Properties of Transition Matrix

Using the above relations, we may show the following properties of the transition matrix:

1. $\mathbf{\Phi}^{-1}(t, \tau) = \mathbf{\Phi}(\tau, t)$
2. $\mathbf{\Phi}(t_1, t_2)\mathbf{\Phi}(t_2, t_3) = \mathbf{\Phi}(t_1, t_3)$
3. $\frac{d}{dt}e^{\mathbf{F}t} = \mathbf{F}e^{\mathbf{F}t}$
4. $\mathbf{F}e^{\mathbf{F}t} = e^{\mathbf{F}t}\mathbf{F}$

In general,

$$f_1(\mathbf{F})f_2(\mathbf{F}) = f_2(\mathbf{F})f_1(\mathbf{F})$$

REFERENCES

1. Frame, J. S., "Matrix Functions and Applications," *IEEE Spectrum*, Vol. 1, No. 6, June 1964.
2. Gantmacher, F. R., *The Theory of Matrices*, Vol. 1, Chelsea Publishing Company, New York, 1959.
3. Bellman, R., *Introduction to Matrix Analysis*, McGraw-Hill, New York, 1960.
4. Pipes, L., *Matrix Methods for Engineering*, Prentice-Hall, Englewood Cliffs, N. J., 1963.
5. Timothy, L. K. and B. E. Bona, *State Space Analysis: An Introduction*, McGraw-Hill, New York, 1968.
6. Ogata, K., *State Space Analysis of Control Systems*, Prentice-Hall, Englewood Cliffs, N. J., 1967.

3

The Analysis of Discrete-time Systems: Time-domain Approach

3.1 Introduction

Several examples were given in Chapter 1 of typical systems operating in discrete time. The common characteristic of these systems is that they contain at least one discrete-time component. It is possible, for instance, for the feedback information to be available only at periodic instances when the feedback channel is time shared. Such a system requires a data-hold circuit, since it essentially operates on an open-loop basis between data transfers. An illustration of this type of system is given by Figure 3.1-1.

Another example of a system containing a discrete component is shown in Figure 3.1-2. Illustrated is a closed-loop system that operates in continuous time except for a digital computer inserted into the control loop as an active systems component. Again, a hold circuit is required to maintain control over the system during intervals between data transfers.

A third class of systems operating in discrete time may be distinguished as one in which all components operate in discrete time. Consider, for instance,

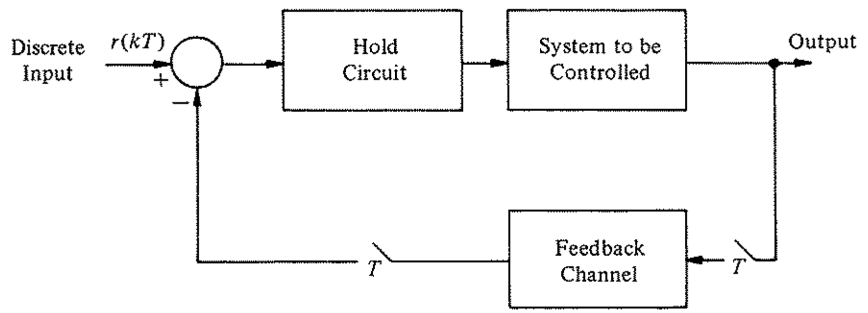


Figure 3.1-1. System with time-shared feedback channel.

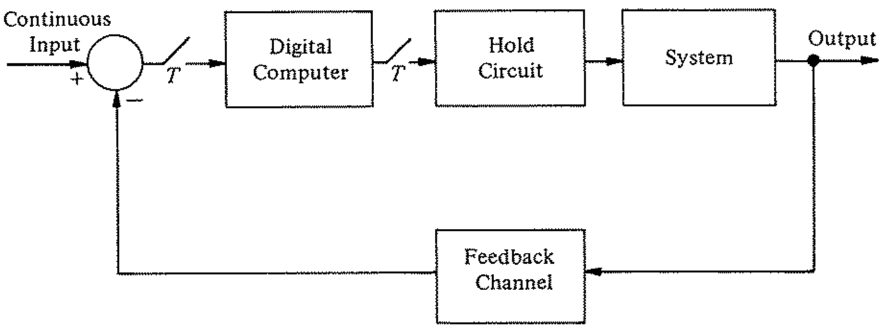


Figure 3.1-2. Computer control system.

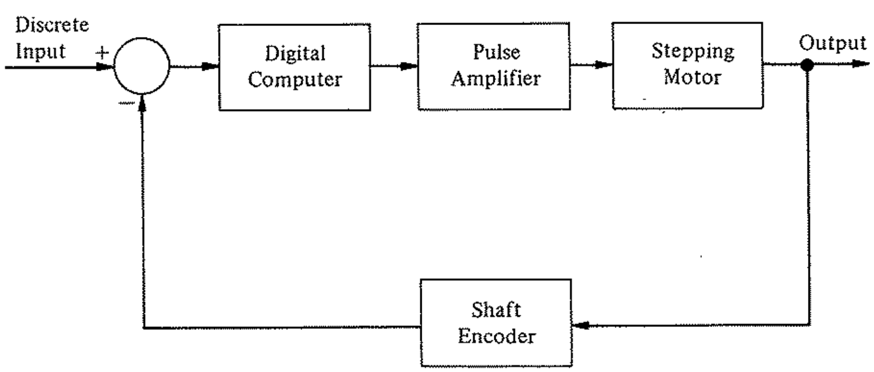


Figure 3.1-3. Digital control system, operating entirely in discrete time.

the system shown in Figure 3.1-3, which consists of a digital computer driving a digital stepping motor with the feedback being provided by a digital shaft encoder. The output of this system is continuous in time, although limited to quantized levels.

