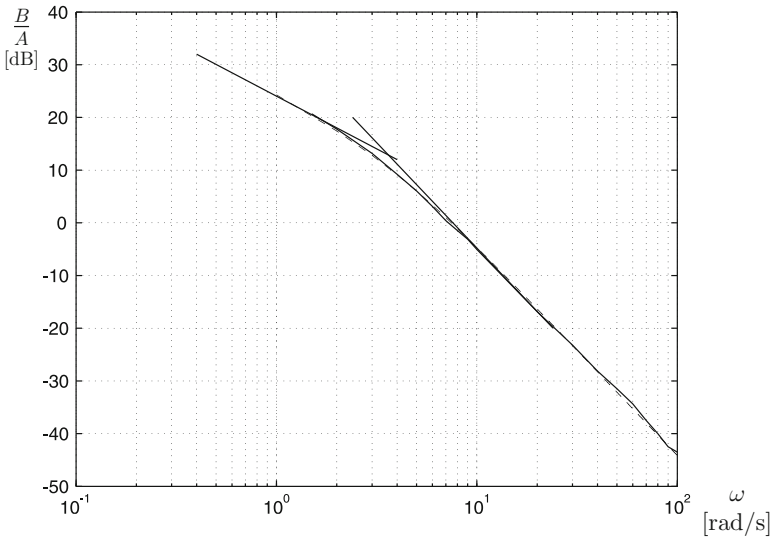

Fig. 11.26 Bode diagram from experimental data used for model identification. Continuous: experimental data (Table 11.1). Dashed: data from the model in (11.52)

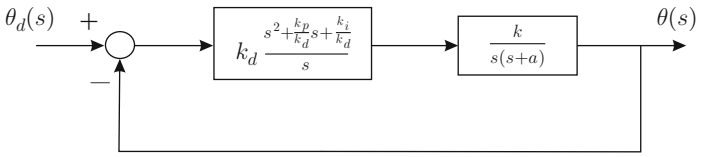

Fig. 11.27 Proportional–integral–derivative control of the position

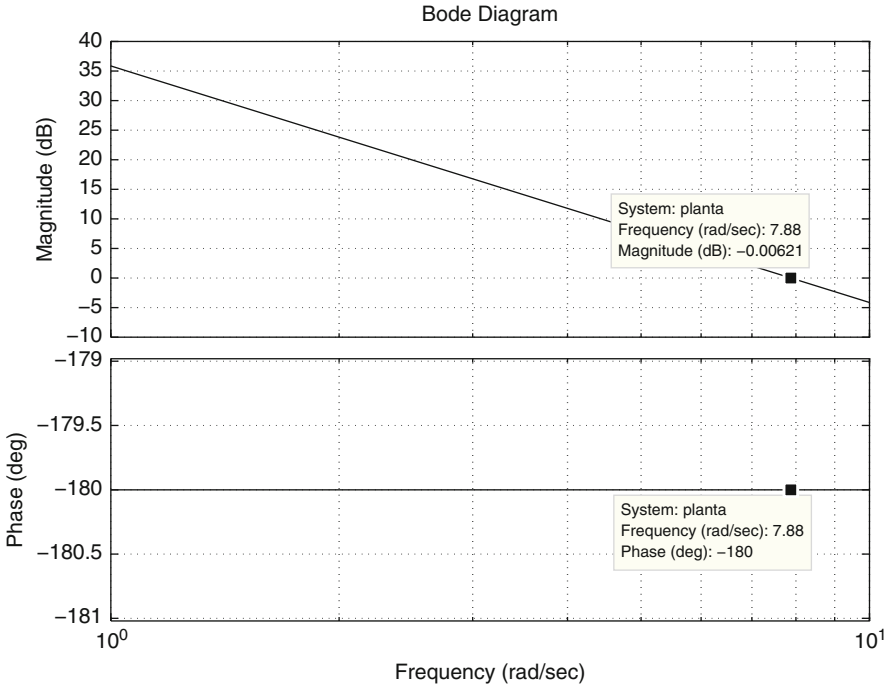

Fig. 11.28 Bode diagrams of the factor $\frac{k}{s^2}$, where (11.54) is used

Proceeding as in Sect. 6.7.3, we can rewrite this transfer function as:

$$G(s)H(s) = k_d z_2 (z+1)\frac{k}{s^2}, \quad z = \frac{s}{z_2}, \quad z_1 = a,$$
$$k_p = k_d(z_1 + z_2), \quad k_i = k_d z_1 z_2. \tag{11.55}$$

Hence, the factor $\frac{k}{s^2}$ is considered the "plant to be controlled" and the factor $k_d z_2(z+1)$ the "controller."

