Chapter 15
Control of a Furuta Pendulum

The Furuta pendulum is depicted in Fig. 15.1. Controlling the Furuta pendulum may be described by referring to the old trick known as the broomstick-balancing problem. In a Furuta pendulum, the person's arm is replaced by a beam technically known as the *arm*, which receives the torque generated by a permanent magnet (PM) brushed direct current (DC) motor at one of its ends. Hence, the arm can only move on a horizontal plane such that its other end describes a circumference. At this end another beam is placed, technically known as the *pendulum*, which joins to the arm by means of some bearing allowing the pendulum free movement. This means that no motor is placed at this joint. However, the bearing in this place constrains the pendulum to only move by rotating on a plane that is orthogonal to the arm. This means that the pendulum's free end can only describe a circumference whose center is placed at the point where the arm and the pendulum join.

One feature that renders the Furuta pendulum interesting to control is that it is inherently unstable. This can be seen by recalling the broomstick-balancing problem: the broomstick always tends to leave its inverted position, falling to the floor. Hence, designing a controller for this problem is not a trivial task and it requires the knowledge of suitable control theory tools.

The variables and the parameters involved in the Furuta pendulum depicted in Fig. 15.1 are the following:

- θ_0 is the arm angular position, in radians, measured with respect to some arbitrary reference position.
- θ_1 is the pendulum angular position, in radians, measured with respect to its inverted position.
- τ is the torque generated by the electric motor that is applied to the arm.
- I_0 is the arm inertia when it rotates around one of its ends plus the motor inertia.
- L_0 is the arm length.
- m_1, l_1 and J_1 stand for the pendulum mass, the location of the center of mass, and the inertia respectively. J_1 is computed assuming that the pendulum rotates

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_15

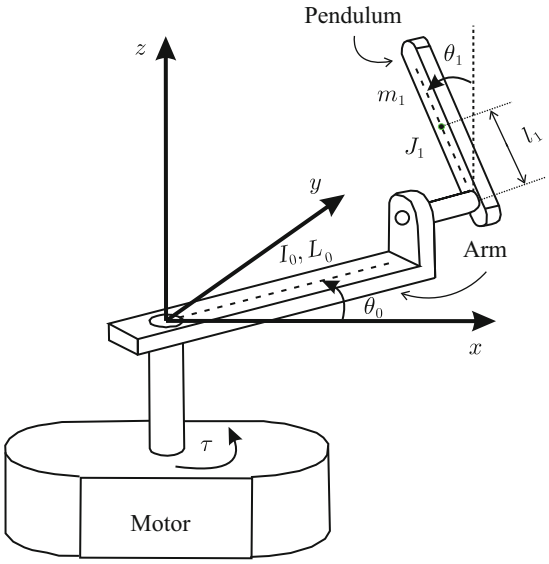

Fig. 15.1 Furuta pendulum

around its center of mass. Note that the pendulum consists of a beam whose mass is not negligible.

- $g = 9.81[\text{m/s}^2]$ is the gravity acceleration.

15.1 Mathematical Model

As described above, the Furuta pendulum is composed of two bodies that interact. The mathematical model for this class of mechanisms is commonly obtained using the *Euler–Lagrange equations* [1]. In the case of the Furuta pendulum, the Euler–Lagrange equations are written as:

$$\begin{aligned} \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_0} - \frac{\partial L}{\partial \theta_0} &= \tau, \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_1} - \frac{\partial L}{\partial \theta_1} &= 0. \end{aligned} \tag{15.1}$$

The zero on the right-hand side of last equation indicates that no external torque is applied at the place where the arm and the pendulum join. The Lagrangian L is defined as:

$$\begin{aligned} L &= K - P, \\ K &= K_0 + K_1, \end{aligned} \tag{15.2}$$

where K_0 is the arm's kinetic energy, K_1 is the pendulum's kinetic energy, and P is the pendulum's potential energy. No potential energy is considered for the arm because it moves on an horizontal plane, i.e., its potential energy can be assumed to be zero.

The computation of the *kinetic energy* for a body whose mass is distributed on a volume can be simplified if it is decomposed into two parts: one part due to translational movement of the body's center of mass (assumed to be a particle) and the other part due to body's rotative movement around its center of mass. However, if the body movement can be described in a simple manner such a decomposition is not necessary. For instance, the arm movement is easily described as a rotative movement of a beam around one of its ends and the rotative movement of the motor around the same axis. Hence, it is not difficult to find that:

$$K_0 = \frac{1}{2} I_0 \dot{\theta}_0^2. \tag{15.3}$$

On the other hand, the pendulum movement is more complex because it depends on the combined movements of the arm and the pendulum. In this case, it is more convenient to decompose the pendulum movement into two parts as explained above: the translational movement of a particle with mass m_1 located at the pendulum center of mass and the rotative movement of a beam around an axis passing through the pendulum's center of mass. Hence:

$$K_1 = \frac{1}{2} J_1 \dot{\theta}_1^2 + \frac{1}{2} m_1 v_1^T v_1, \tag{15.4}$$

where v_1 is a vector that stands for the velocity of the pendulum center of mass, i.e.,:

$$v_1 = \begin{bmatrix} \frac{dX_x}{dt} \\ \frac{dX_y}{dt} \\ \frac{dX_z}{dt} \end{bmatrix}, \quad X = \begin{bmatrix} X_x \\ X_y \\ X_z \end{bmatrix}, \tag{15.5}$$

where X_x, X_y y X_z are the Cartesian coordinates of the pendulum's center of mass. This means that X is the position vector of the pendulum's center of mass. According to Figs. 15.2a and b the following is found:

$$X = \begin{bmatrix} X_x \\ X_y \\ X_z \end{bmatrix} = \begin{bmatrix} L_0 \cos(\theta_0) - l_1 \sin(\theta_1) \sin(\theta_0) \\ L_0 \sin(\theta_0) + l_1 \sin(\theta_1) \cos(\theta_0) \\ l_1 \cos(\theta_1) \end{bmatrix}. \tag{15.6}$$

Using (15.4), (15.5), (15.6) and after rearranging terms, the following is found:

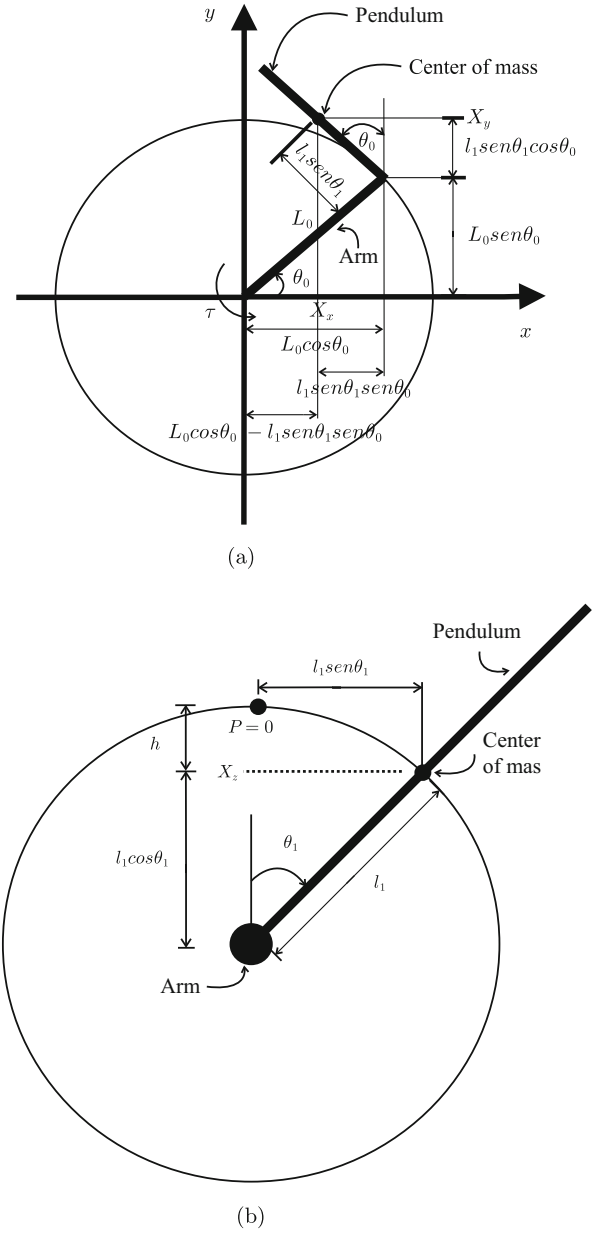

Fig. 15.2 Geometric relationships in the Furuta pendulum . (**a**) Superior view. (**b**) View of the plane orthogonal to arm

$$K_1 = \frac{1}{2}J_1\dot{\theta}_1^2 + \frac{1}{2}m_1\left[(L_0\dot{\theta}_0)^2 + (l_1\dot{\theta}_0)^2\sin^2(\theta_1) + (l_1\dot{\theta}_1)^2 + 2\dot{\theta}_0\dot{\theta}_1 L_0 l_1\cos(\theta_1)\right]. \tag{15.7}$$

Finally, to compute the pendulum *potential energy*, the point where $\theta_1 = 0$ is used as the reference point, i.e., where $P = 0$. Hence, recalling that the potential energy is the scalar product of the applied force vector, i.e., the pendulum's weight $m_1 g$, and the distance vector going from the pendulum's center of mass to the point where $P = 0$, the following is obtained:

$$P = -hm_1 g, \quad h = l_1 - l_1\cos(\theta_1),$$

where the sign "−" is due to the fact that the vectors referred to above have an angle that is greater than 90°. Then:

$$P = m_1 g l_1(\cos(\theta_1) - 1). \tag{15.8}$$

Using (15.3), (15.7), (15.8), the Lagrangian L is formed according to (15.2). After replacing in the Euler–Lagrange equations (15.1), performing the indicated operations and suitably rearranging the resulting terms, the following is found:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = F, \tag{15.9}$$

$$M(q) = \begin{bmatrix} I_0 + m_1(L_0^2 + l_1^2\sin^2(\theta_1)) & m_1 l_1 L_0\cos(\theta_1) \\ m_1 l_1 L_0\cos(\theta_1) & J_1 + m_1 l_1^2 \end{bmatrix},$$

$$C(q,\dot{q}) = \begin{bmatrix} \frac{1}{2}m_1 l_1^2\dot{\theta}_1\sin(2\theta_1) & -m_1 l_1 L_0\dot{\theta}_1\sin(\theta_1) + \frac{1}{2}m_1 l_1^2\dot{\theta}_0\sin(2\theta_1) \\ -\frac{1}{2}m_1 l_1^2\dot{\theta}_0\sin(2\theta_1) & 0 \end{bmatrix},$$

$$g(q) = \begin{bmatrix} 0 \\ -m_1 l_1 g\sin(\theta_1) \end{bmatrix}, \quad F = \begin{bmatrix} \tau \\ 0 \end{bmatrix}, \quad q = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}.$$

The expression in (15.9) represents the Furuta pendulum mathematical model, which is nonlinear because it includes trigonometric functions in addition to several products between velocities $\dot{\theta}_0$ and $\dot{\theta}_1$.

15.2 A Controller to Swing Up the Pendulum

Swinging up of the pendulum is the task where the pendulum is taken from the bottom stable position, i.e., $\theta_1 = \pi$, to the inverted unstable position, i.e., $\theta_1 = 0$ or $\theta_1 = 2\pi$. This is performed experimentally in Sect. 15.7 by using the following controller:

$$\tau = \frac{-\frac{k_\omega}{\det(M(q))}F(q,\dot{q}) - k_\theta\theta_0 - k_\delta\dot{\theta}_0}{k_E E(q,\dot{q}) + \frac{k_\omega}{\det(M(q))}(J_1 + m_1 l_1^2)}, \tag{15.10}$$

$$E(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} + m_1 g l_1(\cos\theta_1 - 1),$$

$$F(q,\dot{q}) = -(J_1 + m_1 l_1^2)m_1 l_1^2\dot{\theta}_1\dot{\theta}_0\sin(2\theta_1) - \frac{1}{2}m_1^2 l_1^3 L_0\dot{\theta}_0^2\cos\theta_1\sin(2\theta_1)$$
$$-m_1^2 l_1^2 L_0 g\cos\theta_1\sin\theta_1 + (J_1 + m_1 l_1^2)m_1 l_1 L_0\dot{\theta}_1^2\sin\theta_1,$$

where $E(q,\dot{q}) = K_0 + K_1 + P$ is the Furuta pendulum's total energy and k_θ, k_δ, k_ω, K_E are positive constants that satisfy:

$$\frac{k_\omega}{k_E} > 2m_1 g l_1(I_0 + m_1 l_1^2 + m_1 L_0^2).$$

The controller in (15.10) was proposed in [2], chapter 6. The interested reader is referred to that book for more details. Swinging up the pendulum in the Furuta pendulum is not an easy task as the only motor in the mechanism actuates directly on the arm but there is no motor actuating directly on the pendulum. The trick to solving this problem is energy regulation, as we explain in the following. Suppose that the Furuta pendulum's total energy is equal to zero and $\dot{\theta}_0 = 0$. Under these conditions we have that:

$$E(q,\dot{q}) = \frac{1}{2}(J_1 + m_1 l_1^2)\dot{\theta}_1^2 + m_1 g l_1(\cos\theta_1 - 1) = 0.$$

This means that:

$$\dot{\theta}_1 = \pm\sqrt{\frac{2m_1 g l_1}{J_1 + m_1 l_1^2}(1 - \cos\theta_1)}. \tag{15.11}$$

In Fig. 15.3 a plot of $\dot{\theta}_1$ satisfying this expression is presented. This plot represents the trajectory where the pendulum evolves when $E(q,\dot{q}) = 0$ and $\dot{\theta}_0 = 0$. Moreover, its orientation represents the sense of movement of the pendulum variables. According to Fig. 15.3, under these conditions the pendulum will reach one of the configurations $(\theta_1, \dot{\theta}_1) = (0,0)$ or $(\theta_1, \dot{\theta}_1) = (2\pi, 0)$. This suggests regulating the Furuta pendulum's total energy and the arm velocity at zero. Once this is accomplished, it is only a matter of time for the pendulum to reach the inverted unstable position. The controller in (15.10) is designed to achieve $E(q,\dot{q}) = 0$, $\dot{\theta}_0 = 0$ and $\theta_0 = 0$ [2], chapter 6. The latter is required to avoid the arm rotating many (or an infinite number of) times.

Consider the following function:

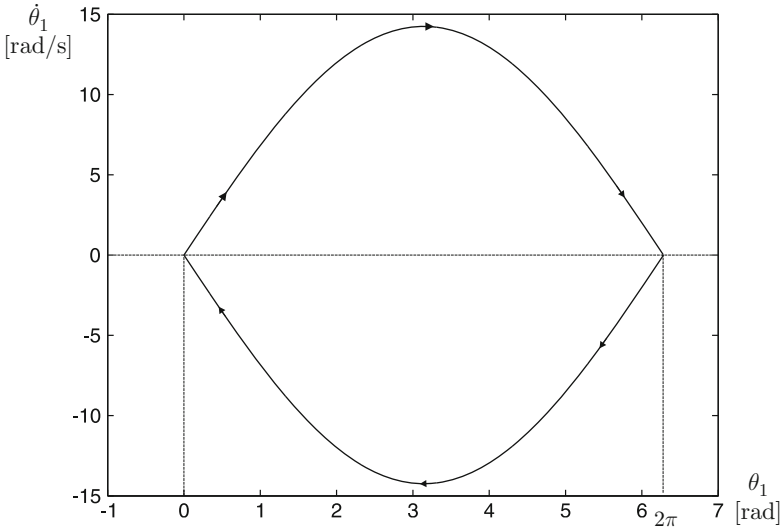

Fig. 15.3 Plot of (15.11) in the phase plane

$$V(q,\dot{q}) = \frac{k_E}{2}E^2(q,\dot{q}) + \frac{k_\omega}{2}\dot{\theta}_0^2 + \frac{k_\theta}{2}\theta_0^2. \tag{15.12}$$

Note that:

$$V(q,\dot{q}) = 0, \quad \text{if } E(q,\dot{q}) = 0,\ \dot{\theta}_0 = 0,\ \theta_0 = 0,$$
$$V(q,\dot{q}) > 0, \quad \text{elsewhere.}$$

These properties are exploited for controller design in [2], chapter 6. The main idea is to choose (15.10) as a controller ensuring that $\dot{V}(q,\dot{q}) < 0$ and $\dot{V}(q,\dot{q}) = 0$ only when $E(q,\dot{q}) = 0$, $\dot{\theta}_0 = 0$ and $\theta_0 = 0$. This forces the system to move in a direction that approaches the conditions $E(q,\dot{q}) = 0$, $\dot{\theta}_0 = 0$, and $\theta_0 = 0$. Once this is achieved the system stays there, i.e., the trajectory in (15.11) is reached. See [2], chapter 6, for more details. However, it is important to stress that this only ensures that the pendulum's inverted unstable configuration is reached, but this does not ensure that the pendulum remains there. To force the pendulum to stay at the inverted unstable configuration it is necessary to employ a stabilizing controller such as that designed in Sect. 15.6. This is performed on the basis of an approximate linear model of the Furuta pendulum, which is obtained in the next section.