Discrete-time systems may be operated in an open-loop or closed-loop manner. The analysis of these systems is generally carried out by using one of two approaches: (1) by discrete state equations or (2) by the use of the z-transform. This chapter serves to demonstrate the application of discrete state techniques.

3.2 Data-hold Techniques

An essential ingredient in the satisfactory operation of a hybrid discrete-time system having components that operate both in discrete time and continuous time is a data-hold device. Its function is to convert a discrete-time function (sequence of numbers) into a continuous-time function in order to provide a suitable input to a continuous-time component.

When the input is a sampled analog signal, it is called a data-hold circuit. However, when the input is a discrete data signal in digital form, such as might originate from a digital computer, the data-hold device provides digital-to-analog conversion in addition to the hold action. Then it is simply called a digital-to-analog converter. As is shown in Chapter 7, a digital-to-analog converter automatically provides hold action. Since mathematically there is no distinction between the two cases, we can treat them alike and provide an identical analysis.

Figure 3.2-1 shows a block diagram representation of a digital-to-analog converter (DAC).

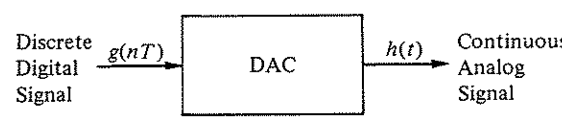


Figure 3.2-1. Block diagram representation of a DAC.

In general, the purpose of a DAC is that of generating a function of continuous time $h(t)$ from a sequence of numbers $g(nT)$ that are separated in time by T-second intervals. It is usually desirable to have the function $h(t)$ correspond roughly to an envelope of the input sequence $g(nT)$. Between sampling times [i.e., $NT \leq t < (N+1)T$], the DAC must extrapolate between the most recent sample and the next to follow.

In effect, the DAC has the properties normally ascribed to an extrapolator. An mth-order extrapolator will be defined as an extrapolator whose present output depends on $m+1$ past sample values. At each sampling time, a new member of the input sequence $g(nT)$ is available so that the extrapolating process must be reinitiated at the sampling instants.

A useful form of extrapolation is that of polynomial extrapolation. Here, it is assumed that the desired signal $h(t)$ may be adequately approximated by an mth-order polynomial; that is,

$$h(nT+\tau) = a_m\tau^m + a_{m-1}\tau^{m-1} + \cdots + a_0 \quad \text{for } 0 \le \tau < T \tag{3.2-1}$$

Since it is desired that $h(t)$ be the envelope of the sequence $g(nT)$, it is natural to require that the output signal $h(t)$ have the value of the input sequence at the sampling times $t = kT$; that is,

$$h(t)|_{t=kT} = h(kT) = g(kT) \quad \text{for all values of } k$$

The coefficients $a_m, a_{m-1}, \ldots, a_0$ for any time interval $nT \le t < (n+1)T$ may be evaluated by forcing $h(t)$ to satisfy the constraints

$$h(kT) = g(kT) \quad \text{for } k = n-m, n-m+1, \ldots, n \tag{3.2-2}$$

That is, $h(nT+\tau)$ is a polynomial that passes through the immediate $m+1$ past values of the input $g(kT)$. At each sampling time the coefficients a_m, $a_{m-1}, \ldots, a_0$ must be reevaluated, since a new data point is available.

Zero-order Hold $(m = 0)$

The simplest type of polynomial extrapolator arises when $h(t)$ is assumed to be a zero-order polynomial $(m = 0)$. In this case, we have, by (3.2-1),

$$h(nT+\tau) = g(nT) \quad \text{for } 0 \le \tau < T, \quad n = 0, \pm 1, \pm 2, \ldots \tag{3.2-3}$$

Figure 3.2-2(a) illustrates a typical response of a zero-order hold.

First-order Hold $(m = 1)$

If $h(t)$ is assumed to be a first-order polynomial, then

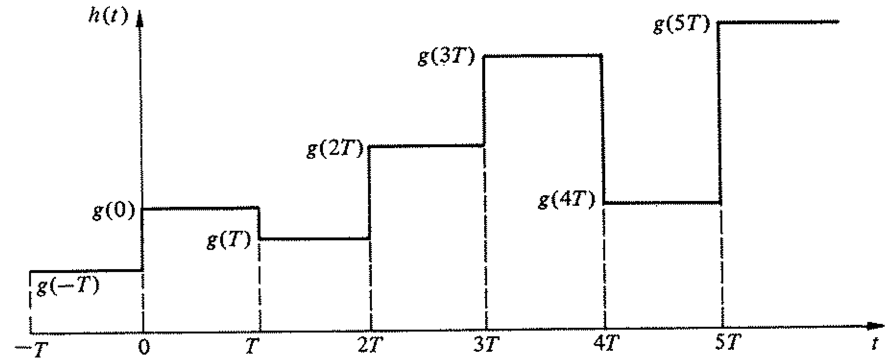
$$h(nT+\tau) = a_1\tau + a_0 \quad \text{for } 0 \le \tau < T \tag{3.2-4}$$