The Bode diagrams of the factor $\frac{k}{s^2}$ are presented in Fig. 11.28 where (11.54) is used. The crossover frequency is $\omega_1 = 7.88$[rad/s] and the phase is $-180°$. This means that the phase margin is 0°. It is chosen to maintain the same crossover frequency and to design a 20% overshoot. Hence, using:

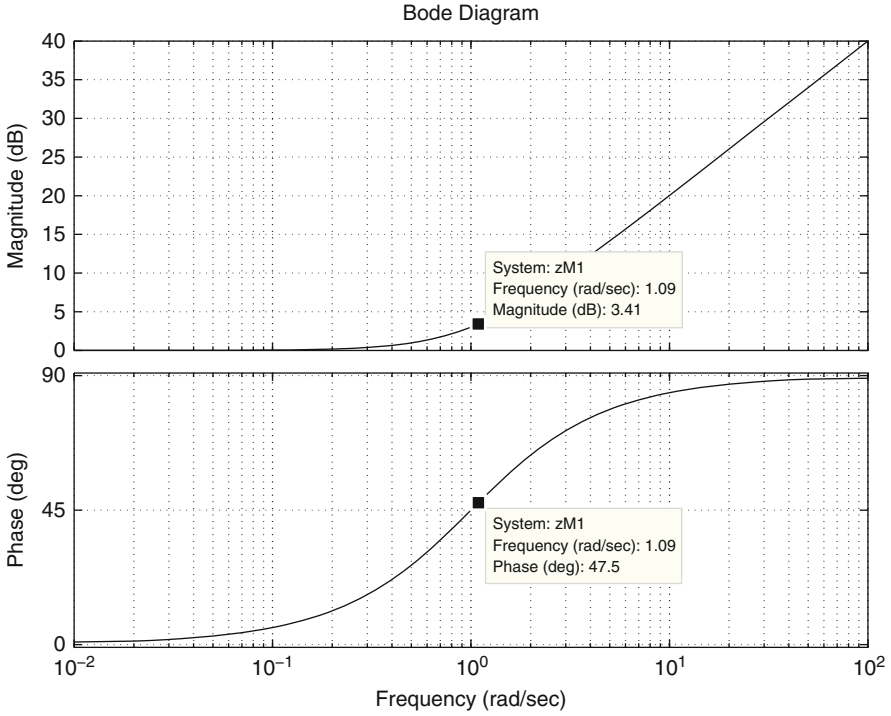

Fig. 11.29 Bode diagrams of the factor $z + 1$. The horizontal axis represents $\frac{\omega}{z_2}$

$$\zeta = \sqrt{\frac{\ln^2(M_p/100)}{\ln^2(M_p/100) + \pi^2}},$$

it is found that $\zeta = 0.4559$ is required. According to Table 6.5, this means that a 47.5° phase margin is needed. Hence, a 47.5° phase lead has to be contributed by the controller, i.e., by the factor $z + 1$, when $\omega = \omega_1 = 7.88$[rad/s]. It is observed in Fig. 11.29 that a 47.5° phase lead is obtained when $\frac{\omega}{z_2} = 1.09$. Hence, using $\omega = \omega_1 = 7.88$[rad/s], the following is obtained: $z_2 = 7.2294$. On the other hand, from the magnitude condition:

$$|k_d z_2(z + 1)|_{\text{dB}} = 20 \log(k_d z_2) + |z + 1|_{\text{dB}} = 0[\text{dB}],$$

where $|z + 1|_{\text{dB}} = 3.41$[dB], according to Fig. 11.29. Thus, we have:

$$k_d = \frac{1}{z_2} 10^{\frac{-3.41}{20}} = 0.0935.$$

Finally, from (11.55) and $z_1 = a$, the following is found:

$$k_p = 1.0177, \quad k_i = 2.4697.$$

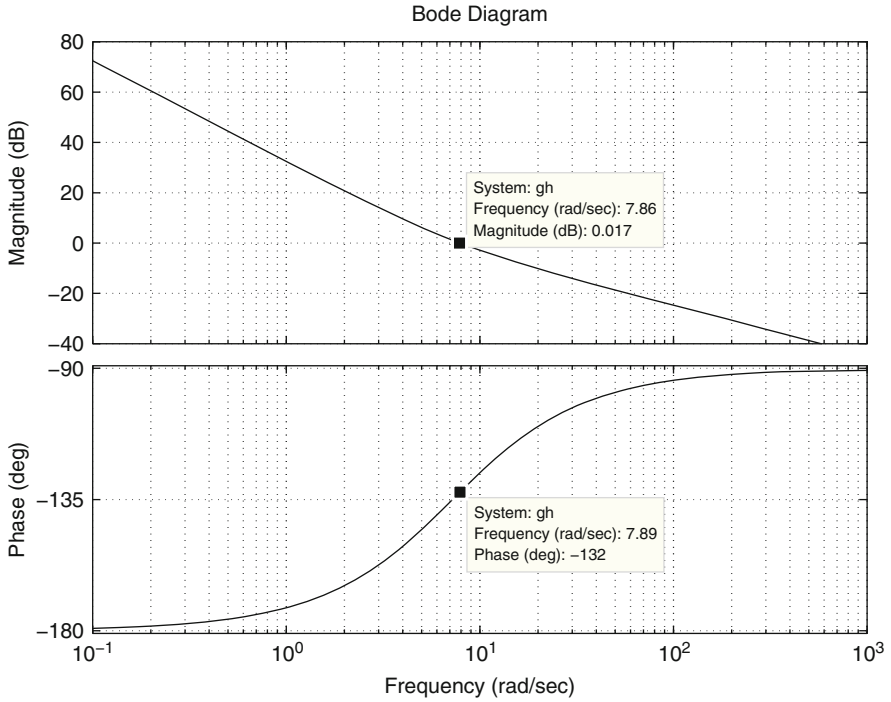

Fig. 11.30 Bode diagrams of the transfer function $\frac{k_d s^2+k_p s+k_i}{s}\frac{k}{s(s+a)}$. $k_d = 0.0935$, $k_p = 1.0177$, $k_i = 2.4697$, and (11.53), (11.54) are used

In Fig. 11.30, a 7.88[rad/s] crossover frequency is observed and a 48° phase margin. This means that the frequency response design is correct. On the other hand, the simulations presented in Fig. 11.31 show a 33% overshoot, instead of the desired 20%, and 0.192[s] as rise time. See Sect. 6.7.1 for an explanation for this difference between the desired and the obtained overshoots. Some experimental results are presented in Fig. 11.32 where 29% is obtained as overshoot and 0.21[s] as rise time. Note that these values are very close to the simulation values.

According to Sect. 6.5.2 an additional phase lag of $-\omega T$[rad] must be considered in Fig. 11.30 to take into account the effects of time delay induced by the numerical computation of the control law in experiments. As the sampling period used in experiments is 0.01[s], then a time delay of $T = 0.01$[s] may be assumed to take into account the worst case and $\omega = 7.88$[rad/s] is the crossover frequency. Then $-\omega T \times 180°/\pi = 4.51°$ is the maximal additional phase lag contributed by the digital implementation of the controller. This shifts the phase margin to 43.49°; hence, no performance deterioration is expected in the experiments, which is corroborated in Fig. 11.32.