15.3 Linear Approximate Model

This book is oriented to the application of linear control techniques, which cannot be directly used when the model of the plant to be controlled is nonlinear such as that shown in (15.9). However, there is a way of solving this problem. The fundamental idea is to find an approximate linear model that represents the model with enough accuracy (15.9) and, then, use such an approximate linear model for controller design. As explained in Sect. 7.3.2, the controller designed in this way is useful for balancing the pendulum at its inverted position $\theta_1 = 0$ only if the pendulum is initially close to such a configuration and remains there, i.e., as long as $\theta_1 \approx 0$.

The first step to this end is to write (15.9) in terms of its state variables. According to Sect. 7.1, given a set of differential equations such as (15.9), a state variables representation can be found by defining the state vector x components as the differential equations' unknown variables and their first $r-1$ time derivatives, where r is the order of the corresponding differential equation. Note that (15.9) consists of two second-order differential equations. The unknown variables in these differential equations are θ_0 and θ_1. Thus, the state vector can be chosen to be given as:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta_0 \\ \dot{\theta}_0 \\ \theta_1 \\ \dot{\theta}_1 \end{bmatrix}. \tag{15.13}$$

Solve (15.9) for $\ddot{q}$ and define:

$$\begin{aligned} \ddot{q} &= \begin{bmatrix} w_1(x_1, x_2, x_3, x_4, \tau) \\ w_2(x_1, x_2, x_3, x_4, \tau) \end{bmatrix}, \\ &= M^{-1}(x_1, x_3)[-C(x_1, x_2, x_3, x_4)[x_2, x_4]^T - g(x_1, x_3) + F]. \end{aligned} \tag{15.14}$$

One important property of matrix $M(q) = M(x_1, x_3)$ is that it is always nonsingular, i.e., the matrix $M^{-1}(x_1, x_3)$ appearing in the last expression always exists [2]. Then, the corresponding state equation can be written as:

$$\dot{x} = f(x, u), \tag{15.15}$$

$$f(x, u) = \begin{bmatrix} f_1(x, u) \\ f_2(x, u) \\ f_3(x, u) \\ f_4(x, u) \end{bmatrix} = \begin{bmatrix} x_2 \\ w_1(x_1, x_2, x_3, x_4, u) \\ x_4 \\ w_2(x_1, x_2, x_3, x_4, u) \end{bmatrix}, \quad u = \tau.$$

Using the ideas in Sect. 7.3.2, an approximate linear model is now obtained that is valid around an operation point. The operation points are the pairs (x^*, u^*)

satisfying:

$$f(x^*, u^*) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

i.e., according to (15.15), (15.14):

$$\begin{aligned} x_2^* = \dot{\theta}_0^* = 0, \quad x_4^* = \dot{\theta}_1^* = 0, \\ \begin{bmatrix} w_1(x_1^*, 0, x_3^*, 0, u^*) \\ w_2(x_1^*, 0, x_3^*, 0, u^*) \end{bmatrix} &= M^{-1}(x_1^*, x_3^*) \\ &\times \left\{ -C(x_1^*, 0, x_3^*, 0) \begin{bmatrix} 0 \\ 0 \end{bmatrix} - g(x_1^*, x_3^*) + \begin{bmatrix} u^* \\ 0 \end{bmatrix} \right\}, \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \end{aligned}$$

From the second row in these conditions, the following is found:

$$\begin{bmatrix} u^* \\ 0 \end{bmatrix} = g(x_1^*, x_3^*) = \begin{bmatrix} 0 \\ -m_1 l_1 g \sin(\theta_1^*) \end{bmatrix},$$

i.e.,

$$u^* = 0, \quad x_3^* = \theta_1^* = \pm n\pi,$$

where n can be any positive or zero integer. Note that there is no condition for $x_1^* = \theta_0^*$, which means that this variable can be chosen arbitrarily. As it is desired to balance the pendulum at its inverted position, i.e., $n = 0$, the following operation point is chosen:

$$x^* = \begin{bmatrix} x_1^* \\ x_2^* \\ x_3^* \\ x_4^* \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad u^* = 0. \tag{15.16}$$

According to (7.22), (7.23), (7.24), the nonlinear model in (15.14), (15.15), can be approximated by the linear model:

$$\dot{z} = Az + Bv, \tag{15.17}$$

$$A = \left[\begin{array}{cccc} \frac{\partial f_1(x,u)}{\partial x_1} & \frac{\partial f_1(x,u)}{\partial x_2} & \frac{\partial f_1(x,u)}{\partial x_3} & \frac{\partial f_1(x,u)}{\partial x_4} \\ \frac{\partial f_2(x,u)}{\partial x_1} & \frac{\partial f_2(x,u)}{\partial x_2} & \frac{\partial f_2(x,u)}{\partial x_3} & \frac{\partial f_2(x,u)}{\partial x_4} \\ \frac{\partial f_3(x,u)}{\partial x_1} & \frac{\partial f_3(x,u)}{\partial x_2} & \frac{\partial f_3(x,u)}{\partial x_3} & \frac{\partial f_3(x,u)}{\partial x_4} \\ \frac{\partial f_4(x,u)}{\partial x_1} & \frac{\partial f_4(x,u)}{\partial x_2} & \frac{\partial f_4(x,u)}{\partial x_3} & \frac{\partial f_4(x,u)}{\partial x_4} \end{array} \right]_{x=x^*,u=u^*},$$

$$B = \left[\begin{array}{c} \frac{\partial f_1(x,u)}{\partial u} \\ \frac{\partial f_2(x,u)}{\partial u} \\ \frac{\partial f_3(x,u)}{\partial u} \\ \frac{\partial f_4(x,u)}{\partial u} \end{array} \right]_{x=x^*,u=u^*},$$

where $z = x - x^*$ and $v = u - u^*$. The expressions in (15.9), (15.14), (15.15), (15.16), (15.17), define a mathematical procedure that (although tedious it is straightforward) yields:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-g m_1^2 l_1^2 L_0}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{(I_0 + m_1 L_0^2) m_1 l_1 g}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{J_1 + m_1 l_1^2}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2} \\ 0 \\ \frac{-m_1 l_1 L_0}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2} \end{bmatrix}. \tag{15.18}$$

The expressions (15.17), (15.18), represent the approximate linear model we were looking for. It is important to stress that this model is also valid if $x_3^* = \theta_1^* = 2\pi$. This is because $\sin(x_3^*)$ does not change when evaluated at $x_3^* = 0$ or $x_3^* = 2\pi$. The reader is advised to verify this fact. As stated above, using (15.17) and (15.18), a linear state feedback controller can be designed that, however, is only ensured to perform well if the system state and the input are constrained to evolve around the operation point (15.16). This is also stated by indicating that $z \approx 0$ and $v = u = \tau \approx 0$ (because $u^* = 0$). This means that taking the pendulum from $\theta_1 = \pi$ to $\theta_1 = 0$, or $\theta_1 = 2\pi$, is a task that cannot be ensured with a controller designed on the basis of the model (15.17), (15.18). Such a problem is solved by using the controller in (15.10).

15.4 A Differential Flatness-Based Model

The differential flatness property of the model (15.17), (15.18) is studied in this section. To simplify the algebraic manipulation of model (15.17), (15.18), the following constants are defined:

$$a = \frac{-gm_1^2 l_1^2 L_0}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2}, \quad b = \frac{(I_0 + m_1 L_0^2) m_1 l_1 g}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2}$$

$$c = \frac{J_1 + m_1 l_1^2}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2}, \quad d = \frac{-m_1 l_1 L_0}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2}.$$

Hence, the controllability matrix is given as:

$$C_o = \left[B \;\; AB \;\; A^2B \;\; A^3B \right] = \begin{bmatrix} 0 & c & 0 & ad \\ c & 0 & ad & 0 \\ 0 & d & 0 & bd \\ d & 0 & bd & 0 \end{bmatrix}.$$

After some straightforward computations it is found that the determinant of this matrix is given as:

$$\det(C_o) = \frac{m_1^4 l_1^4 L_0^2 g^2}{[I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2]^4} \neq 0.$$

Thus, the system (15.17), (15.18), is controllable. According to Sect. 8.2, Chap. 8, this implies that the system (15.17), (15.18) is differentially flat. The inverse matrix of C_o is given as:

$$C_o^{-1} = \frac{adj(C_o)}{\det(C_o)}, \quad adj(C_o) = Cof^T(C_o)$$

where $Cof^T(C_o)$ is the transposed cofactors matrix of C_o. After some straightforward computations we find:

$$C_o^{-1} = \begin{bmatrix} \star & \star & \star & \star \\ \star & \star & \star & \star \\ \star & \star & \star & \star \\ C_{14} & 0 & C_{34} & 0 \end{bmatrix} \frac{1}{\det(C_o)},$$

$$C_{14} = \frac{gm_1^3 l_1^3 L_0^2 \left[m_1^2 l_1^2 L_0^2 - (J_1 + m_1 l_1^2)(I_0 + m_1 L_0^2) \right]}{[I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2]^4},$$

$$C_{34} = \frac{gm_1^3 l_1^3 L_0^2 \left[-\frac{(J_1+m_1l_1^2)^2(I_0+m_1L_0^2)}{m_1 l_1 L_0} + m_1 l_1 L_0 (J_1 + m_1 l_1^2) \right]}{[I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2]^4},$$

where symbol "$\star$" indicates the entries of the matrix that we do not care for. According to Proposition 8.1, the flat output y is given as:

$$y = k \begin{bmatrix} \frac{C_{14}}{\det(C_o)} & 0 & \frac{C_{34}}{\det(C_o)} & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix},$$

where k is an arbitrary nonzero constant, which, by convenience, is proposed to be defined as:

$$k = -\frac{\det(C_o)[I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2]^4}{gm_1^3 l_1^3 L_0^2 (J_1 I_0 + I_0 m_1 l_1^2 + J_1 m_1 L_0^2)},$$

because this yields the simple expression:

$$y = z_1 + h z_3, \quad h = \frac{J_1 + m_1 l_1^2}{m_1 l_1 L_0}.$$

Using the model (15.17), (15.18), and differentiating the flat output y four times, the following is found:

$$\begin{aligned} \dot{y} &= z_2 + h z_4, \\ \ddot{y} &= (a + bh) z_3, \\ y^{(3)} &= (a + bh) z_4, \\ y^{(4)} &= b\ddot{y} + d(a + bh)\tau, \\ d(a + bh) &= \frac{g(J_1 I_0 + I_0 m_1 l_1^2 + J_1 m_1 L_0^2)}{L_0[I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2]} d < 0. \end{aligned} \tag{15.19}$$

Using the Laplace transform and assuming zero initial conditions the following transfer function is found:

$$\frac{Y(s)}{\tau(s)} = \frac{d(a + bh)}{s^2(s^2 - b)}, \tag{15.20}$$

where $Y(s)$ and $\tau(s)$ stand for the flat output and applied torque Laplace transforms respectively. Note that this transfer function has a negative gain, because $d(a +$

$bh) < 0$, and four real open-loop poles located at $s = 0$ (two of them), $s = -\sqrt{b} < 0$, $s = \sqrt{b} > 0$. Recall that b is a positive real number.

15.5 Parameter Identification

A photography of the experimental prototype that has been built is presented in Fig. 15.27. By direct measurement, the following simple mechanism parameters are found:

$$m_1 = 0.02218[\text{Kg}], \quad l_1 = 0.129[\text{m}], \quad L_0 = 0.155[\text{m}].$$

The joint between the pendulum and the arm is composed of an encoder and a solid cylinder $l = 0.019[\text{m}]$ in length, with a radius $R = 0.012/2[\text{m}]$ and a mass $m = 0.015[\text{Kg}]$. Inertia of this cylinder when rotating on its longitudinal axis is given as [3, 4], pp. 271:

$$I_C = \frac{1}{2} m R^2.$$

The pendulum is a stick with a mass m_1 and length $2l_1$. Inertia of this stick when rotating on an axis that is orthogonal to the stick, passing through its longitudinal center, is given as [3, 4], pp. 271:

$$I_1 = \frac{1}{12} m_1 (2l_1)^2.$$

The pendulum inertia is computed by adding these inertias, i.e.,:

$$J_1 = I_C + I_1 = 0.0001845[\text{Kg m}^2].$$

The motor is a cylinder with a diameter $D = 0.05[\text{m}]$. It is assumed that the rotor is also a cylinder with a diameter equal to one half the motor diameter, i.e., the rotor radius is given as $r_r = \frac{1}{2}\frac{D}{2}$. The motor mass is $m_m = 0.6[\text{Kg}]$. The rotor mass is assumed to be one half the motor mass, and 0.005[Kg] of a screw joining the arm and the motor shaft must also be considered. Thus, the rotor inertia is computed as:

$$I_r = \frac{1}{2}\left(\frac{m_m}{2} + 0.005\right) r_r^2.$$

The arm is a stick with a length L_0 and mass $m_0 = 0.027[\text{Kg}]$ rotating on an axis that is orthogonal to the arm, passing through one of its ends. Hence, the inertia of this stick is [3]:

Table 15.1 Numerical values of a Furuta pendulum

Symbol	Description	Value	Units
g	Gravity constant	9.81	m/s^2
l_1	Distance to center of mass (pendulum)	0.129	m
L_0	Arm length	0.1550	m
m_1	Pendulum mass	0.02218	Kg
J_1	Pendulum inertia	0.0001845	Kg m^2
I_0	Arm inertia	0.00023849	Kg m^2

$$I_s = \frac{1}{3} m_0 L_0^2.$$

Thus, the arm inertia is computed as:

$$I_0 = I_r + I_s = 0.00023849[\text{Kg m}^2].$$

The numerical parameter values of the Furuta pendulum that has been built are summarized in Table 15.1.

15.6 Design of a Stabilizing Controller

In this section, a controller is designed that is intended to regulate the Furuta pendulum variables at $x = x^*$ or, equivalently, at $z = 0$. This is performed on the basis of the approximate linearized model (15.17), (15.18). This implies that the controller is useful only if the initial conditions are close to the desired configuration, i.e., $x(0) \approx x^*$ or, equivalently, $z(0) \approx 0$. Thus, the controller designed in this section is not capable of taking the pendulum from the bottom stable position $x_3 = \pi$ to the inverted unstable position $x_3 = 0$ or $x_3 = 2\pi$. Such a task must be performed using the controller presented in (15.10). The interested reader is encouraged to review [2], chapter 6 (also see Sect. 15.2 in the present book).

It is known, from Sect. 7.11, that given the system (15.17), (15.18), it is possible to accomplish $\lim_{t\to\infty} z(t) = 0$ using the following state feedback controller:

$$\begin{aligned} \tau &= -Kz, \quad K = [k_1, k_2, k_3, k_4], \\ &= -k_1 z_1 - k_2 z_2 - k_3 z_3 - k_4 z_4. \end{aligned} \tag{15.21}$$

This requires selection of the gain vector K such that all the eigenvalues of matrix $A - BK$ are suitably assigned. Of course, a basic requirement is that the real part of all of the eigenvalues of matrix $A - BK$ are negative. However, the eigenvalues of matrix $A - BK$ must also be assigned such that a desired closed-loop performance is accomplished. Moreover, sometimes it is required to choose a new gain vector

K to further improve the performance. Solving this problem is not an easy task in practice, where fast and damped eigenvalues are not the only things that matter because of noise and some other problems related to hardware implementation.