with the requirements corresponding to (3.2-2) being

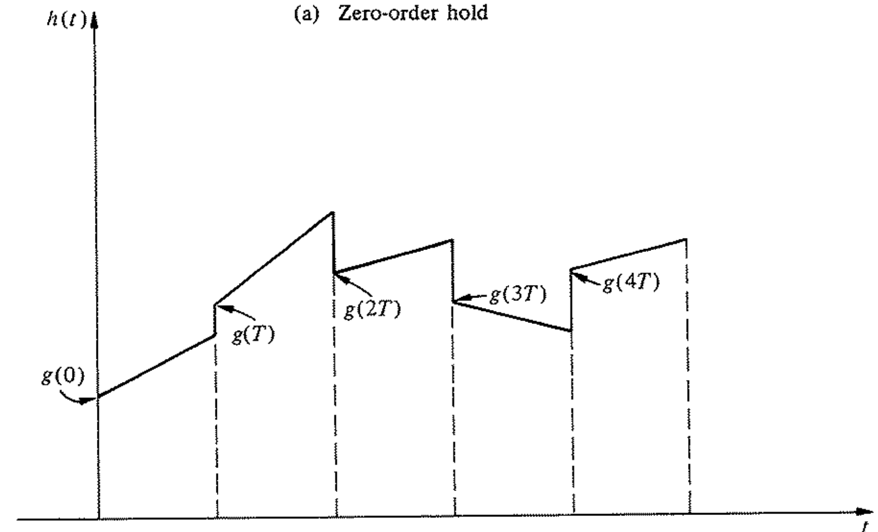
$$h(nT) = g(nT)$$
$$h([n-1]T) = g([n-1]T)$$

which, when τ is set equal to 0 and $-T$ in (3.2-4), gives us

$$a_0 = g(nT), \qquad a_1 = \frac{g(nT) - g([n-1]T)}{T}$$



(a) Zero-order hold



(b) First-order hold

Figure 3.2-2. Typical responses of zero- and first-order holds.

Therefore, the first-order extrapolator is characterized by

$$h(nT+\tau) = \frac{g(nT) - g([n-1]T)}{T}\tau + g(nT) \quad \text{for } 0 \le \tau < T, \quad n = 0, \pm 1, \pm 2, \ldots \tag{3.2-5}$$

A typical response of the first-order hold is shown in Figure 3.2-2(b).

The higher-order holds $(m \ge 2)$ may be generated in a like manner. In general, most modern systems do not use higher-order holds because of the inherent delay which they introduce into the system's response. Also, they are highly noise-sensitive and much more difficult to implement.

3.3 Open-loop Sampled-data Systems

An example of a basic discrete-time system consists of a sampling element, a hold circuit, and a continuous-time system, as shown schematically in Figure 3.3-1. The input $r(t)$ is sampled periodically at intervals T seconds apart to

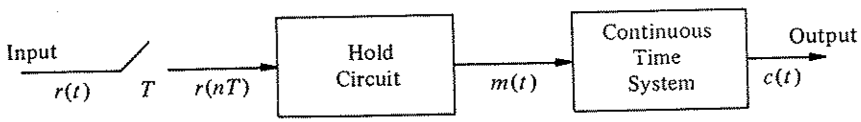


Figure 3.3-1. Open-loop sampled-data system.

generate the sequence of numbers $r(nT)$. The hold circuit changes this discrete-time function to a piecewise continuous function. If it is a zero-order hold, then a function of the form shown by Figure 3.2-2(a) is produced. For higher-order holds, more complicated forms will be generated. The output of the hold circuit represents the input to the continuous-time system. The analysis of such systems is directed at determining the response of the continuous-time system. We shall, therefore, derive a mathematical model suitable to carry out this objective.

The zero-order hold is the most commonly employed hold device. In view of the graph of Figure 3.2-2(a), the output of a zero-order hold is a periodically piecewise constant function. That is,

$$m(t_k + \tau) = r(t_k), \qquad 0 \leq \tau < t_{k+1} - t_k$$

where $r(t_k)$ is the value of the sampled function $r(t)$, at time $t = t_k$. If the sampling occurs at constant intervals then $t_k = kT$ and $t_{k+1} - t_k = T$. In this case, the hold-circuit output can be written as

$$m(kT + \tau) = r(kT), \qquad 0 \leq \tau < T \tag{3.3-1}$$

In Chapter 2 we developed the state equations for a linear, time-invariant, continuous system governed by the continuous state equations

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{G}m(t)$$
$$c(t) = \mathbf{C}\mathbf{x}(t) + dm(t)$$

when its input $m(t)$ is of the form (3.3-1). Namely,

$$\mathbf{x}[(k+1)T] = \mathbf{A}(T)\mathbf{x}(kT) + \mathbf{B}r(kT) \tag{3.3-2}$$
$$\mathbf{c}(kT) = \mathbf{C}\mathbf{x}(kT) + dr(kT) \tag{3.3-3}$$

where $\mathbf{x}(kT)$ is the continuous system's state at time kT, $r(kT)$ is the value of the input signal $r(t)$ at time kT, and

$$\mathbf{A}(T) = e^{\mathbf{F}T}, \qquad \mathbf{B}(T) = \int_0^T e^{\mathbf{F}\tau}\mathbf{G}\, d\tau$$

The matrices $\mathbf{F}$ and $\mathbf{G}$ are the system matrix and the input matrix of the continuous-time system, respectively. Equations (3.3-2) and (3.3-3) may be used to calculate the open-loop system response to the input $r(t)$ at the sampling times kT. It must be understood that the sampling operation permits the passage of the function $r(t)$ only at the sampling instants nT. We consider an example to demonstrate the use of (3.3-2) and (3.3-3).