Finally, in Fig. 11.33, we compare the control scheme in Fig. 11.27, $k_d = 0.0935$, $k_p = 1.0177$, $k_i = 2.4697$, with that presented in (11.38). The controller

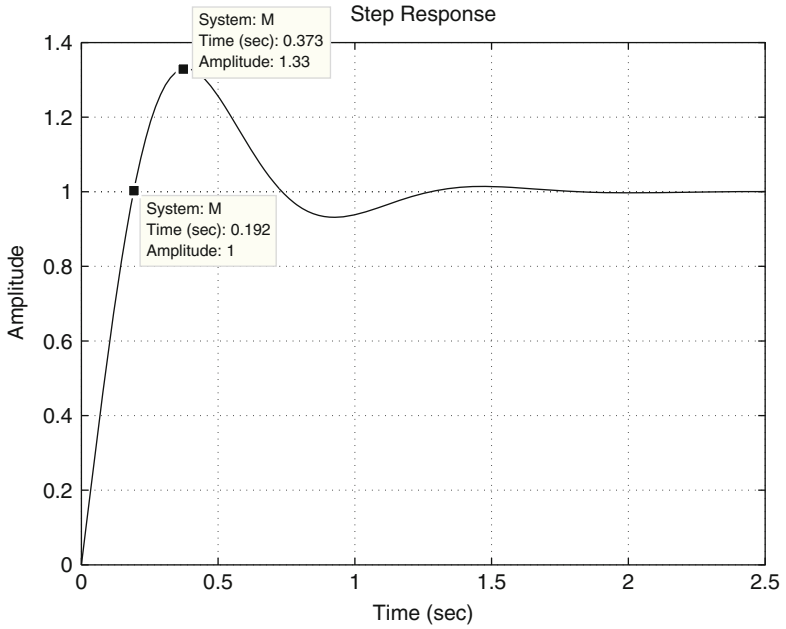

Fig. 11.31 Simulation response of the closed loop system in Fig. 11.27. $k_d = 0.0935$, $k_p = 1.0177$, $k_i = 2.4697$

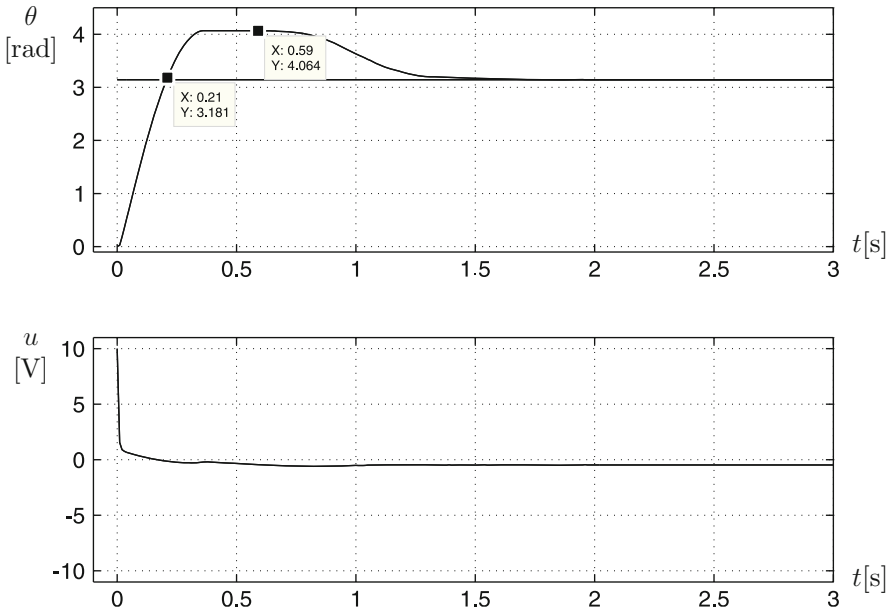

Fig. 11.32 Experimental results for the control scheme in Fig. 11.27. $k_d = 0.0935$, $k_p = 1.0177$, $k_i = 2.4697$