In this section, a methodology is introduced to select the vector gain K that exploits the differential flatness property of the system (15.17), (15.18). The main idea is that a linear state space differentially flat system can be written in terms of a transfer function without any zeros (see Sect. 15.4). Then, we can apply classical control design tools such as the root locus to select the controller gains. An advantage of this approach is that it gives the designer important information on how to select the controller gains to improve performance in a desired direction. This methodology is presented in detail in the following.

According to Sect. 15.4, the state space system (15.17), (15.18), is equivalent to the transfer function in (15.20). To control this open-loop unstable plant consider the block diagram in Fig. 15.4. Note that this is a multi-loop positive feedback system. The reason for positive feedback is the negative gain $d(a+bh) < 0$. As a matter of fact, the closed-loop transfer function is given as:

$$\frac{Y(s)}{Y_d(s)} = \frac{\beta d(a+bh)}{s^4 - d(a+bh)k_v s^3 - (b+\beta d(a+bh))s^2 - k_d \beta d(a+bh)s - k_p \beta d(a+bh)}, \tag{15.22}$$

where $Y_d(s) = 0$. It is not difficult to realize that a negative $d(a+bh)$ renders it possible for all the coefficients of the characteristic polynomial to be positive, a necessary (but not sufficient) condition to ensure that all its roots have a negative real part.

The transfer function of the two internal loops in Fig. 15.4 is given as:

$$\frac{\ddot{Y}(s)}{\ddot{Y}_d(s)} = F(s) = \frac{\beta d(a+bh)}{s^2 - d(a+bh)k_v s - (b+\beta d(a+bh))} \tag{15.23}$$

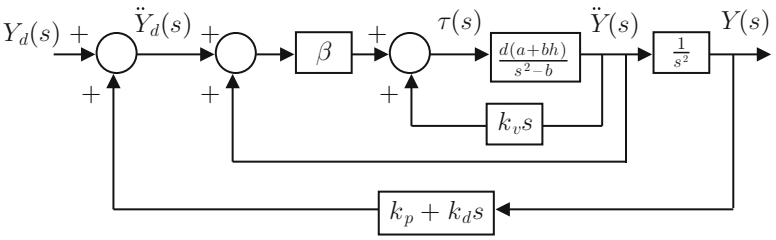

Fig. 15.4 Block diagram of the closed-loop control system

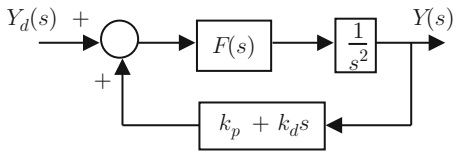

Fig. 15.5 Equivalent block diagram to that in Fig. 15.4

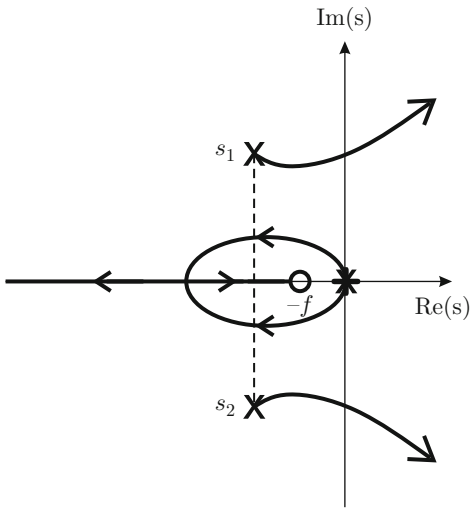

Fig. 15.6 Root locus for the closed-loop control system in Fig. 15.5

Hence, the block diagram in Fig. 15.4 can be rewritten as in Fig. 15.5. Using the numerical parameters given in Table 15.1, the following is found:

$$d(a + bh) = -1.2186 \times 10^5, \quad b = 93.9951.$$

Taking into account these numerical values $k_v = 2.3755 \times 10^{-4}$ and $\beta = 0.0041$ are proposed. This selection of controller gains renders the transfer function in (15.23) stable with poles assigned at the locations shown in Table 15.2a). Using:

$$\frac{\pm 180^\circ (2q + 1)}{n - m}, \quad q = 0, 1, 2, \ldots,$$

where $n = 4$ and $m = 1$, it is found that the root locus diagram of the block diagram in Fig. 15.5 has three asymptotes at $\pm 60^\circ$ and $\pm 180^\circ$ with respect to the positive real axis. Thus, $f = k_p/k_d = 1$ is proposed to shape the root locus diagram as indicated in Fig. 15.6. Note that the open-loop zero at $s = -f$ is not a closed-loop zero as corroborated by (15.22). Choose $k_d = 2.88$ and compute $k_p = k_d f = 2.88$ to assign the poles of (15.22) as shown in Table 15.2a).

According to the block diagram in Fig. 15.4, using the expressions in (15.19) it is found that the controller is given as:

$$\tau = k_v y^{(3)} + \beta(\ddot{y}_d + \ddot{y}), \tag{15.24}$$

Table 15.2 Controller gains and the assigned open-loop and closed-loop poles

	Controller gains	Poles of (15.23)	Poles of (15.22)
a)	$k_v = 2.3755 \times 10^{-4}$, $\beta = 0.0041$, $k_d = 2.88$, $k_p = 2.88$	$s_{1,2} = -14.4734 \pm 14.0048j$	$-12.1989 \pm 11.8679j$, -2.7284, -1.8206
b)	$k_v = 1.2 \times 2.3755 \times 10^{-4}$, $\beta = 1.4 \times 0.0041$, $k_d = 4.59$, $k_p = 4.59$	$s_{1,2} = -17.3681 \pm 17.4300j$	$-13.535 \pm 14.37j$, -6.3736, -1.2926
c)	$k_v = 1.4 \times 2.3755 \times 10^{-4}$, $\beta = 1.8 \times 0.0041$, $k_d = 5.62$, $k_p = 5.62$	$s_{1,2} = -20.2628 \pm 19.8676j$	$-15.5358 \pm 16.1280j$, -8.2292, -1.2247
d)	$k_v = 2 \times 2.3755 \times 10^{-4}$, $\beta = 2.8 \times 0.0041$, $k_d = 7$, $k_p = 7$	$s_{1,2} = -28.9468 \pm 21.6100j$	$-22.4899 \pm 14.2991j$, -11.7395, -1.1744

Table 15.3 State feedback controller gains

	Controller gains		Controller gains		Controller gains		Controller gains
a)	$k_1 = -0.0118$ $k_2 = -0.0118$ $k_3 = -0.2742$ $k_4 = -0.0298$	b)	$k_1 = -0.0263$ $k_2 = -0.0263$ $k_3 = -0.3962$ $k_4 = -0.0509$	c)	$k_1 = -0.0415$ $k_2 = -0.0415$ $k_3 = -0.5189$ $k_4 = -0.0728$	d)	$k_1 = -0.0804$ $k_2 = -0.0804$ $k_3 = -0.8269$ $k_4 = -0.1304$

$$= k_v y^{(3)} + \beta(k_p y + k_d \dot{y} + \ddot{y}),$$
$$= \beta k_p z_1 + \beta k_d z_2 + (\beta(a + bh) + \beta k_p h) z_3 + (k_v(a + bh) + \beta k_d h) z_4.$$

Comparing (15.21) and the last expression in (15.24), it is concluded that:

$$k_1 = -\beta k_p, \quad k_2 = -\beta k_d, \quad k_3 = -(\beta(a + bh) + \beta k_p h),$$
$$k_4 = -(k_v(a + bh) + \beta k_d h). \tag{15.25}$$

Using these relations and the numerical values in Table 15.1, the state feedback controller gains presented in Table 15.3a) are obtained. It has been also verified that the eigenvalues of matrix $A - BK$ are equal to the poles of (15.22) presented in Table 15.2a), as expected.

Some simulation results are presented in Fig. 15.7a (continuous line) when using the model (15.22), the controller gains in Table 15.2a), i.e., in Table 15.3a), and the numerical values in Table 15.1. All the initial conditions have been set to zero except for $y(0) = 1$[rad]. A time constant of about 1[s] is observed, which means

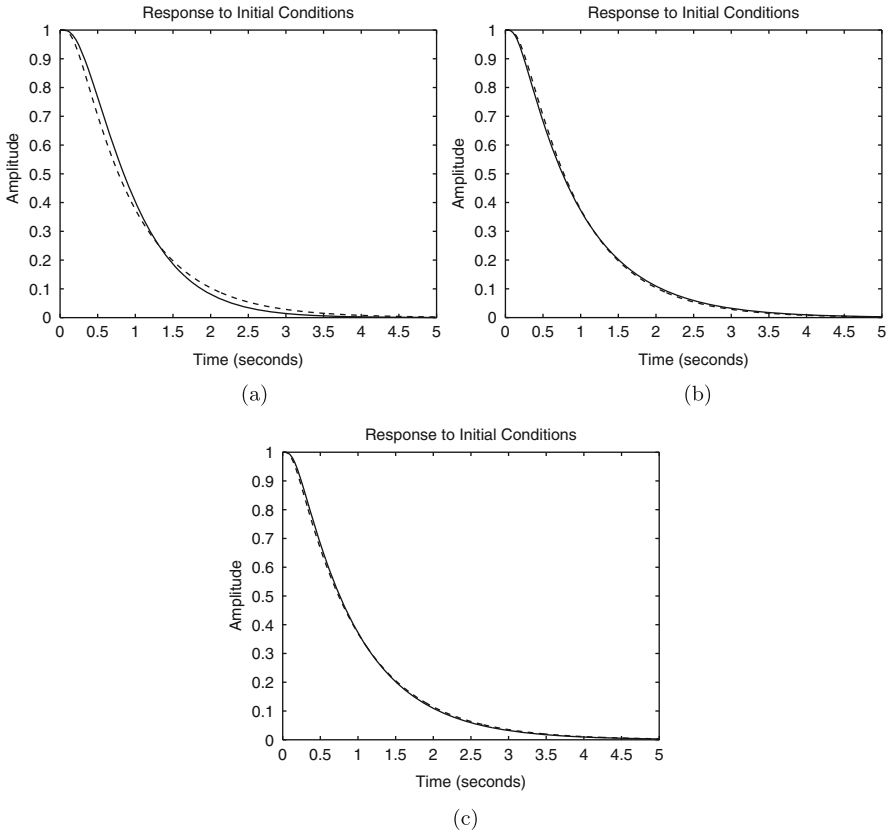

Fig. 15.7 Flat output y response obtained in simulation when using the controller (15.21) with gains in Table 15.3 . (**a**) Continuous, Table 15.3a. Dashed, Table 15.3b. (**b**) Dashed, Table 15.3b. Continuous, Table 15.3c. (**c**) Continuous, Table 15.3c. Dashed, Table 15.3d

that the transient response is dominated by the slowest closed loop pole located at $s = -1.8206$.

Some other simulation results are shown in Fig. 15.8 when using the controller in (15.10) to swing up the pendulum and the controller in (15.21) to stabilize the pendulum at the inverted configuration. The Furuta pendulum parameters that have been considered are those presented in Table 15.1. The controller gains for (15.21) are shown in Fig. 15.3a) whereas the controller gains for (15.10) are $K_E = 1900$, $k_\omega = 1$, $k_\delta = 3$, $k_\theta = 3$. All initial conditions were set to zero except for $\theta_1(0) = 2.9$[rad]. The switching condition between the controllers is the following:

if: $\sqrt{3(\theta_1 - \theta_{1d})^2 + 0.1\dot{\theta}_1^2} < 0.2$, then the controller in (15.21) is on,

else: controller in (15.10) is on.

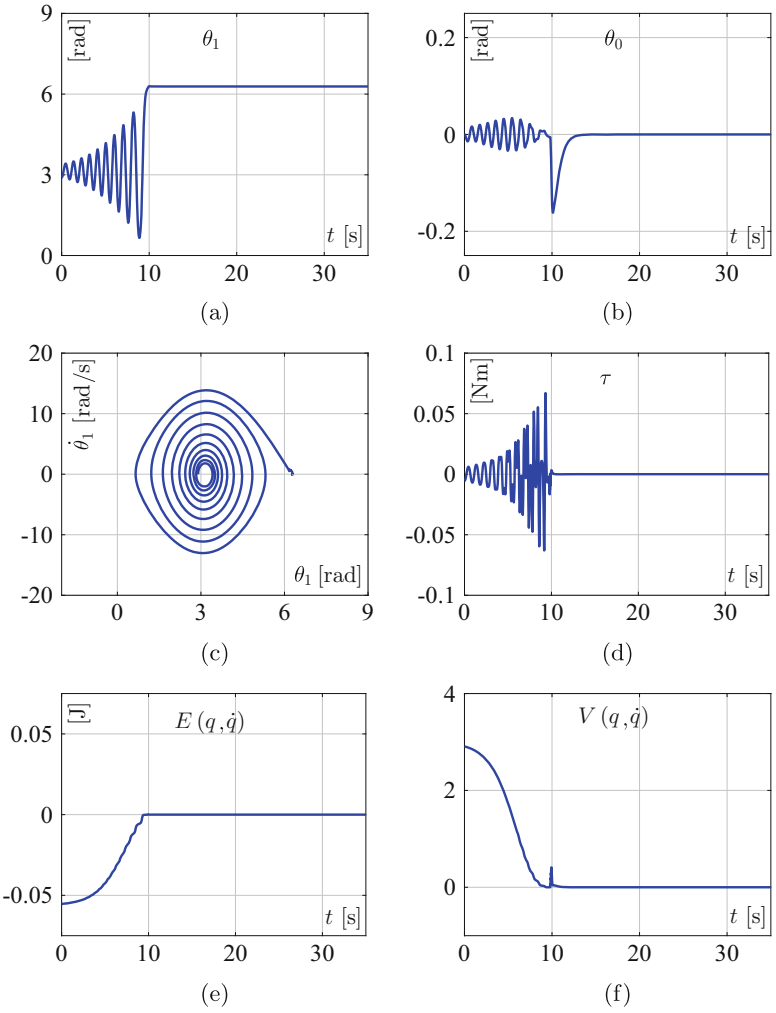

Fig. 15.8 Simulation results when swinging up and stabilizing the Furuta pendulum with the controllers in (15.10) and (15.21)

It is observed that the pendulum reaches the inverted unstable configuration whereas the arm position and the applied torque converge to zero, as desired. Also, the energy in (15.10) and the function V in (15.12) converge to zero. Finally, the phase plane plot in Fig. 15.8c clearly shows that the pendulum converges to the desired inverted configuration with zero velocity.

These simulations were performed using the MATLAB/Simulink diagram shown in Fig. 15.9 where all To work space blocks at the bottom save data in an array with sampling time -1. Four main blocks are shown that contain the code shown

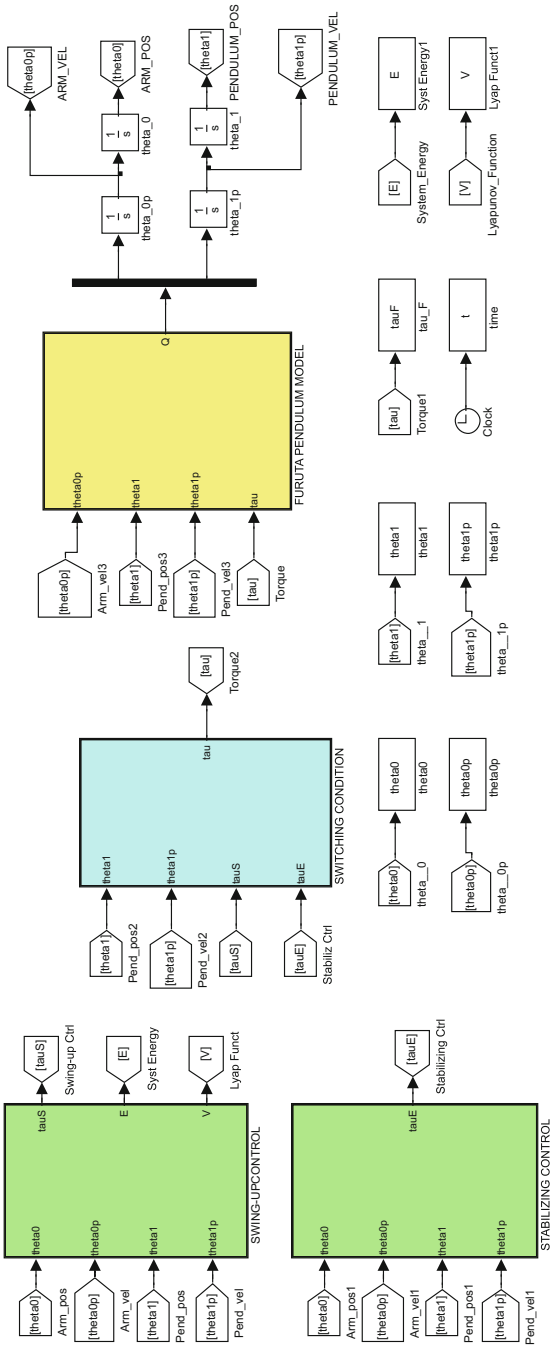

Fig. 15.9 MATLAB/Simulink diagram for simulations in Fig. 15.8

in Appendix G. Once the simulation in Fig. 15.9 is run, the MATLAB code in Sect. G.1.5 is executed in an m-file to obtain the plots in Fig. 15.8.