EXAMPLE

Compute the response of the system shown in Figure 3.3-2 to a square wave input and a sinusoidal wave input for various sampling rates.

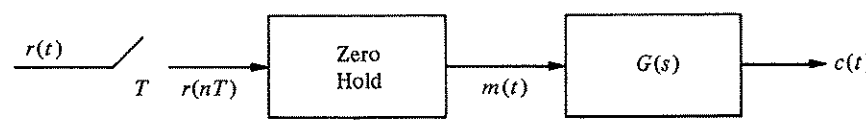


Figure 3.3-2. Open-loop sampled system with zero-order hold.

Let the transfer function of the continuous time system be

$$G(s) = \frac{s+1}{(s+2)(s+10)} \tag{3.3-4}$$

Using nested programming to develop a state model, we obtain

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -12 & 1 \\ -20 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} m(t) \tag{3.3-5}$$
$$c(t) = x_1$$

By any one of the methods of Appendix 2 we determine that

$$e^{\mathbf{F}T} = \mathbf{A}(T) = \begin{bmatrix} -\frac{1}{4}e^{-2T} + \frac{5}{4}e^{-10T} & \frac{1}{8}e^{-2T} - \frac{1}{8}e^{-10T} \\ -\frac{5}{2}e^{-2T} + \frac{5}{2}e^{-10T} & \frac{5}{4}e^{-2T} - \frac{1}{4}e^{-10T} \end{bmatrix} \tag{3.3-6}$$

and

$$\begin{aligned} \mathbf{B}(T) &= \int_0^T \mathbf{A}(\tau)\mathbf{G}\,d\tau \\ &= \int_0^T \begin{bmatrix} -\frac{1}{8}e^{-2\tau} + \frac{9}{8}e^{-10\tau} \\ -\frac{5}{4}e^{-2\tau} + \frac{9}{4}e^{-10\tau} \end{bmatrix} d\tau \\ &= \begin{bmatrix} \frac{1}{20} + \frac{1}{16}e^{-2T} - \frac{9}{80}e^{-10T} \\ -\frac{2}{5} + \frac{5}{8}e^{-2T} - \frac{9}{40}e^{-10T} \end{bmatrix} \end{aligned} \tag{3.3-7}$$

For the specific case, where $T = 1$ for instance, we obtain

$$\mathbf{A}(1) = \begin{bmatrix} -.0338 & .0169 \\ -.338 & .169 \end{bmatrix}$$

and

$$\mathbf{B}(1) = \begin{bmatrix} .0584 \\ -.315 \end{bmatrix}$$

Thus, for this example, the discrete state equations for a piecewise constant input are

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} = \begin{bmatrix} -.0338 & .0169 \\ -.338 & .169 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} + \begin{bmatrix} .0584 \\ -.315 \end{bmatrix} r(kT) \tag{3.3-8}$$

$$c(kT) = x_2(kT)$$

Equations (3.3-8) serve as the mathematical model for this problem.

It is interesting to study the effect of the sample and hold operations on the input signal. Figure 3.3-3 and 3.3-4 show waveforms for $m(t)$ and $c(t)$ for the square wave and sinusoidal wave inputs at various sampling frequencies. A study of these waveforms reveals two important aspects of sampling a signal and then passing it through a zero-order hold circuit. First, we note that the higher the sampling frequency, the better the zero-order hold is capable of generating a time function that represents a good reproduction of the input.

The second observation we wish to make is that the output of the zero-order hold is an approximate reproduction of the input but appears with a time lag relative to the unsampled signal. An exact reproduction is possible in the case of the square wave. However, the sinusoidal wave may suffer greatly when passed through the zero-order hold.

Both figures show the output $c(t)$ for each of the sampling periods used. For the square wave input, the output $c(t)$ is identical to the output for the equivalent unsampled system ($T = 0$) except with time lag equal to the sampling period as long as T is less than one-half the period of the square wave input. For the sine wave input, the output becomes seriously affected as the sampling period increases. For $T = .1$ the output is very nearly equal to that for the unsampled system ($T = 0$). However, when the sampling period is increased to $T = .5$ and $T = 1.0$, it is very apparent that the output cannot be regarded as the sinusoidal response of a linear system.