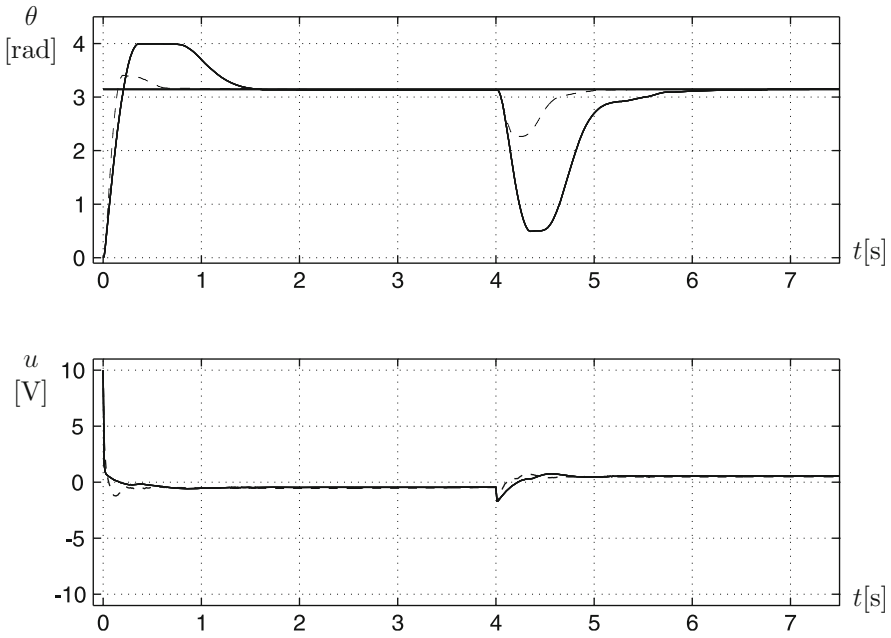

Fig. 11.33 Experimental comparison of the control scheme in Fig. 11.27 (continuous line) and controller in (11.38) (dashed line)

gains of the controller in (11.38) were computed using $t_r = 0.19$[s], $M_p = 20\%$, (11.9), (11.10), (11.11), $\sigma = \zeta\omega_n$, (11.36), (11.37), and $f = 5$. This yields $k_{p1} = 2.3571$, $k_{p2} = 0.889$, $k_{i1} = 11.7855$ and $k_{d2} = 0.1995$. A constant disturbance is applied for $t \geq 4$[s] via software. This has been done by adding a constant 2.5[V] to the voltage delivered by the controllers, which has to be sent to the motor. We realize that the effects of the disturbance are much smaller when the controller in (11.38) is used. Also, a faster response is achieved with the controller in (11.38).

Figures 11.26, and 11.28 to 11.31, have been plotted using the following MATLAB code in an m-file:

```
clc
clear
W=[1.5 2 3 4 5 6 7 8 9 10 20 30 40 50 60 70 80 90 100];
B=[64.97 46.535 27.335 17.305 12.045 8.585 6.29 5.065 4.2
3.355 0.86 0.415 0.235 0.16 0.115 0.08 0.06 0.045 0.04];
A=[6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6];
magnitude=20*log10(B./A);
figure(1)
semilogx(W,magnitude,'r-')
grid on
hold on
line([0.4 4],[32 12])
```

```
line([2.4 24],[20 -20])
k=62;
a=3.653;
g=tf(k,[1 a 0]);
ww=1:1:100;
[mg,fase]=bode(g,ww);
for i=1:100,
mag(i)=20*log10(mg(1,1,i));
end
semilogx(ww,mag,'k--')
hold off
plant=tf(k,[1 0 0]);
figure(2)
bode(planta)
grid on
z1=a;
w1=7.88;
Mpd=20;
PhaseLead_d=47.5;
zM1=tf([1 1],1)
figure(3)
bode(zM1)
grid on
z2=w1/1.09;
kd=10^(-3.4/20)/z2
kp=kd*(z1+z2)
ki=kd*z1*z2
PID=tf([kd kp ki],[1 0]);
plant2=tf(k,[1 a 0]);
gh=PID*plant2;
figure(4)
bode(gh)
grid on
M=feedback(gh,1,-1);
figure(5)
step(M)
grid on
%%%%%%%%%%%%%% 2-DOF controller design
tr=0.19;
Mp=20;
z=sqrt(log(0.2)^2/(log(0.2)^2+pi^2))
wd=(pi-atan(sqrt(1-z^2)/z))/tr
wn=wd/sqrt(1-z^2)
sigma=z*wn;
f=5;
kp1=(sigma^2+wd^2)/k
ki1=(sigma^2+wd^2)*f/k
kp2=2*sigma*f/k
kd2=(2*sigma+f-a)/k
```

11.8.3 Prototype Construction

The electric diagram in addition to the computer program used to evaluate the control algorithms is identical to the electric diagram and the Builder 6 C++ code presented in Chap. 12. The only difference is that in Chap. 12 two encoders have to be read, whereas in the present chapter only one encoder is to be read. Also, a microcontroller PIC16F877A is used in Chap. 12 and in the present chapter. The code for programming this microcontroller is identical in both chapters. Again, in the present chapter only one encoder has to be read. Thus, the code for programming is simpler in the present chapter.