On the other hand, in Sect. 8.3, Chap. 8, it has been explained that a limit cycle is expected to appear in practice when using the above controller together with the controller gains in Table 15.2a). This requires the block diagram in Fig. 15.4 to be modified to that in Fig. 8.7 where a dead zone "static" nonlinearity is considered. Moreover, it has been shown that the closed-loop system can also be represented as in Fig. 8.8 where:

$$G(s) = \frac{-d(a+bh)(k_v s^3 + \beta s^2 + \beta k_d s + \beta k_p)}{s^2(s^2 - b)}, \quad -d(a+bh) > 0,$$

is defined. Furthermore, it is shown in Sect. 15.7 that a limit cycle actually appears in experiments. Recall that a dead zone nonlinearity is induced in servomechanisms because of static friction at the motor shaft. As is presented in Sect. 15.7, in these experiments it is observed that this limit cycle induces wide oscillations of the Furuta pendulum arm. This motivates new designs intended to reduce the amplitude of the oscillations. Moreover, the observed oscillations are also slow. This suggests that the controller must increase the stiffness of the closed-loop system. As stiffness is related to faster responses, this implies that the oscillation frequency must be increased.

According to Sect. 8.3, to obtain limit cycles with smaller amplitudes the polar plot of the equivalent open-loop transfer function $G(s)$ must intersect the negative real axis at a point σ located farther to the left. On the other hand, to increase the frequency of the oscillations, the phase of $G(j\omega)$ must reach -180° at a greater frequency $\omega = \omega_\sigma$.

The point σ can be placed farther to the left of the negative real axis by increasing the magnitude of $G(s)$. According to the previous expression, this requires the controller magnitude to be increased, i.e., to increase the controller gains k_v and β. On the other hand, according to Fig. 8.9, $G(j\omega)$ reaches a -180° phase at greater frequencies if the phase lead contributed by the controller is forced to appear at greater frequencies. As the cubic and the quadratic terms in $G(s)$ contribute larger phase leads at greater frequencies, we can force this phase lead to appear at greater frequencies by rendering the coefficients of the first-order term and the constant term larger than the coefficients of the cubic and the quadratic terms, i.e., by increasing k_d and k_p.

Based on these ideas, additional sets of controller gains to be tested experimentally are obtained in the following. Note, however, that these designs must ensure closed-loop stability. Thus, we proceed using the root locus method employing Fig. 15.6.

Our design strategy is to move the closed-loop poles farther to the left of the imaginary axis. This suggests moving the open-loop zero at $s = -f$ to the left. However, this may create two root locus branches starting at the origin but belonging to the right half-plane (see Fig. 15.6). This is because the zero at $s = -f$ may attract the open-loop complex conjugate poles at s_1, s_2, if it moves to the left. Hence,

instability or badly damped closed-loop poles may result if $f > 0$ is increased. Thus, it is proposed only to move the three fastest closed-loop poles to the left. First, propose larger values for k_v and β to render both the real and the imaginary parts of the open loop poles s_1 and s_2 in Fig. 15.6 larger (also see $s_{1,2}$ in Table 15.2). This renders the pendulum response faster and well damped,[1] and allows larger values for both k_d and k_p to be selected. This is because, although the root locus branches starting at s_1 and s_2 move toward the right half-plane as k_d increases, it is possible to select larger values for k_d before these branches are too close to the right half-plane if s_1 and s_2 move to the left. Note that k_p is larger if k_d is larger because $k_p = k_d f$.

This is the procedure that has been followed to find the controller gains in Table 15.2b), c) and d). Using these values and (15.25), the controller gains in Table 15.3b), c), and d), have been computed. Note that the three fastest closed loop poles move to the left, going from a) to d) in Table 15.2 or, equivalently, in Table 15.3. Finally, recall that it has been shown in Figs. 8.9 and 8.10 that the phase of $G(j\omega)$ reaches $-180°$ at a greater frequency and the plot of $G(s)$ intersects the negative real axis at a point σ, which moves to the left, going from a) to d) in Tables 15.2 and 15.3, i.e., in Table 8.1.

In Figs. 15.7a, b, and c the flat output responses y are compared when using the controller gains in Table 15.3a), b), c), and d). In these simulations, all the initial conditions have been set to zero except for $y(0) = 1$. We have also considered that the dead zone nonlinearity is not present. This has been done merely to verify that the response is slightly faster going from a) to d) in Table 15.3, as expected. However, this is true only at the beginning of the response. We stress that larger settling times are obtained because the slowest closed-loop pole in Tables 15.2b), c), and d), moves to the right.

Finally, recall that, according to Sect. 6.5.2, a time delay T is induced when implementing a continuous time controller with a digital computer. For the experiments that are presented next, $T = 0.01$[s] may be assumed to consider the worst case time delay as 0.01[s] is the sampling period. This time delay does not affect the magnitude of the open-loop frequency response, but contributes an additional phase lag given as $-\omega T$[rad]. This is because e^{-Ts} appears as a factor of the open-loop transfer function $G(s)$ plotted in Fig. 8.9. In that figure, for controller gains in Table 15.3a), the crossover frequency is $\omega = 29$[rad/s]. Hence, the additional phase lag is $-\omega T \times 180°/\pi = -16.6°$. Thus, the phase margin in Fig. 8.9 shifts from 58° to 41.39°, which is still a good phase margin. This means that the time delay induced in practice by the digital implementation of the controller does not have a significant effect when using the controller gains in Table 15.3a).

[1] Note, in Fig. 15.4, that k_v and β are the feedback gains of signals $\ddot{y}$ and $y^{(3)}$, which, according to (15.19), represent feedback of the pendulum position and velocity, $z_3 = \theta_1 - \theta_1^*$ and $z_4 = \dot{\theta}_1$.

15.7 Experimental Tests

In Fig. 15.10 the experimental results obtained when the controller (15.10) (also see [2], chapter 6) is used to swing up the pendulum and the controller (15.21) is used to stabilize the mechanism at $z = x - x^* = 0$ are presented. The numerical parameters $K_E = 480$, $k_\omega = 1$, $k_\delta = 3$, $k_\theta = 17$, were employed for controller (15.10) whereas the gains in Table 15.3a) were used for controller (15.21). Note that the desired pendulum position $x_3^* = \theta_1^* = 2\pi$ is reached and the pendulum remains there for $t \geq 8.8$[s]. However, the arm position does not reach the desired position $x_1^* = \theta_0^* = 0$ but oscillates around a nonzero arm position. To better observe this behavior and to better compare with cases when different controller gains are employed, in Fig. 15.11 a zoom-in on the variables θ_0 and $\theta_1 - 2\pi$ is presented. Moreover, the time axis has been modified to consider $t = 0$ at the time when the stabilizing controller (15.21) is switched on, i.e., when $t = 8.8$[s] in Fig. 15.10. Hence, it is concluded that the pendulum really remains at $\theta_1 = 2\pi$ whereas the arm describes large peak-to-peak oscillations of about 0.6[rad] around a constant position of about -0.8[rad].

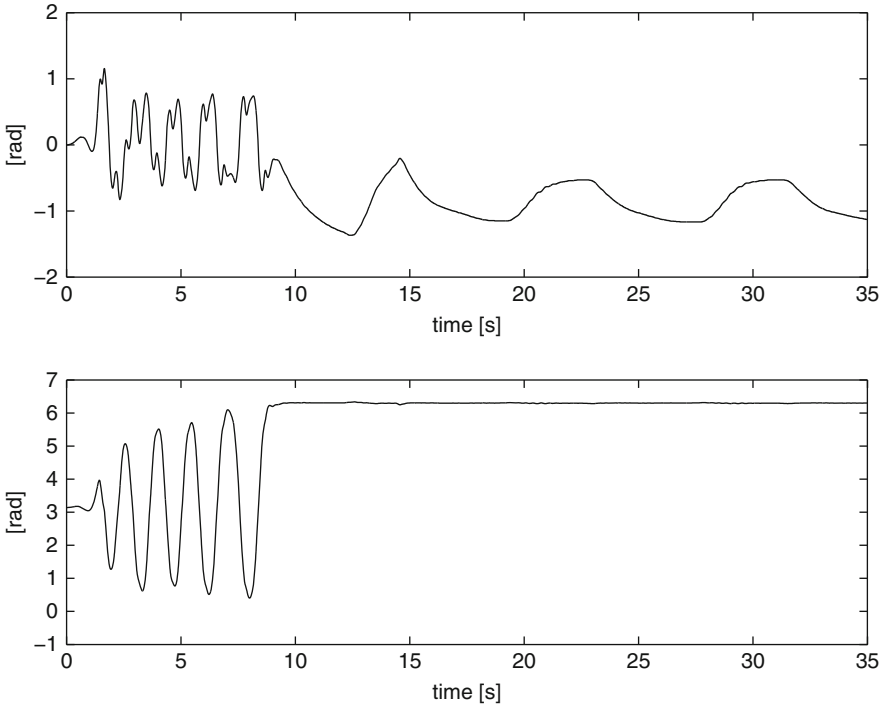

Fig. 15.10 Experimental results when using the controller gains in Table 15.3a). Top line: θ_0[rad]. Bottom line: θ_1[rad]

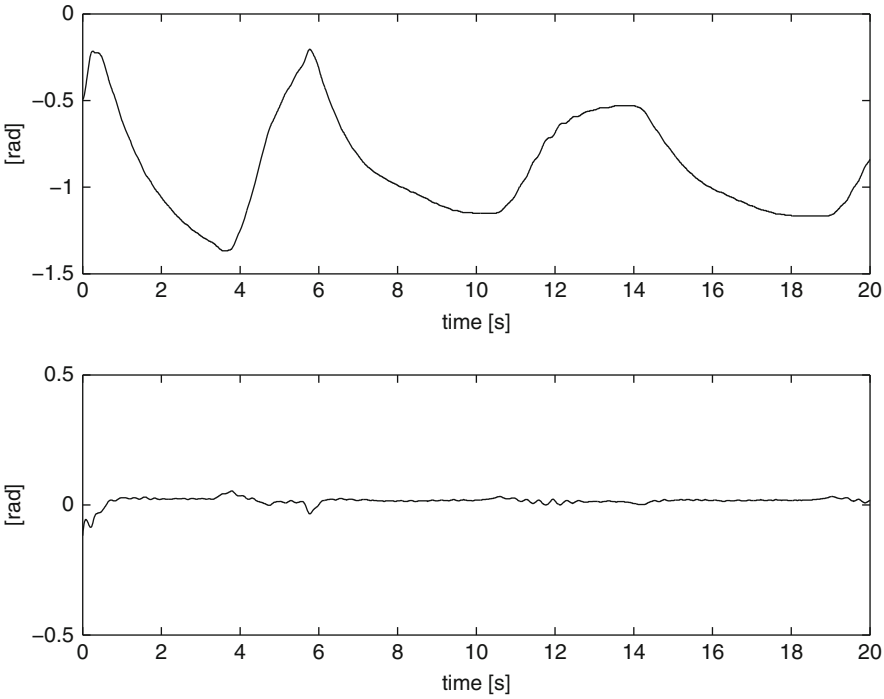

Fig. 15.11 Experimental results when using the controller gains in Table 15.3a). Top line: θ_0[rad]. Bottom line: $\theta_1 - 2\pi$[rad]

In Fig. 15.12 the flat output $y = z_1 + hz_3$ response[2] is presented, in addition to the response of its components z_1 and hz_3. Note that the time axis in this figure is the same as in Fig. 15.11. It is observed that y and z_1 remain close to each other while oscillating because hz_3 is close to zero. These observed oscillations represent a limit cycle that has been predicted to exist because of a dead zone induced by static friction at the motor shaft. Furthermore, the static friction is also responsible for the large constant position value of -0.8[rad] around which arm oscillations are performed. We wonder, however, why oscillations are not present in the pendulum position. To explain this, consider (15.19) where:

$$y = z_1 + hz_3, \quad \ddot{y} = (a + bh)z_3,$$

and assume that the flat output oscillation is sinusoidal, i.e., $y = Y_0 \sin(\omega t)$ where Y_0 is a positive constant representing the oscillation amplitude. It is not difficult to find that, under these conditions:

[2]Recall that $z_1 = \theta_0$ and $z_3 = \theta_1 - 2\pi$ in this case.

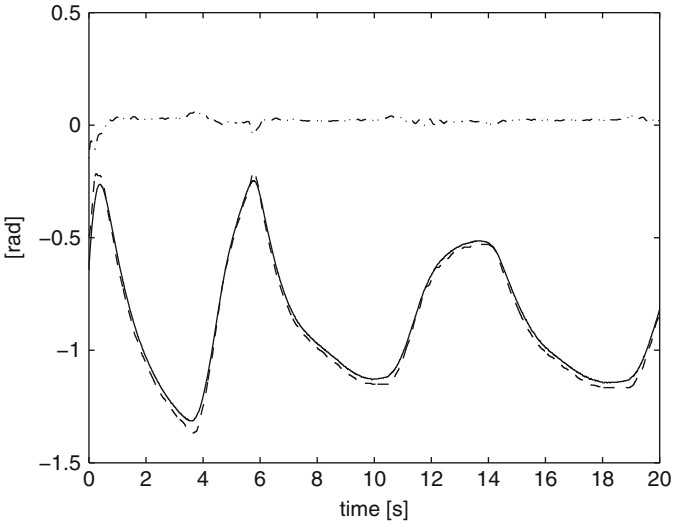

Fig. 15.12 Experimental results when using the controller gains in Table 15.3a). Continuous: y[rad]. Dashed: z_1[rad]. DashâĂŤdot: hz_3[rad]

$$z_1 = Y_0\left(1 + \frac{h\omega^2}{a + bh}\right)\sin(\omega t), \quad z_3 = -Y_0\frac{\omega^2}{a + bh}\sin(\omega t). \tag{15.26}$$

This means that z_3 is very small if the oscillation frequency is small and, hence, $y \approx z_1$. Note that, in Fig. 15.12, the oscillation frequency is, approximately, $\omega = \frac{2\pi}{8.5} = 0.7392$[rad/s], which yields $\frac{\omega^2}{a+bh} = 0.0086$. This explains why oscillations are not observed in the pendulum position and why z_1 remains close to y in the above experiments.