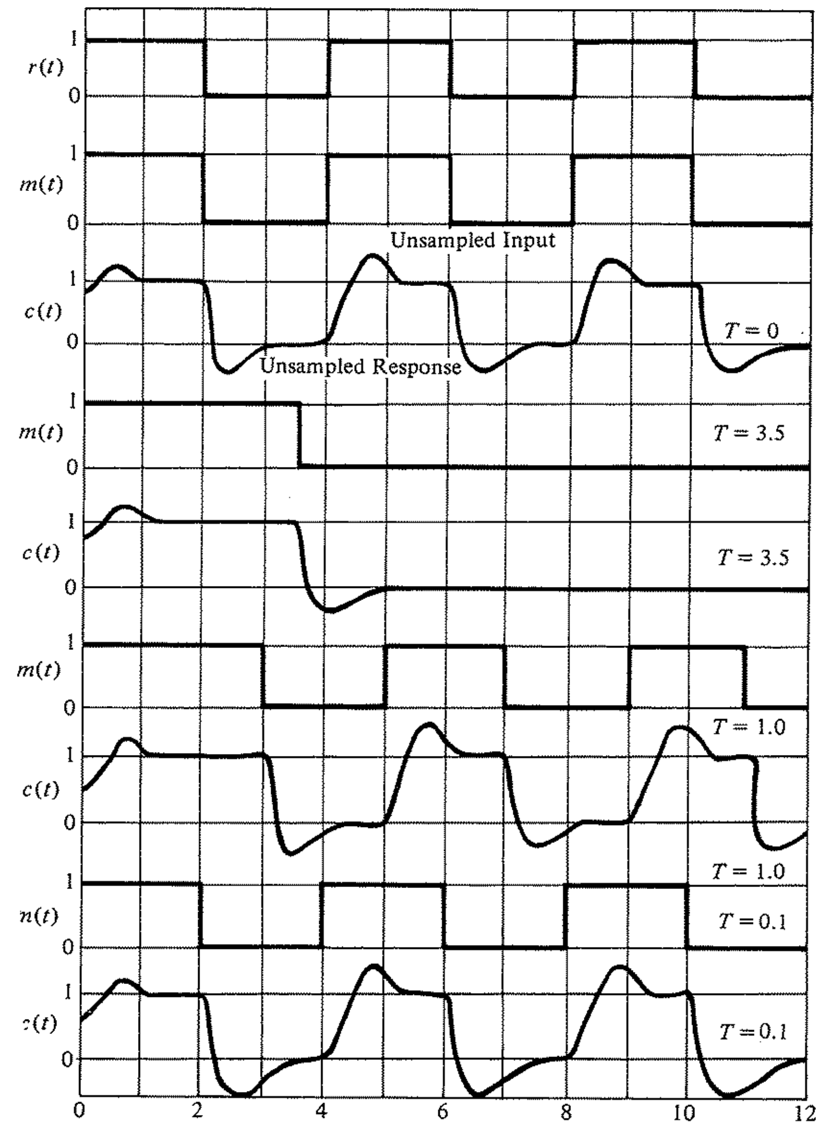


Figure 3.3-3. Response of a sampled-data system to a squarewave input for various sampling rates.

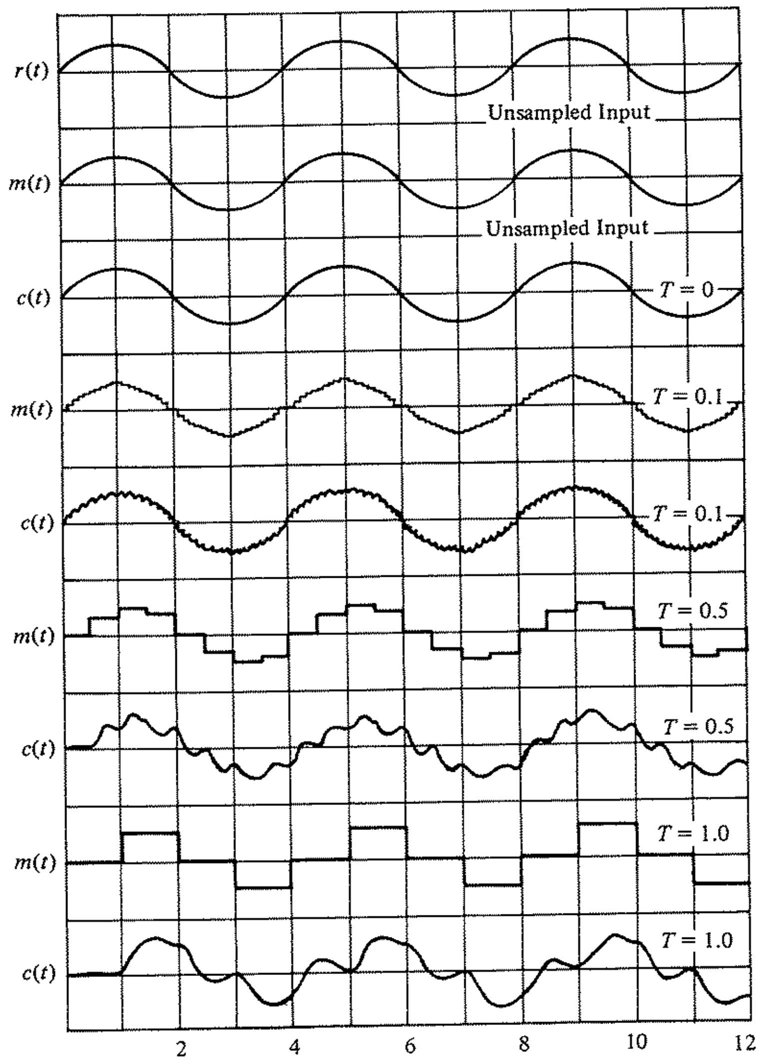


Figure 3.3-4. Response of a sampled system to a sinusoidal input for various sampling rates.