In particular, the identification experiments described in Sect. 11.8.1 require to the voltage signal $u = A\sin(\omega t)$[V] to be applied at the motor armature terminals. This is performed by writing the code line "iast=A*sin(w t)," provided that A=3 and w are defined as constants at the beginning of the program. This is all that we have to do as software and hardware are so contrived that the numerical value of "iast" appears as the voltage at the motor armature terminals.

This prototype has been built because it presents some advantages when performing the frequency response experiments described in Sect. 11.8.1. When using the prototype depicted in Fig. 11.18 for frequency response identification, the motor position has a large drift, which poses difficulties for the measurement of the position amplitude. Such a problem is eliminated with the new prototype. See Chap. 12.

11.9 Summary

The position of a PM brushed DC motor has been controlled under the effect of an external torque disturbance. Two controllers have been tested: a classical PID controller and a modified PID controller. Although both controllers solve the problem, the modified PID controller has a better performance. As this controller requires numerical values of the motor parameters, an experimental identification of the motor model is also presented. In this chapter, it has also been shown how to tune a classical PID controller using the numerical values of the plant parameters. However, the main advantage of this controller is that it can be tuned by a trial and error procedure, i.e., without any information about the plant parameters (see Sect. 5.3 in Chap. 5). On the other hand, a PD position controller has also been designed to solve the trajectory tracking problem. This is a complex task requiring the exact values of the motor parameters. Detailed explanations are included on how to implement these controllers using a microcontroller and a personal computer. The reason for using a personal computer is that the implementation of the PD controller for trajectory tracking requires computation of sine and cosine functions, which consumes too much time when using a microcontroller.

When a control system is implemented in practice, it is very important that the signal processing performed by the software and hardware precisely corresponds

to the mathematical operations indicated by the controller. Control systems that are built in this book satisfy this requirement and this can be corroborated by following their careful descriptions.

11.10 Review Questions

1. Consider the PID position control of a DC motor. What would happen if either the torque disturbance or the desired position were not constant?
2. Industrial robots employ PID controllers, even for trajectory tracking. How do you think that this task can be accomplished (see Sect. 6.7.4)?
3. How does each one of the classical PID controller gains affect the response of a position control system? If a disturbance were rejected very slowly, how would you modify the controller gains to improve performance?
4. In Chap. 10, it was shown that a PI controller is suitable for compensating for the effects of an external torque disturbance in a velocity control system, Why do you think that a PID controller has to be used instead of a PI controller in the case of position control?
5. List some advantages and disadvantages when using a microcontroller or a personal computer to implement digital controllers.
6. A PM brushed DC motor can also be controlled without using an internal electric current loop if it is assumed that the armature inductance is negligible, especially in small motors. What changes must be performed in the circuits of Figs. 11.18 and 11.21 to work without an electric current loop?
7. Describe the time response and the frequency response methods presented in this chapter to experimentally identify the parameters of a DC motor.
8. What is a lead compensator?
9. What are the advantages of using lead compensators as position controllers?
10. From the theoretical point of view, the response of a position control system can be rendered as fast and as well damped as desired using a PID controller. List some problems arising in practice when the PID controller gains are too large.

References

1. DAC0800 8-bit Digital-to-Analog Converter, Data sheet, National Semiconductor Corporation, 1995.
2. PIC16F877A Enhanced Flash Microcontroller, Data sheet, Microchip Technology Inc., 2003.
3. Custom Computer Services Incorporated, CCS C Compiler Reference Manual, 2003.
4. PCL-812PG User's Manual, *Advantech*, Taiwan, 1996.
5. K. Ogata, *Modern control engineering*, 4th edition, Prentice-Hall, Upper Saddle River, 2002.
6. J. Chiasson, *Modeling and high-performance control of electric machines*, IEEE Press-Wiley Interscience, New Jersey, 2005.
7. Parker Automation, Position Systems and Controls, Training and Product Catalog DC-ROM, *Compumotor's Virtual Classroom*, 1998.