Finally, in Fig. 15.13 the phase plot of the pendulum variables during the swinging-up stage of the experiment is presented. Note that the experiment begins at $(\theta_1, \dot{\theta}_1) = (\pi, 0)$ and finishes at $(\theta_1, \dot{\theta}_1) = (2\pi, 0)$. The theoretical trajectory described in Fig. 15.3 is also presented. As explained in Sect. 15.2, the controller in (15.10) is intended to force the pendulum variables to converge to the trajectory in Fig. 15.3. As observed in Fig. 15.13, this is almost accomplished and, because of that, the pendulum swings up successfully. Note, however, that convergence to the trajectory in Fig. 15.3 is not accomplished asymptotically because of the imperfections in control system implementation. Such system implementation imperfections involve the fact that the power electronics stage in our experimental prototype is not capable of supplying the electric current required by the controller (15.10) to the DC motor used as actuator. Moreover, it is important to say that this has forced us to multiply by a factor of 5.8 the torque computed with the controller in (15.10)

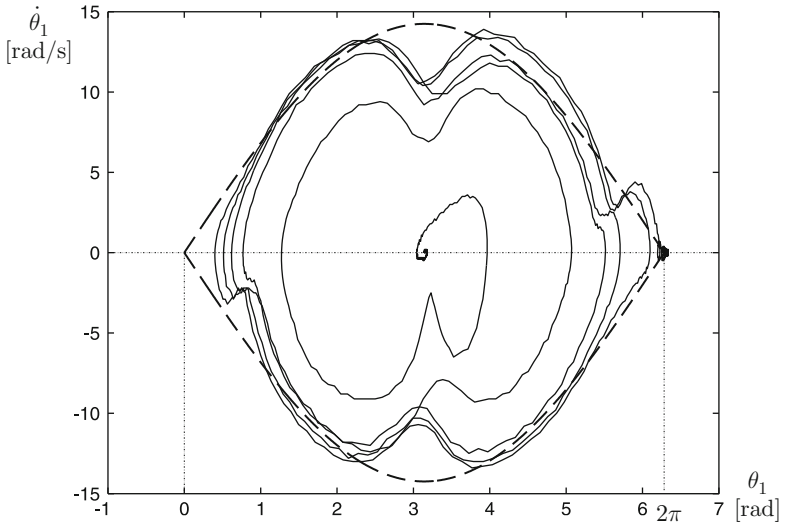

Fig. 15.13 Phase plot of the pendulum variables during the swinging-up stage of the experiment (continuous). The trajectory presented in Fig. 15.3 is also shown here (dashed)

to obtain the value of torque to be commanded to the DC motor actuator.[3] This is explicitly expressed in the Builder 6 C++ program code presented in Sect. 15.10. The use of the factor 5.8 was chosen empirically after several tests.

In Figs. 15.14, 15.15, 15.16 the experimental results obtained when using the controller gains in Table 15.3b) are presented. In Figs. 15.17, 15.18, 15.19 the experimental results obtained when using the controller gains in Table 15.3c) are presented. These results are plotted on identical axes as in Figs. 15.10, 15.11, 15.12 to render the comparison between the results obtained when different controller gains are used fair. $t = 0$ is set in Figs. 15.15 and 15.16 at the point of time in Fig. 15.14 where $t = 13.8$[s]. Also, $t = 0$ is set in Figs. 15.18 and 15.19 at the point of time in Fig. 15.17 where $t = 12.9$[s].

In all the above results it is observed that the pendulum position is successfully stabilized at $\theta_1 = 2\pi$. When the controller gains in Table 15.3b) are used, the following is concluded. Although the arm position still oscillates, the amplitude is smaller and the frequency $\omega = 2\pi/5 = 1.2566$[rad/s] is greater than values in Figs. 15.10, 15.11, 15.12. Note that these results verify the theoretical predictions that have motivated the redesign of the controller gains. It is also observed that the oscillations are performed around a constant arm position of about -0.35[rad], which is smaller than in the case of the controller gains in Table 15.3a). Note that

[3]Torque computed by the stabilizing controller (15.21) is not multiplied by any factor different from unity.

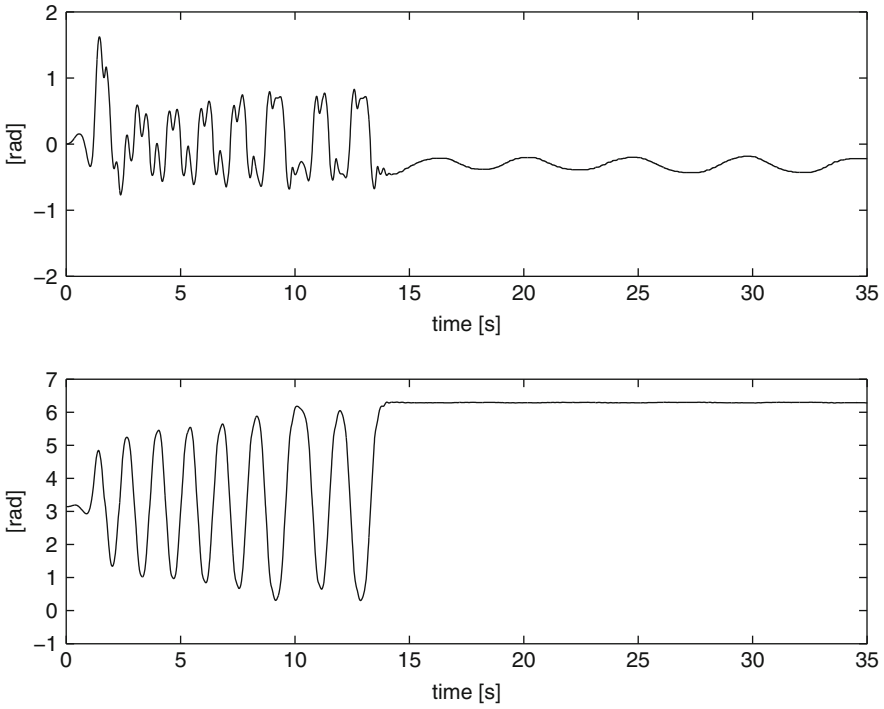

Fig. 15.14 Experimental results when using the controller gains in Table 15.3b). Top line: θ_0[rad]. Bottom line: θ_1[rad]

the above values yield $\frac{\omega^2}{a+bh} = 0.0250$ which explains, again, why oscillations do not appear in the pendulum position.

The results obtained when using the controller gains in Table 15.3c) show that the limit cycle disappears. As can be seen in Figs. 15.17, 15.18, 15.19, the oscillation amplitude in both the flat output y and the arm position θ_0 are even smaller, whereas the oscillation frequency has increased to about $\omega = 2\pi/2.3 = 2.7318$[rad/s]. Furthermore, both these variables reach constant values, i.e., oscillation disappears, for $t \geq 7$[s] in Figs. 15.18 and 15.19. Note that the pendulum position reaches its desired constant value. The constant steady-state error in the arm position is about 0.2[rad], i.e., even smaller than the mean error of -0.35[rad] for controller gains in Table 15.3b).

Note the following. The small threshold δ in a dead zone nonlinearity (see Sect. 8.3) is uncertain and changes during normal operation because static friction also has these properties. As no movement is produced when the generated torque is smaller than δ (or the applied voltage is small enough), we may wonder whether limit cycles might be avoided by forcing them to appear only at a very small amplitudes. Hence, this may explain why limit cycle has disappeared.

It is important to say, however, that limit cycle avoidance is not achieved every time an experiment is performed with the controller gains in Table 15.3c).

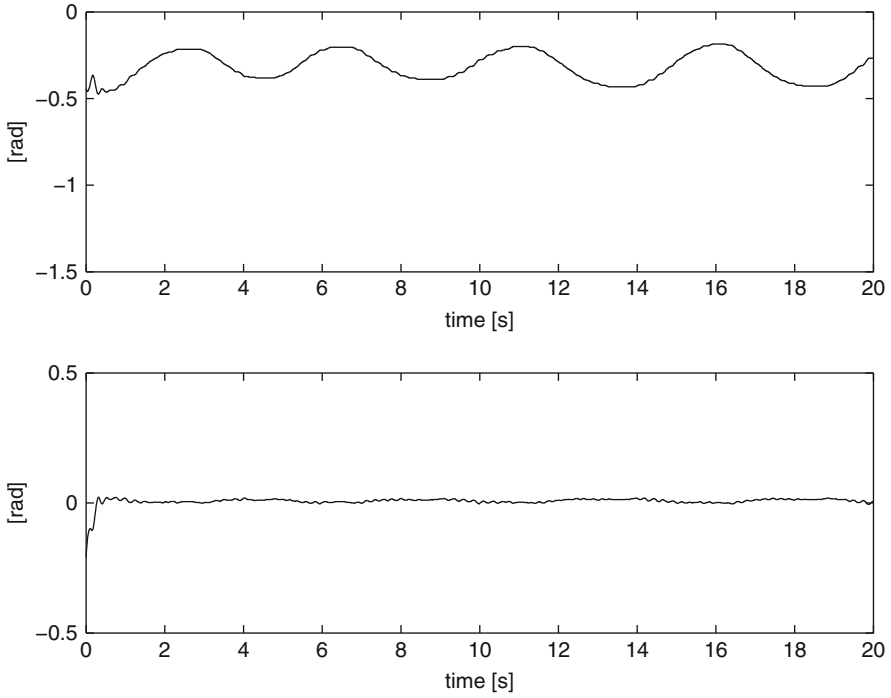

Fig. 15.15 Experimental results when using the controller gains in Table 15.3b). Top line: θ_0[rad]. Bottom line: $\theta_1 - 2\pi$[rad]

Sometimes very small and very slow oscillations are observed on both z_1 and y whereas z_3 remains at zero with no oscillations. Thus, limit cycle avoidance appears as a random event, perhaps because, as stated earlier, the small threshold δ is uncertain and changes during normal operation.

In Figs. 15.20, 15.21, 15.22, the experimental results obtained when using the controller gains in Table 15.3d) are presented. $t = 0$ is set in Figs. 15.21 and 15.22 at the point of time in Fig. 15.20 where $t = 3.95$[s]. Note that, again, a limit cycle is not present, i.e., the pendulum and arm positions in addition to the flat output, reach constant values in a steady state after the stabilizing controller (15.21) is switched on. It is also observed that the pendulum reaches the desired position $\theta_1^* = 2\pi$ and the arm remains closer to its desired position, $\theta_0^* = 0$, than in the case where the controller gains in Table 15.3c) were used. Note that the oscillation frequency is difficult to measure in this case, but comparing the settling times in Figs. 15.21 and 15.18 we conclude that the frequency is greater for the controller gains in Table 15.3d).

Recall that, according to theoretical arguments, the controller gains in Table 15.3d) are expected to result in even smaller amplitude oscillations than when the controller gains in Table 15.3c) are used. On the other hand, it has been

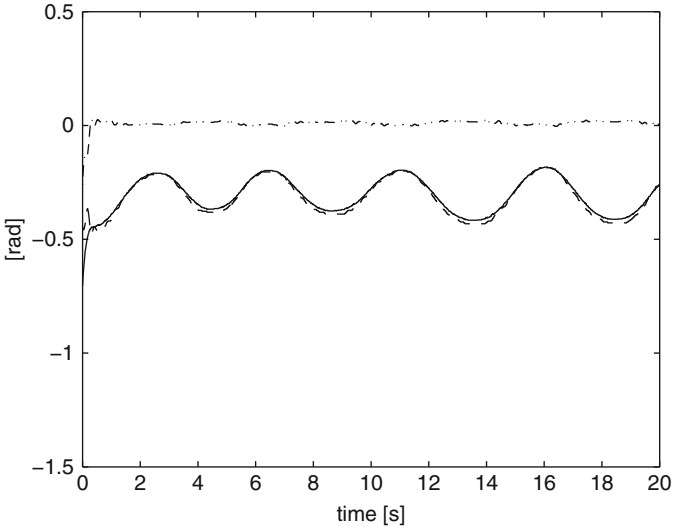

Fig. 15.16 Experimental results when using the controller gains in Table 15.3b). Continuous: y[rad]. Dashed: z_1[rad]. Dashdot: hz_3[rad]

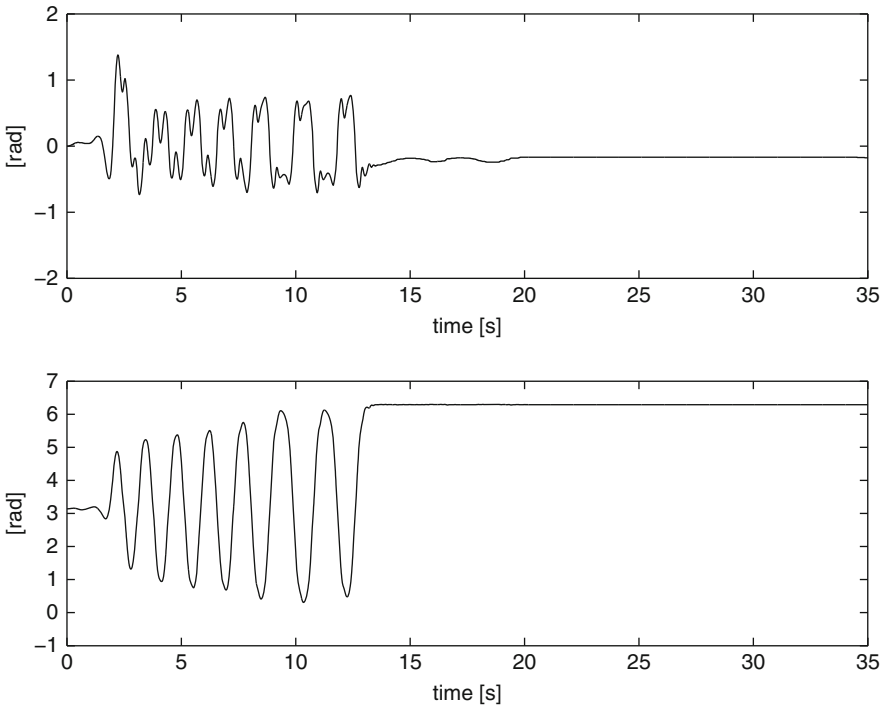

Fig. 15.17 Experimental results when using the controller gains in Table 15.3c). Top line: θ_0[rad]. Bottom line: θ_1[rad]

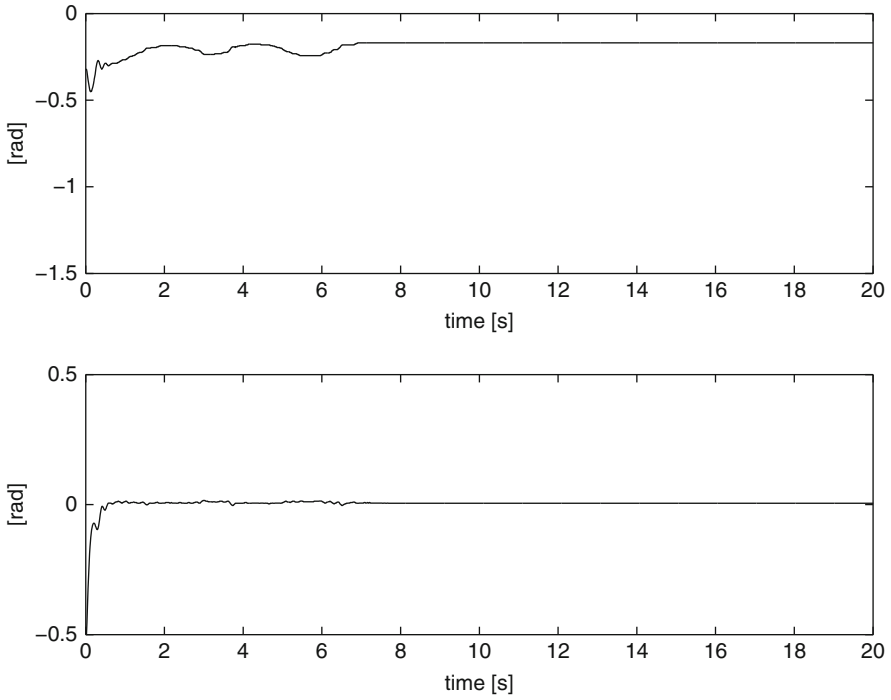

Fig. 15.18 Experimental results when using the controller gains in Table 15.3c). Top line: θ_0[rad]. Bottom line: $\theta_1 - 2\pi$[rad]