The sampled-data system described above was simulated on an analog computer at a speed 10 times slower than real time. A diagram of the analog computer circuit used is shown in Figure 3.3-5. The sample and hold circuit is implemented by use of an integrator, which is switched between the OPERATE and RESET modes in synchronism with the sampling rate. The

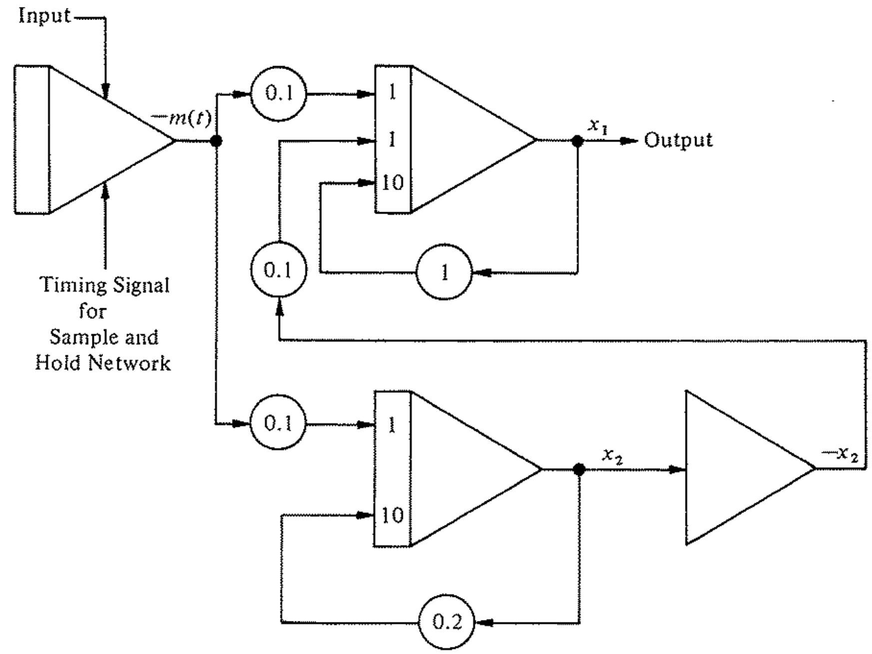


Figure 3.3-5. Analog simulation of sampled-data system.

input is supplied through the initial condition terminal. When the integrator is in RESET mode, it tracks the input function; when it is in OPERATE mode, it stores the input function. If the RESET time is kept very short and the integrator's time constant is very small, the integrator effectively samples the input during RESET and holds it during OPERATE. An integrator operating in this fashion is called a *track-store unit*, which is frequently utilized in hybrid computations (see Chapter 9, Section 9.5).

The timing signal may be generated by means of the circuit shown in Figure 3.3-6(a). A triangular wave generator output is fed into a comparator network, which is biased with a fixed voltage slightly less than the peak output of the triangular wave generator. The comparator produces a pulse signal of the same frequency as the wave generator. The wave shapes are shown in Figure 3.3-6(b).

The above example effectively demonstrates several important characteristics of sampled systems with zero-order holds. Most outstanding are the facts that the sampling process introduces data loss and that the holding process introduces time lag.

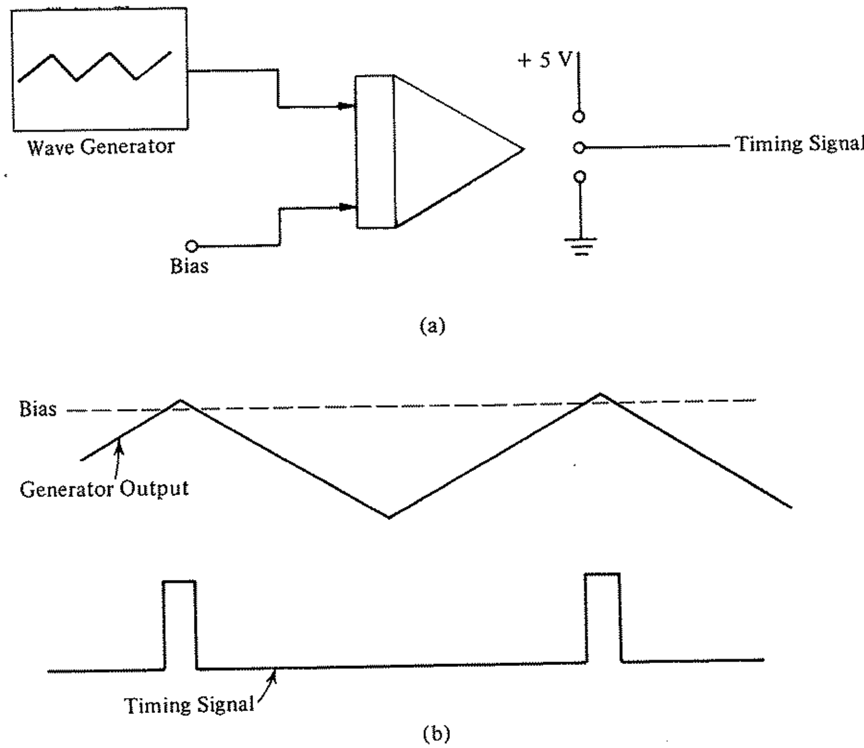


Figure 3.3-6. Generation of timing signal.

3.4 Discrete State Equations of Closed-loop Sampled-data Systems

When certain signals in a conventional closed-loop feedback system are used only at discrete times, such a system may be viewed as a sampled-data system. A symbolic representation of a typical sampled-data system is shown in Figure 3.4-1. Here the sampling operation is applied to the error signal. Frequently, it is required to sample the feedback signal. To illustrate procedures applicable to the analysis of sampled-data closed-loop systems, let us consider the following example.

EXAMPLE 3.4-1

For the system shown in Figure 3.4-2 calculate the response to a step input for sampling periods of .1, 1, and 4 seconds.