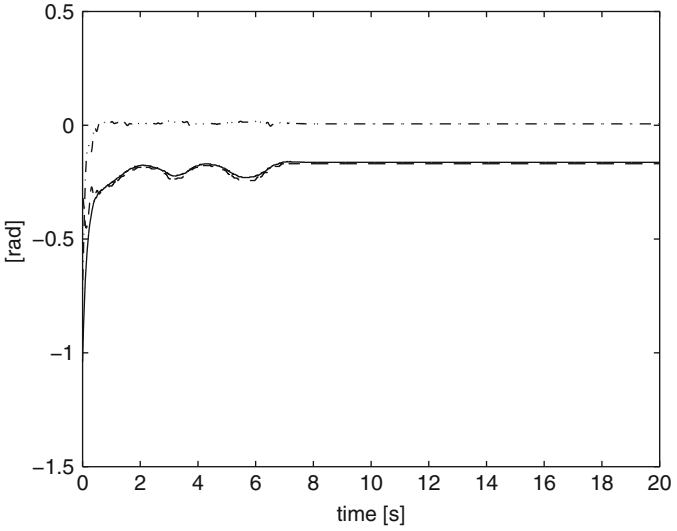

Fig. 15.19 Experimental results when using the controller gains in Table 15.3c). Continuous: y[rad]. Dashed: z_1[rad]. Dash–dot: hz_3[rad]

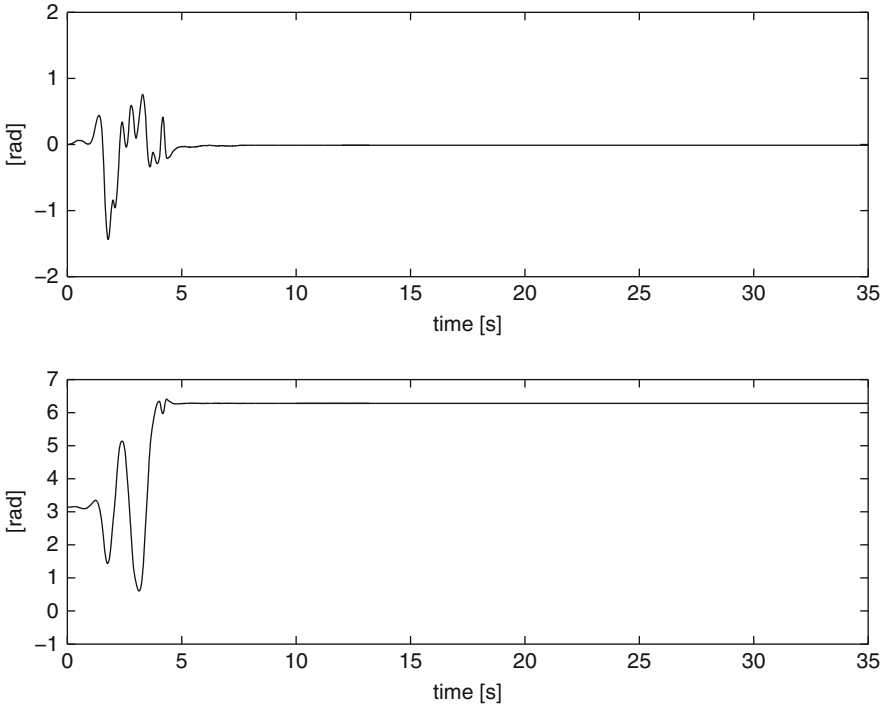

Fig. 15.20 Experimental results when using the controller gains in Table 15.3d). Top line: θ_0[rad]. Bottom line: θ_1[rad]

previously explained that forcing, by design, limit cycles to appear only for small amplitudes may result in limit cycle avoidance. This, again, explains why a limit cycle is not present for when the controller gains in Table 15.3d) are used.

The Furuta pendulum prototype used to perform all the above experiments is equipped with an internal electric current loop, driven by a linear proportional–integral (PI) controller, whose task is to force the electric current i flowing through the DC motor used as an actuator to reach a desired value i_d. This desired value is computed as $i_d = \tau_d / k_m$, where k_m is the torque constant of the motor and τ_d is torque computed by either (15.10) or (15.21). As the motor generates an electromagnetic torque given as $\tau = k_m i$, if i is equal to i_d, then $\tau = \tau_d$, which means that the motor generates exactly the torque required by both the swinging-up and the stabilizing controllers. Achieving $i = i_d$ is the task of the linear PI electric current controller cited above, which requires careful design of the gains k_{pi} and k_{ii} for this PI controller. In all the above experiments we have used $k_{pi} = 0.8$ and $k_{ii} = 130$.

In the following, an example is presented to show how the controller gains k_{pi} and k_{ii} may affect the closed-loop system performance. In Figs. 15.23, 15.24, 15.25 some experimental results are presented when using the controller gains in

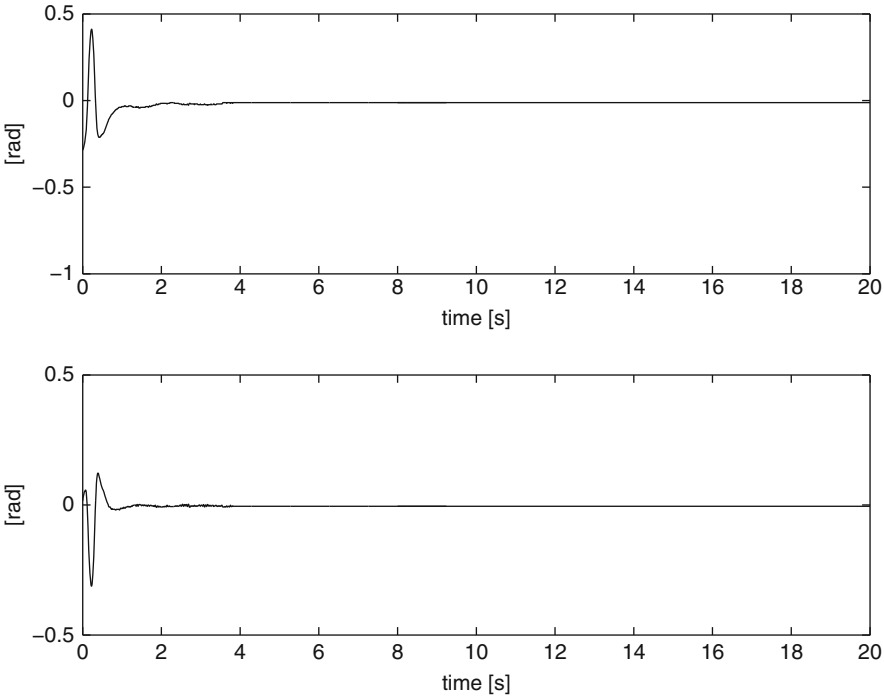

Fig. 15.21 Experimental results when using the controller gains in Table 15.3d). Top line: θ_0[rad]. Bottom line: $\theta_1 - 2\pi$[rad]

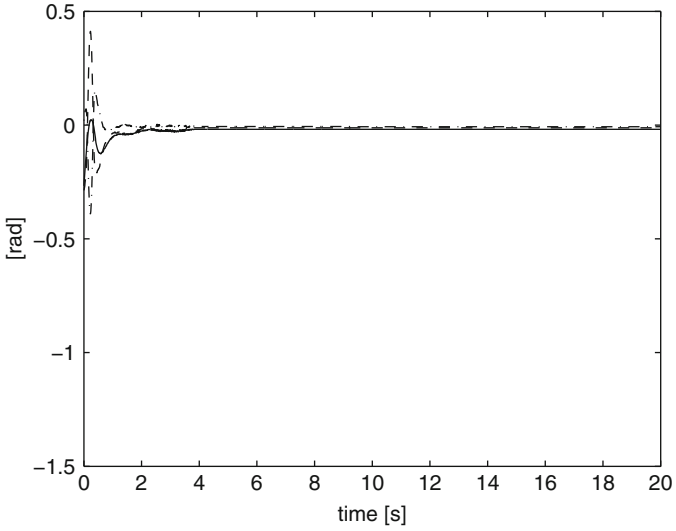

Fig. 15.22 Experimental results when using the controller gains in Table 15.3d). Continuous: y[rad]. Dashed: z_1[rad]. Dash-dot: hz_3[rad]

Table 15.3d) and $k_{pi} = 0.03$ and $k_{ii} = 130$. $t = 0$ is set in Figs. 15.24 and 15.25 at the point of time in Fig. 15.23 where $t = 9$[s]. Note that all three variables, the pendulum and the arm positions in addition to the flat output, are oscillatory the whole time that the stabilizing controller (15.21) is switched on. Despite this, the pendulum remains around the inverted position $\theta_1^* = 0$ and the arm remains close to its desired position $\theta_0^* = 0$. Note that the oscillation frequency is about $\omega = 2\pi/0.15 = 41.8879$[rad/s], which yields $\frac{\omega^2}{a+bh} = 27.7230 \gg 1$. This and (15.26) explain why the amplitude of oscillations in the pendulum position z_3, or hz_3, is much larger that of the oscillation in the flat output y; hence, the oscillation in the arm position z_1 has the same amplitude as that in hz_3. Moreover, it is clear from Fig. 15.25 that the oscillation in hz_3 is 180° out of phase with respect to the oscillation in z_1. This is also explained by (15.26). Finally, note, from Fig. 15.25, that the settling time of y is very close to that in Fig. 15.7c, which has been obtained through simulations.

Despite these spectacular experimental corroborations of theoretical predictions, we wonder why a limit cycle exists. Recall that according to theoretical arguments, the controller gains in Table 15.3d) are expected to result in even smaller amplitude oscillations than when the controller gains in Table 15.3c) are used. Thus, the reader may wonder why a clear limit cycle appears again for the gains in Table 15.3d) despite limit cycles being avoided for the gains in Table 15.3c).

In Fig. 15.26, the measured electric current through the motor i and the desired current i_d computed as $i_d = \tau_d/k_m$ are presented. Note that i has a phase lag of 90° with respect to i_d for $t \geq 9$[s], i.e., for the whole time when the linear stabilizing controller (15.21) is switched on. According to Fig. 8.9, an additional phase lag of 90° appearing at any place in the loop of Fig. 8.8 is enough to produce closed-loop instability. According to Fig. 15.26 such a phase lag is contributed by the DC motor electric dynamics, which has been neglected during the design stage. Moreover, it can be observed in Fig. 15.26 that the amplitude of i increases until a maximum amplitude is reached. This shows instability despite the fact that the electric current amplitude does not grow to infinity because, as we have explained before, it saturates, as the power electronics stage of our prototype can only supply a limited maximal electric current.

In Sect. 8.3.4 a limit cycle study is presented when a saturation nonlinearity is present and the electric dynamics of the DC motor used as actuator is taken into account. There, the corresponding closed-loop block diagram is that shown in Fig. 8.15. The plant model and the parameters considered in that study correspond to the Furuta pendulum with parameters in Table 15.1 and the controller gains considered in this part of the present chapter, i.e.,:

$$k_v = 2 \times 2.3755 \times 10^{-4}, \quad \beta = 2.8 \times 0.0041, \quad k_d = 7, \quad k_p = 7,$$

Moreover, in Sect. 8.3.4, the following motor parameters are also considered:

$$k_m = k_b = 0.0368, \quad R = 2.4[\text{Ohm}], \quad k_{ii} = 130 \quad L = 0.03[\text{H}],$$

which correspond to the DC motor actually used in the Furuta pendulum prototype that has been employed to perform the experiments reported in the present chapter.

In Sect. 8.3.4 it is found that a stable limit cycle exists when $k_{pi} = 0.03$. Moreover, the closed-loop system is unstable but the limit cycle converts such instability into a sustained oscillation, i.e., a stable limit cycle. This limit cycle is predicted to appear at a frequency $\omega \approx 48$[rad/s], i.e., approximately the same frequency measured in Figs. 15.23, 15.24, 15.25. Thus, these experimental results are formally explained by the study on limit cycles performed in Sect. 8.3.4.

To solve this problem k_{pi} has been increased from 0.03 to $k_{pi} = 0.8$ and $k_{ii} = 130$ is kept. The reason for this is that k_{pi} contributes with an open-loop zero located at $s = -k_{ii}/k_{pi}$. Hence, if k_{pi} is larger, this open-loop zero increases the phase lead for any given frequency, thus reducing the phase lag contributed by the DC motor electric dynamics and restoring closed-loop stability.

In Sect. 8.3.4, it is also found that any stable limit cycle due to saturation does not exist when $k_{pi} = 0.8$. Moreover, the closed-loop system is stable in this case. Thus, the selection of $k_{pi} = 0.8$ is formally justified by the study on limit cycles performed in Sect. 8.3.4.

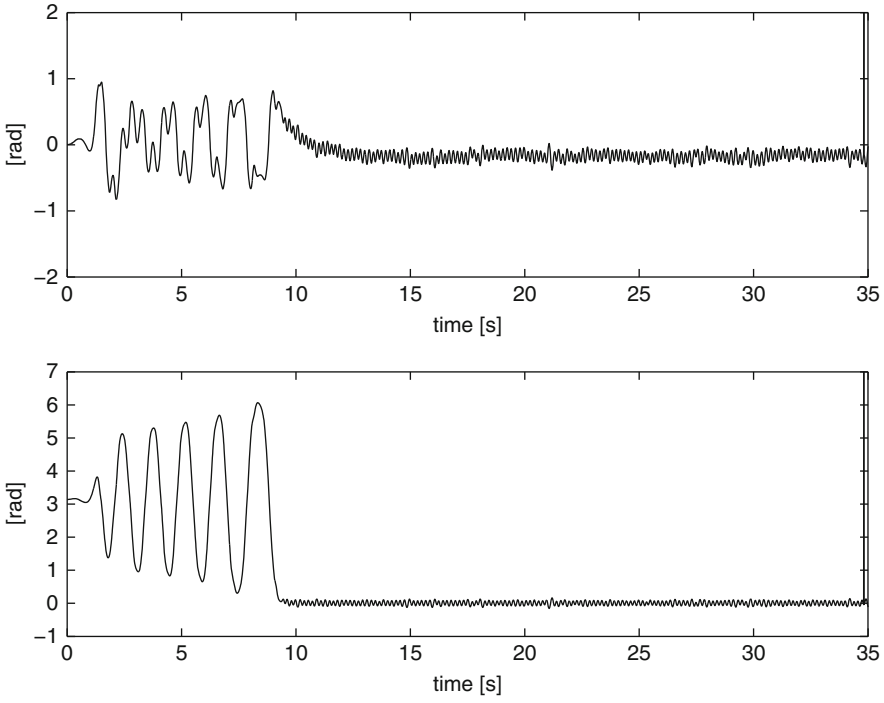

Fig. 15.23 Experimental results when using the controller gains in Table 15.3d). Top line: θ_0[rad]. Bottom line: θ_1[rad]

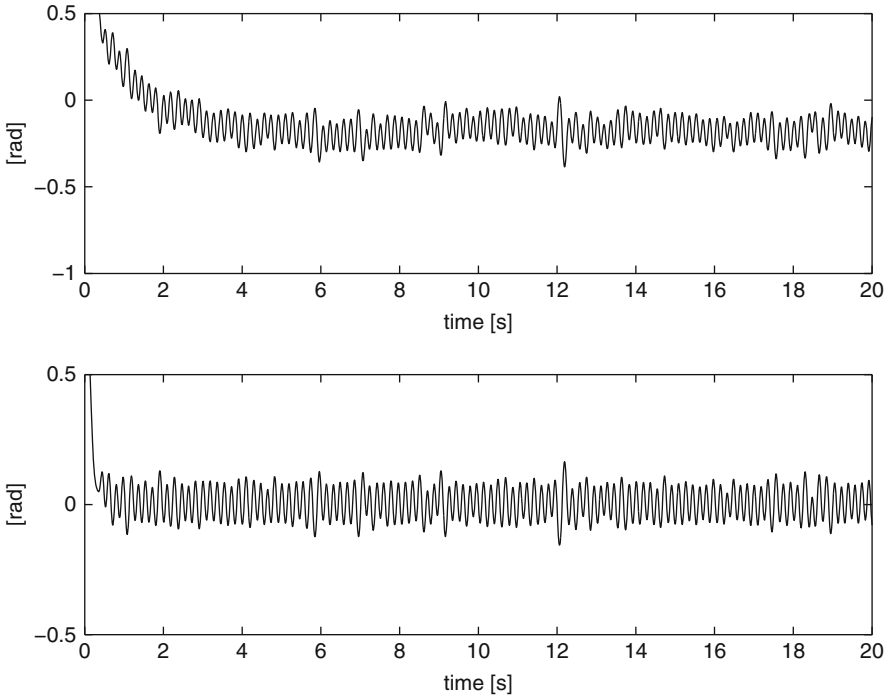

Fig. 15.24 Experimental results when using the controller gains in Table 15.3d). Top line: θ_0[rad]. Bottom line: θ_1[rad]

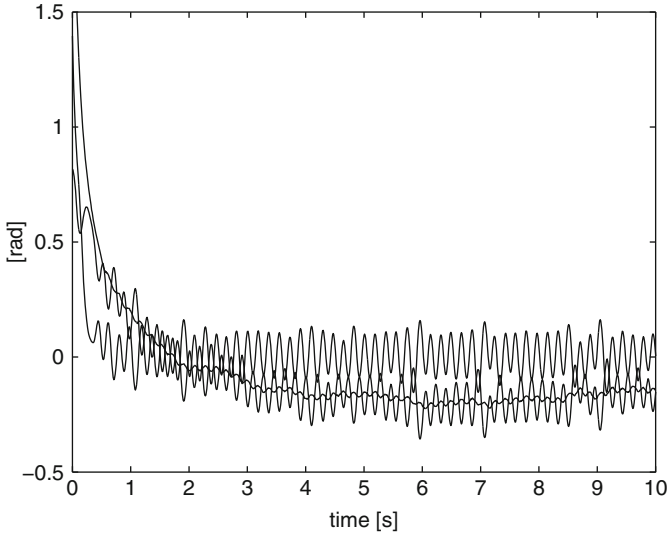

Fig. 15.25 Experimental results when using the controller gains in Table 15.3d). Smallest amplitude oscillatory line: y[rad]. Lower oscillatory line: z_1[rad]. Upper oscillatory line: hz_3[rad]

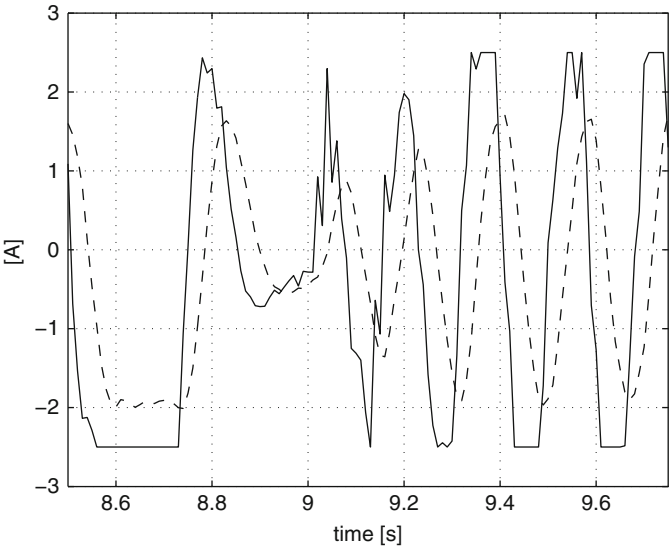

Fig. 15.26 Experimental results when using the controller gains in Table 15.3d) and $k_{pi} = 0.03$ and $k_{ii} = 130$. Continuous: i_d[A]. Dashed: i[A]

Finally, let us say that in a first series of experiments $k_{pi} = 0.03$ and $k_{ii} = 130$ have been used. Although we have found good results for the controller gains in Table 15.3a),b),c), we have found the problem described in Figs. 15.23, 15.24, 15.25 for the controller gains in Table 15.3d). Once the problem was understood, the PI electric current controller gains have been adjusted to $k_{pi} = 0.8$ and $k_{ii} = 130$ and all the experiments were repeated using these gains. These final experimental tests are those that we have presented above.

15.8 Control System Construction

A Furuta pendulum has been built whose numerical parameters are shown in Table 15.1. A picture of this Furuta pendulum is presented in Fig. 15.27 and the electric diagram of the complete control system is depicted in Fig. 15.28. The control algorithm is implemented in a Builder 6 C++ program, which is executed by a portable computer. The portable computer receives the mechanism output data from a PIC16F877A microcontroller [7] and sends the mechanism input data back to the microcontroller. Then, the microcontroller sends the corresponding voltage signal to the motor. The Builder 6 C++ program executed by the portable computer and the C program executed by the microcontroller are listed in Sects. 15.10 and 15.11 respectively.

Fig. 15.27 The experimental prototype working

The L298 integrated circuit contains two full-bridge drivers that are parallel connected to be used as a power amplifier. The power signal is applied on a PM brushed DC motor model Tohoko Ricoh 7K00011, which is employed as the mechanism actuator. This motor is provided with a 400 pulses/revolution encoder, which is used to measure the arm position. The pendulum position is measured by a 1000 pulses/revolution encoder model S1-1000-250-I-B-D from USDigital (the bottom encoder in Fig. 15.28). Communication between the portable computer and the microcontroller is rendered possible by a MAX232. Finally, three AND logic gates manage the enable bits for the L298 full-bridge. The complete control system is put to work through SWITCH_1.

The mechanism is provided with an electric current loop with a PI controller driven by the difference $i_d - i$, where i is the electric current through the motor armature and i_d is the desired current. The desired current is computed as $i_d = \tau / k_m$, where τ is obtained from either (15.10) or (15.21) and $k_m = 0.0368$[Nm/A]. This ensures that $i \approx i_d$ for all time; hence, torque generated by the motor is approximately equal to the desired torque given by either (15.10) or (15.21). The electric current through the motor armature i is converted to a voltage signal by a $1[\Omega]$, 5[W], power resistance, which is series connected to the motor armature terminals. This voltage is measured by the differential voltage amplifier located at the left upper corner in Fig. 15.28. After that, this signal is suitably amplified and an adder amplifier is used to center the zero current value at a $+2.5$[V] level. Then, the measured current is low-pass filtered and sent to a 10-bit analog/digital converter in the microcontroller whose input analog range is $[0, +5]$[V]. This allows the electric current to be measured in the range $[-2.5, +2.5]$[A]. Recall that the $1[\Omega]$ power

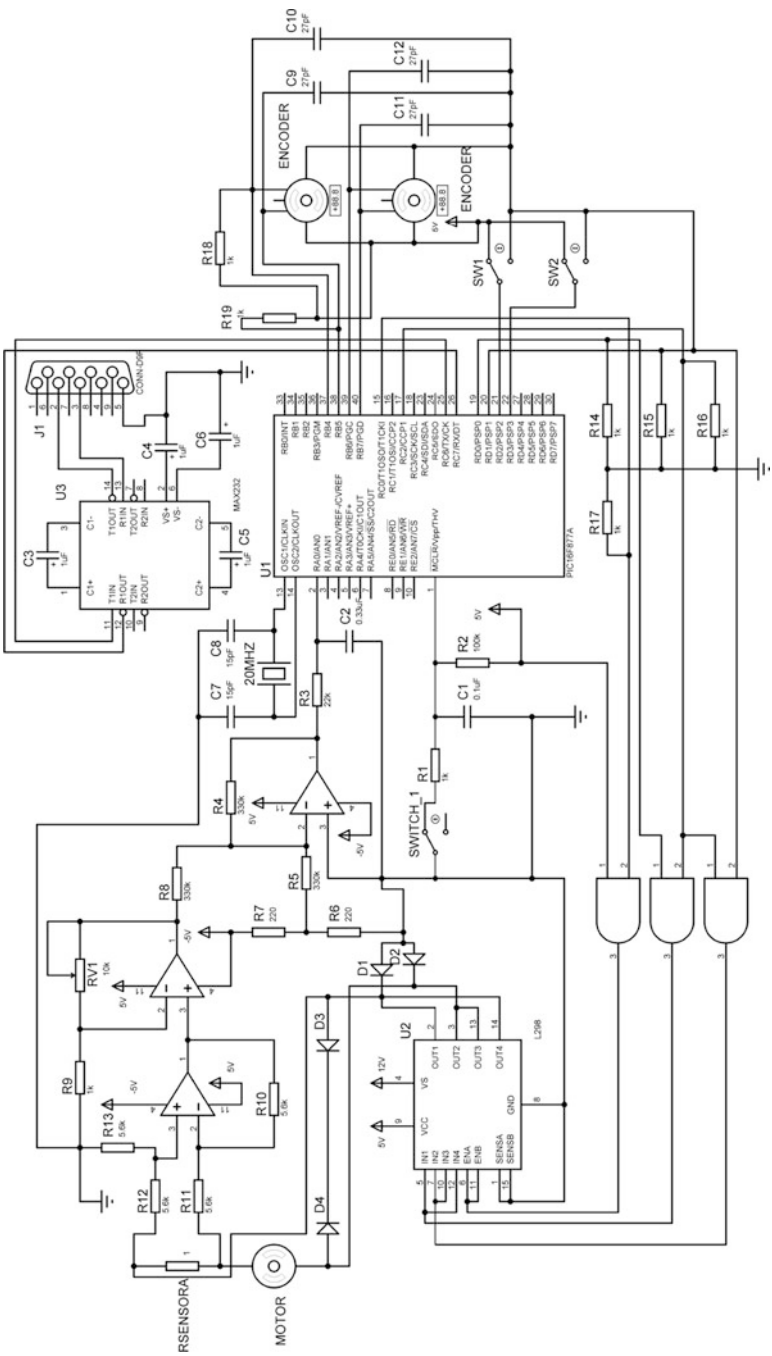

Fig. 15.28 Electric diagram of the Furuta pendulum control system

resistance converts the electric current into a voltage signal whose numerical value is equal to the electric current in amperes.

Some of the results presented in this chapter were reported for the first time in [6].

15.9 Sampling Period Selection

According to Sect. 8.4, Chap. 8, the selection of a suitable available bandwidth Ω_a is an important step when designing a controller for an unstable plant. Recall that, according to (15.20), the Furuta pendulum is an unstable plant with a single, real, unstable pole at $s = \sqrt{b} > 0$, where:

$$b = \frac{(I_0 + m_1 L_0^2) m_1 l_1 g}{I_0(J_1 + m_1 l_1^2) + J_1 m_1 L_0^2}. \tag{15.27}$$

According to (8.31), if a constant minimal sensitivity S_{min} is achieved for all $\omega \in [0, \Omega_a]$, then the relation between S_{min}, Ω_a and the unstable pole is given as:

$$S_{min} = \mathrm{e}^{\frac{\pi}{\Omega_a}\sqrt{b}} > 1. \tag{15.28}$$

As explained in [5], the most severe limitation of the size of the available bandwidth is the sampling period T_s. It is also suggested in [5] that the sampling frequency ω_s might be selected such that:

$$\frac{\omega_s}{\Omega_a} = 16, \tag{15.29}$$

because this allows sampling two or three times in one radian.

On the other hand, if it is assumed that a specific S_{min} is achieved for all $\omega \in [0, \Omega_a]$, the corresponding phase margin K_f is obtained as follows. According to Sect. 8.4, the distance from the polar plot of $G(j\omega)H(j\omega)$ to the critical point $(-1, j0)$ is computed as $r = 1/S_{min}$. Hence, define a circle $y^2 + (x+1)^2 = r^2$ as the plot of $G(j\omega)H(j\omega)$ whose distance to $(-1, j0)$ is constant and equal to $r = 1/S_{min}$. To correctly measure the phase margin, we need to find the point where this circle intersects the circle $y^2 + x^2 = 1$, i.e., where $|G(j\omega)H(j\omega)| = 1$. Simple algebra shows that the intersection point is given as:

$$x = \frac{r^2 - 2}{2}, \quad y = -\sqrt{1 - x^2}, \tag{15.30}$$

The phase margin is then computed as:

Table 15.4 Effect of sampling period on the achievable phase margin

T_s[s]	$\omega_s = \frac{2\pi}{T_s}$[rad/s]	Ω_a[rad/s]	S_{min}	$r = \frac{1}{S_{min}}$	K_f[°]
0.005	1256.6	78.5398	1.4737	0.6785	39.6651
0.01	628.3185	39.2699	2.1719	0.4604	26.6191
0.02	314.1593	19.6350	4.7172	0.2120	12.1690
0.04	157.0796	9.8175	22.2521	0.0449	2.5751

$$K_f = \arctan\left(\frac{y}{x}\right). \tag{15.31}$$

The numerical values that have been tested and the results obtained using (15.27)–(15.31) are presented in Table 15.4. The numerical values in Table 15.1 have also been taken into account to compute $\sqrt{b}$. Phase margins obtained with 5[ms] and 10[ms] sampling periods are considered to be acceptable. Moreover, recall that these phase margins may be rendered larger as the sensitivity can be designed to be smaller, at some specific frequencies, than S_{min} in Table 15.4. Of course, the price for this is that the sensitivity would be larger at some other frequencies, as explained in Sect. 8.4. Finally, because of hardware limitations, a sampling period smaller than 10[ms] has not been accomplished. Thus:

$$T_s = 10[\text{ms}]$$

is the sampling period used to control the Furuta pendulum that has been built.

15.10 The Builder 6 C++ Program

In this section, the Builder 6 C++ program executed by the portable computer is presented. In the following, some of the most important parts of this computer program are explained. The corresponding flow diagram is presented in Fig. 15.29.

First, some important constants, such as the mechanism parameters and the controller gains, are defined. The control cycle is contained in the instruction "void ProcessByte(BYTE byte)". At the beginning of the control cycle, the arm position is first read, then the pendulum position is obtained, and, finally, the electric current is read. These variables are converted into radians and amperes respectively. Note that a 2000 constant value is added to the pendulum encoder count. This is done to assign $\theta_1(0) = \pi$ when the pendulum is at its bottom stable position, i.e., the pendulum's initial position. The inverted pendulum position "theta1d" is assigned to be either 0[rad] or 2π[rad], depending on which of these values is more suitable. The condition to switch on the stabilizing controller (15.21) is set as $3(\theta_1 - \text{theta1d})^2 + \dot{\theta}_1^2 < 0.05$. If this is not the case then the controller (15.10) is kept working. Once this decision is taken, the torque to be applied is computed by the corresponding controller. In the case of the swinging-up controller, torque that must

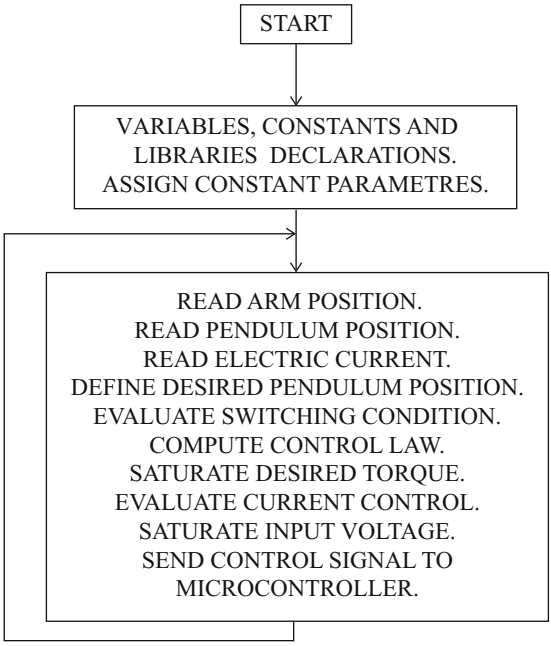

Fig. 15.29 Flow diagram for PC programming when controlling a Furuta pendulum

be generated by the DC motor actuator is obtained multiplying by 5.8 the torque computed with the controller in (15.10). This is done to heuristically compensate for the fact that the power electronics stage cannot supply enough electric current, as explained in Sect. 15.7.