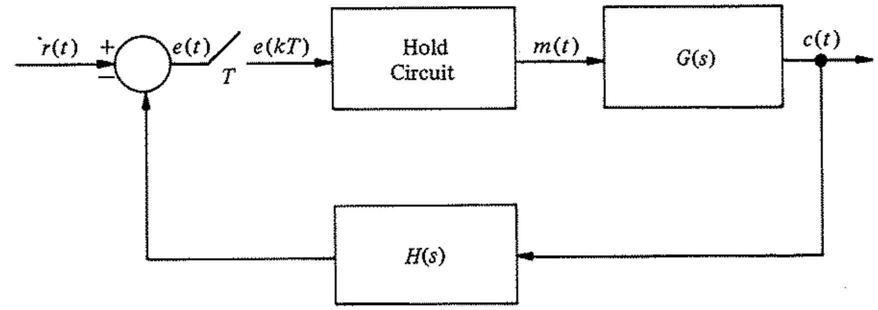


Figure 3.4-1. Closed-loop sampled-data system.

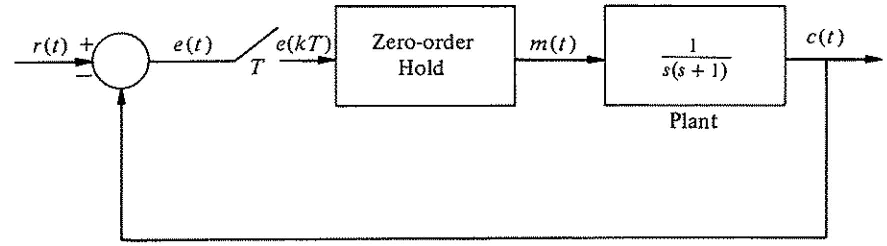


Figure 3.4-2. Block diagram of a sampled system.

Since $e(t) = r(t) - c(t)$, we have

$$e(kT) = r(kT) - c(kT) \tag{3.4-1}$$

The state equations for the plant under control (see Section 1.5) are

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} m(t) \tag{3.4-2}$$

Since $m(t)$ is the output of a zero-order hold, it is a piecewise constant input. Consequently, we may develop discrete state equations relating $c(kT)$ to $e(kT)$.

The transition matrix corresponding to (3.4-2) is obtained from Section 1.5.

$$\mathbf{A}(T) = e^{\mathbf{F}T} = \begin{bmatrix} e^{-T} & 0 \\ 1 - e^{-T} & 1 \end{bmatrix}$$

and the discrete input matrix is

$$\mathbf{B}(T) = \begin{bmatrix} 1 - e^{-T} \\ T - 1 + e^{-T} \end{bmatrix}$$

The discrete state equations are given by

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} = \begin{bmatrix} e^{-T} & 0 \\ 1-e^{-T} & 1 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} + \begin{bmatrix} 1-e^{-T} \\ T-1+e^{-T} \end{bmatrix} e(kT) \tag{3.4-3}$$

$$c(kT) = x_2(kT)$$

Substituting (3.4-1) into (3.4-3) yields

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} = \begin{bmatrix} e^{-T} & 0 \\ 1-e^{-T} & 1 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} + \begin{bmatrix} 1-e^{-T} \\ T-1+e^{-T} \end{bmatrix} [r(kT) - x_2(kT)]$$

$$c(kT) = x_2(kT)$$

which simplifies to

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} = \begin{bmatrix} e^{-T} & e^{-T}-1 \\ 1-e^{-T} & 2-T-e^{-T} \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} + \begin{bmatrix} 1-e^{-T} \\ T-1+e^{-T} \end{bmatrix} r(kT) \tag{3.4-4}$$

$$c(kT) = x_2(kT)$$

These relationships represent the closed-loop discrete state equations. We consider now the response of this system for three sampling periods ($T = .1$, 1, and 4 seconds) subject to the step input; therefore,

$$r(kT) = 1, \qquad k = 0, 1, 2, \ldots$$

and we make the assumption that the system is initially at rest; that is,

$$x_1(0) = x_2(0) = 0$$

For $T = .1$, expression (3.4-4) becomes

$$\begin{bmatrix} x_1[.1(k+1)] \\ x_2[.1(k+1)] \end{bmatrix} = \begin{bmatrix} .905 & -.095 \\ .095 & .995 \end{bmatrix} \begin{bmatrix} x_1(.1k) \\ x_2(.1k) \end{bmatrix} + \begin{bmatrix} .095 \\ .005 \end{bmatrix}$$

$$c(.1k) = x_2(.1k)$$

By repeated application of these equations for $k = 0, 1, \ldots$, we obtain

$$\{c(0),\ c(.1),\ c(.2),\ \cdots\} = \{0,\ .005,\ .019,\ .041,\ .071,\ .106,\ .146,\ \ldots\}$$

For $T = 1$ the equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} .368 & -.632 \\ .632 & .632 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} .632 \\ .368 \end{bmatrix}$$

and the first few numbers in the output sequence are

$$\{c(0),\ c(1),\ c(2),\ \ldots\} = \{.0,\ .368,\ 1.000,\ 1.399,\ 1.399,\ 1.147,\ .894,\ \ldots\}$$

For $T = 4$ we have

$$\begin{bmatrix} x_1[4(k+1)] \\ x_2[4(k+1)] \end{bmatrix} = \begin{bmatrix} .0183 & -.98 \\ .98 & 2.02 \end{bmatrix} \begin{bmatrix} x_1(4k) \\ x_2(4k) \end{bmatrix} + \begin{bmatrix} .98 \\ 3.02 \end{bmatrix}$$

The output is

$$\{c(0),\ c(4),\ c(8),\ \ldots\} = \{.0,\ 3.02,\ -2.11,\ 5.34,\ -4.82,\ 8.6,\ -8.8,\ \ldots\}$$

The responses for the three cases are plotted in Figure 3.4-3. Clearly, for $T = .1$ and $T = 1$, the response is underdamped. For both cases the response is stable and follows the input. On the other hand, when the sampling period is increased to $T = 4$, the response shows that the sampled system has become unstable, since the oscillation amplitude grows with time.