The desired electric current is then obtained in terms of torque to be generated. After that, the voltage to be applied to the motor is computed as the output of a PI electric current controller driven by the electric current error. Then, this voltage is restricted to the range $[-10, +10]$[V]. The voltage information is sent to the microcontroller PIC16F877A as an 8-bit word where the highest bit represents the sign and the lowest 7 bits contain the magnitude information. Some data are saved in a file to be plotted when the experiment has finished. Then, the program returns to the beginning of the instruction "void ProcessByte(BYTE byte)" where it remains waiting until new data on the arm position are received. This means that the sampling period is actually established by the microcontroller, as explained in the next section.

```
//Program for control of a Furuta pendulum
//---------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "Main.h"
#include <math.h>
#include <stdio.h>
#include <share.h>
#include <conio.h>
```

```
#include <dos.h>
//---------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"
//---------------------------------------------------
#define Ts 0.01
#define pprb 400.0
#define pprp 1000.0
#define pi_ 3.141593
#define kpi 0.8 // 0.03
#define kii 130.0
#define bi -2.5
#define VM 10.0
#define IM 2.5
//-------------------Swing up controller----------------
#define ke 480
#define kw 1.0
#define kth 17.0
#define kd 3.0
//----------------Some constants-------------------------
#define g 9.81
#define km 0.0368
// ------------- Furuta pendulum parameters ---------------
#define m1 0.02218
#define l1 0.129
#define L0 0.1550
#define J1 0.0001845
#define I0 0.00023849
#define K 5.8
#define k1 0.0415
#define k2 0.0415
#define k3 0.5189
#define k4 0.0728
// -----------------------------------------
FILE *ptrMonit;
TMainForm *MainForm;
unsigned char flagcom=0,flagfile=0,signo_sal,pwm; // 8 bits
unsigned short int pos1,pos2,corr; // 16 bits
float uf,pwmf,t=0,escp=pi_/(2.0*pprp),escb=pi_/(2.0*pprb);
float cambio=-1.0,signo,theta1=pi_,tau,E,F,det_d;
float norma2,theta0p=0,theta1p=0,theta0_1=0,theta1_1=0,
theta0=0;
float st1,ct1,s2t1,l1c=l1*l1,m1c=m1*m1,L0c=L0*L0,theta0pc;
float CF1=-(J1+m1*l1c)*m1*l1c,CF2=-0.5*m1c*l1c*l1*L0;
float CF4=-m1c*l1c*L0*g,theta1pc,V,theta1d,CF3=(J1+m1*l1c)
*m1*l1*L0;
float id,ic,ei,propi,intei,mi=2.0*IM/1023.0,iTs=1/Ts,
escs=127/VM;
float au1,au2,au3;
//---------------------------------------------------------------
void ProcessByte(BYTE byte)
{
if(flagcom!=0)
```

```
flagcom++;
if((byte==0xAA)&&(flagcom==0))
{
pos1=0;
pos2=0;
corr=0;
flagcom=1;
}
if(flagcom==2)
{
pos1=byte;
pos1=pos1<<8;
}
if(flagcom==3)
{
pos1=pos1+byte;
}
if(flagcom==4)
{
pos2=byte;
pos2=pos2<<8;
}
if(flagcom==5)
{
pos2=pos2+byte+2000;
}
if(flagcom==6)
{
corr=byte;
corr=corr<<8;
}
if(flagcom==7)
{
theta0_1=theta0;
theta1_1=theta1;
corr=corr+byte;
ic=mi*corr+bi;
theta0=(signed short int)pos1;
theta0=escb*theta0; //arm position
theta1=(signed short int)pos2;
teta1=escp*teta1; //pendulum position
theta0p=(theta0-theta0_1)*iTs; //arm velocity
theta1p=(theta1-theta1_1)*iTs; //pendulum velocity
if(theta1<pi_)
{
theta1d=0.0;
}
else
{
theta1d=6.2832;
}
norma2=3.0*(theta1-theta1d)*(theta1-theta1d)+theta1p*theta1p;
if((norma2<0.05)||(cambio==1.0))
{
```

```
// ------------- Stabilizing controller ------------------
tau=k1*theta0 + k2*theta0p + k3*(theta1-theta1d) + k4*theta1p;
cambio=1.0;
}
else
{
//---------------Swinging up controller--------------------
st1=sin(theta1);
ct1=cos(theta1);
s2t1=sin(2*theta1);
theta0pc=theta0p*theta0p;
theta1pc=theta1p*theta1p;
au1=CF3*st1*theta1pc +CF4*ct1*st1;
F=CF1*s2t1*theta0p*theta1p +CF2*ct1*s2t1*theta0pc +au1;
au2=m1c*l1c*L0c*st1*st1;
det_d=(I0+m1*l1c*st1*st1)*(J1+m1*l1c)+J1*m1*L0c+au2;
au3=m1*l1c*theta1pc+m1*l1c*st1*st1*theta0pc+m1*L0*l1*ct1*
theta0p*theta1p;
E=0.5*(I0*theta0pc+J1*theta1pc+m1*L0c*theta0pc+au3)+m1
*g*l1*(ct1-1.0);
tau=(-(kw*F)-(det_d*(kd*theta0p+kth*theta0)))/(det_d*ke
*E+kw*(J1+m1*l1c));
tau=K*tau;
V=(ke*E*E)/2.0+kw*theta0pc/2.0+(kth*theta0*theta0)/2.0;
}
id=tau/km;
//---PI Electric current loop-----------------------
if(id>IM)
id=IM;
if(id<-IM)
id=-IM;
ei=id-ic;
propi=kpi*ei;
if((intei<VM)&&(intei>(-VM)))
intei=intei+kii*Ts*ei;
else
{
if(intei>=VM)
intei=0.95*VM;
if(intei<=(-VM))
intei=-0.95*VM;
}
uf=propi+intei;
if(uf>VM)
uf=VM;
if(uf<-VM)
uf=-VM;
pwmf=uf;
//----------------Screen --------------------------
MainForm->Edit1->Text = FloatToStr (theta0p);
MainForm->Edit2->Text = FloatToStr (theta1p);
MainForm->Edit3->Text = FloatToStr (theta0);
MainForm->Edit4->Text = FloatToStr (theta1);
MainForm->Edit5->Text = FloatToStr (id);
```

```
MainForm->Edit6->Text = FloatToStr (E);
MainForm->Edit7->Text = FloatToStr (t);
MainForm->Edit8->Text = FloatToStr (ic);
//--Input voltage: 7 bits, magnitude, plus 1 bit for sign
pwmf=escs*pwmf;
pwm=(unsigned char)abs(pwmf);
if(pwmf<0)
pwm=pwm+128;
// Sending byte
MainForm->Acknowledge();
MainForm->send_byte(pwm);
/*Open/create a file */
if(flagfile==0)
if((ptrMonit=fopen("MONIT.TXT", "w"))==NULL){}
flagfile=1;
/* Writing to text file */
fprintf(ptrMonit,"%3.3f\t%3.3f\t%3.3f\n",t,theta0,theta1);
t=t+Ts;
flagcom=0;
}//closing last if in falgcom
}//closing function ProcessByte
//---------------------------------------------------------
__fastcall TMainForm::TMainForm(TComponent* Owner)
: TForm(Owner), SerialPort(1, ProcessByte), fAcknowledge(true)
{
if (SerialPort.IsReady() != TRUE)
MessageBox(NULL, "Port problems ", "Error", MB_OK);
}
//---------------------------------------------------------
void TMainForm::send_byte(unsigned char byte_sal)
{
if (fAcknowledge == false)
return;
SerialPort.WriteByte(byte_sal);
fAcknowledge = false;
}
//---------------------------------------------------------
void TMainForm::Acknowledge()
{
fAcknowledge = true;
}
void __fastcall TMainForm::Button1Click(TObject *Sender)
{
/* close the file */
fclose(ptrMonit);
Close();
}
//---------------------------------------------------------
```

15.11 The PIC16F877A Microcontroller C Program

The C program executed by the microcontroller is presented in this section. In the following, some of the most important parts of this program are explained. The interested reader is referred to [8] for a precise explanation of the instructions in this program. The corresponding flow diagram is presented in Fig. 15.30.

First, some important definitions are made concerning both hardware and software. After that, an interruption is defined that is intended to increment or decrement counts of encoders used to measure both the arm and the pendulum positions. The sampling period is set to 10[ms] by assigning "tope=196", i.e., $T_s = 4 \times 256 \times 196/(20 \times 10^6) = 0.01$[s] as indicated at the end of the program. The control cycle is contained within the instruction "while(TRUE)". There, the Furuta pendulum variables are measured: the electric current first, then the arm position, and, finally, the pendulum position. As the next step, the arm position, the pendulum position, and the electric current are sent (in this order) to the portable computer. Once this is performed, the system remains waiting until the voltage data are received from the portable computer. Then, the voltage information is sent to the L298 full-bridge driver. This is done by taking into account the voltage sign contained in the highest bit and the voltage magnitude contained in the lowest 7 bits. After that, the program remains waiting for the sampling period to end. This is established by the instruction "while(TMR0<tope);". Once this is accomplished, the program returns to the beginning of the "while(TRUE)" cycle.

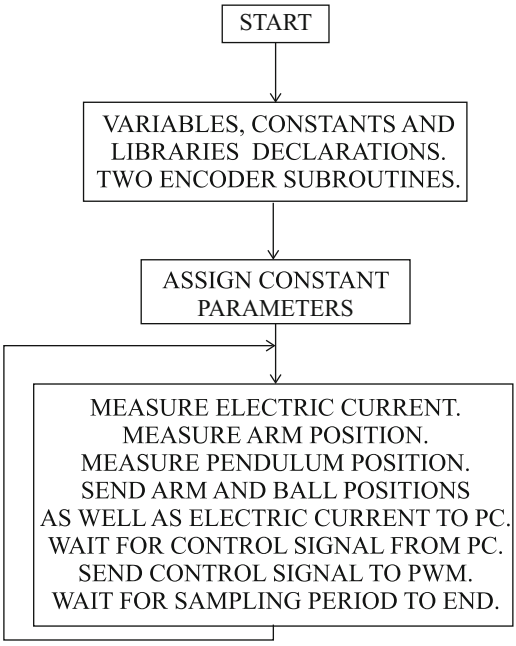

Fig. 15.30 Flow diagram for PIC16F877A microcontroller programming when used as an interface between a Furuta pendulum and a personal computer

```
// Program for data exchange with a PC
// PIC16F877A and PCWH V3.43 compiler
#include<16f877A.h>
#device adc=10 //adc 10 bits
#include<stdlib.h>
#include<math.h>
#fuses HS,NOWDT,PUT,NOBROWNOUT,NOLVP,NOWRT,NOPROTECT,NOCPD
#use delay(clock=20000000) // Xtal frequency
//Serial port configuration
#use rs232(baud=115200,XMIT=PIN_C6,RCV=PIN_C7,BITS=8,PARITY=N)
//Port and register addresses
#byte OPTION= 0x81
#byte TMR0 = 0x01
#byte CVRCON= 0x9D
#byte ADCON1= 0x9F
#byte PORTA = 0x05
#byte PORTB = 0x06
#byte PORTC = 0x07
#byte PORTD = 0x08
#byte PORTE = 0x09

#bit PC0 = 0x07.0
#bit PD0 = 0x08.0
#bit PD1 = 0x08.1
#bit PD2 = 0x08.2
#bit PD3 = 0x08.3
#bit PD4 = 0x08.4
#bit PD5 = 0x08.5
//------------ Variables------------//
int16 inter,cuenta1,cuenta2,conv;
int8 cuentaH1,cuentaL1,cuentaH2,cuentaL2,puerto,AB,AB_1,BC,
BC_1,aux;
int8 convH,convL,tope,dato,pwm;
//------------Encoder interruption ------------//
#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))
{
cuenta1--;
}
else
{
cuenta1++;
}
AB_1=AB;

BC=((puerto)&(0xC0))>>6;
```

```
aux=BC^BC_1;
if(aux!=0)
if(aux!=3)
if(((BC_1<<1)^BC)&(0x02))
{
cuenta2--;
}
else
{
cuenta2++;
}
BC_1=BC;
}

//------------ Main program------------//
void main(void)
{
set_tris_a(0b11111111);
set_tris_b(0b11111111);
set_tris_c(0b10000000);// Serial communication and pwm pins
set_tris_d(0b00001100);
set_tris_e(0b11111111);
PC0=0;
OPTION=0x07; //timer0 1:256
setup_ccp1(CCP_PWM); //pwm configuration
//T=(1/clock)*4*t2div*(period+1)
setup_timer_2(T2_DIV_BY_16,255,1);
PORTC=0;
TMR0=0;
cuenta1=0; //arm initial condition
cuenta2=0; //pendulum initial condition
set_pwm1_duty(0);
AB=0;
AB_1=0;
BC=0;
BC_1=0;
if(PD2)
tope=196;
else
tope=20;
setup_adc(ADC_CLOCK_INTERNAL);//ADC internal clock
setup_adc_ports(ALL_ANALOG);
set_adc_channel(0);
delay_us(15);
enable_interrupts(global);
enable_interrupts(int_rb);
PC0=1;

while(TRUE)
{
conv=read_adc();
inter=(conv)&(0xFF00);
convH=inter>>8;
```

```
convL=(conv)&(0x00FF);
PD4=1; //data sending starts

inter=(cuenta1)&(0xFF00);
cuentaH1=inter>>8;
cuentaL1=(cuenta1)&(0x00FF);

inter=(cuenta2)&(0xFF00);
cuentaH2=inter>>8;
cuentaL2=(cuenta2)&(0x00FF);

putc(0xAA); //sending to serial port
putc(cuentaH1);
putc(cuentaL1);
putc(cuentaH2);
putc(cuentaL2);
if(PD3)
{
putc(convH);
putc(convL);
}

PD4=0; //data sending finishes
PD5=1; //timer is waiting
do
{
if(kbhit())
{
dato=getc();//getting plant input data
if(dato>=128)
{
PD0=0;
PD1=1;
dato=dato-128;
pwm=dato<<1;
}
else
{
PD0=1;
PD1=0;
pwm=dato<<1;
}
set_pwm1_duty(pwm); //sending to pwm
}
}
while(TMR0<tope); //each timer count = (4/FXtal)*256 sec
PD5=0; //timer finishes wait
TMR0=0;
} //closing infinite while
} //closing main
```

15.12 Summary

In this chapter, a Furuta pendulum has been built, in addition to all the interfaces required for closed-loop control. The control scheme that has been used is known as linear state feedback control. It consists in measuring the positions and the velocities of both bodies comprising the mechanism to multiply them by constants and then adding these terms to compute the torque that must be generated by the DC motor used as an actuator. As the mechanism is unstable and nonlinear, the solution of this control problem and the experimental evaluation of the designed control system are of great interest. In this problem, the Euler–Lagrange equations for physical system modeling are very useful. This approach is particularly valuable when modeling mechanical systems composed of several bodies that describe independent movements (degrees-of-freedom).

In this chapter, it has been explained how to obtain the numerical values of the mechanism parameters. This is important to stress, as each of the two bodies comprising the mechanism is composed of smaller mechanical pieces, each contributing to the total body mass, center of mass, and inertia. It has been shown that the classical mechanics formulas related to these variables are useful for computing the mechanism parameters.

A linear approximate model has been obtained from the complete nonlinear model of the mechanism. This step is instrumental in designing a controller for this mechanism, which, however, is valid only if the initial configuration of the mechanism is close to the desired configuration.

15.13 Review Questions

1. Why is the Furuta pendulum open-loop unstable?
2. Give a descriptive explanation on why it is important to control the arm position aside from the pendulum position.
3. Describe the procedure followed to identify the parameters of the complete mechanism.
4. Explain what is represented by each of the operation points of the Furuta pendulum.
5. Why does the potential energy of the mechanism only take into account the pendulum's potential energy?
6. What does a linear approximation of a nonlinear model represent?
7. What is the condition that a linear state feedback controller must satisfy to render the closed-loop system stable?
8. Why is it important for the Furuta pendulum to be controllable? What can be done if this is not the case?

References

1. H. Goldstein, C. Poole, and J. Safko, *Classical Mechanics*, Chapter 1, 3rd Edition, Addison Wesley, New York, 2000.
2. I. Fantoni and R. Lozano, *Non-linear control for underactuated mechanical systems*, Springer, London, 2002.
3. R. C. Hibbeler, *Engineering Mechanics Dynamics*, 10th Edition, Prentice Hall, Upper Saddle River, NJ, 2004.
4. M. Alonso and E. J. Finn, *Physics*, Addison-Wesley, Reading, MA, 1992.
5. G. Stein, Respect the unstable, *IEEE Control Systems Magazine*, August, pp. 12–25, 2003.
6. V. M. Hernández-Guzmán, M. Antonio-Cruz, and R. Silva-Ortigoza, Linear state feedback regulation of a Furuta pendulum: design based on differential flatness and root locus, *IEEE Access*, Vol. 4, pp. 8721–8736, December, 2016
7. PIC16F877A Enhanced Flash Microcontroller, Data sheet, Microchip Technology Inc., 2003.
8. Custom Computer Services Incorporated, CCS C Compiler Reference Manual, 2003.