It is interesting to compare these three responses with the equivalent unsampled system. The closed-loop transfer function is

$$C(s) = \frac{1}{s^2 + s + 1} R(s)$$

and for $R(s) = 1/s$ the output is

$$C(s) = \frac{1}{s^2 + 2(\frac{1}{2})s + 1} \frac{1}{s}$$

From the transfer function it is indicated that the damping factor is $\zeta = .5$. The response can be easily computed; when compared with the response of the .1 second sampled system, it is found to be almost identical to it. The difference is so small that the plot over the time scale selected in Figure 3.4-3 would not reveal it.

The above example serves to demonstrate a procedure to be followed in the analysis of a sampled system using discrete state techniques. It furthermore alerts the designer to the fact that the selection of the sampling rate is no

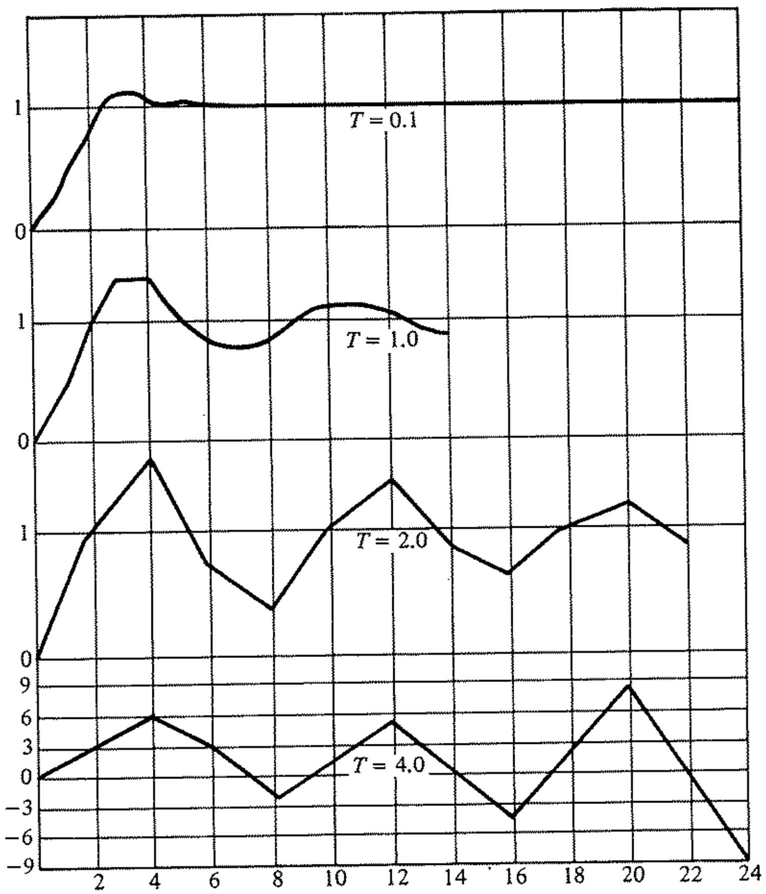


Figure 3.4-3. Response of system for various sampling rates. Input is unit step. (Note scale change on bottom curve.)

arbitrary matter. For reasons of design simplicity a low sampling rate is advisable; this, however, works against the requirement of good data passage through the sample and hold devices. Furthermore, the stability of the system is threatened if the sampling period is selected too large. More will be said about the stability of a sampled system in a subsequent section.

In order to generalize the concepts developed by this example, let us consider the configuration shown in Figure 3.4-4. Since $m(t)$ is a piecewise constant signal, it follows by the discussion of Chapter 2, Section 2.5, that a state variable representation for the plant will be of the form

$$\begin{aligned} \mathbf{x}[(k+1)T] &= \mathbf{A}(T)\mathbf{x}(kT) + \mathbf{B}(T)m(kT) \\ y(kT) &= c(kT) = \mathbf{C}\mathbf{x}(kT) \end{aligned} \tag{3.4-5}$$

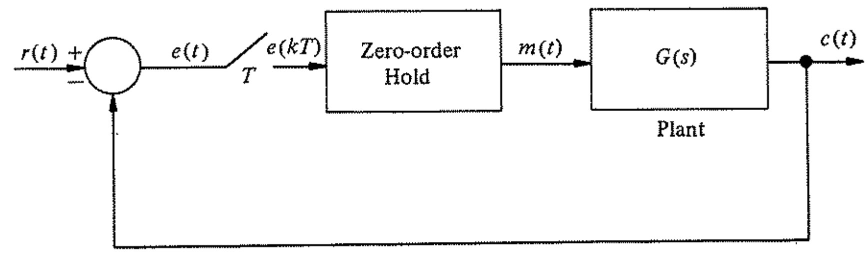


Figure 3.4-4. Generalized sampled-data system.

where the direct transmission matrix is assumed zero. Now

$$m(kT) = e(kT) = r(kT) - c(kT) = r(kT) - \mathbf{C}\mathbf{x}(kT)$$

so that (3.4-5) becomes

$$\begin{aligned} \mathbf{x}[(k+1)T] &= \{\mathbf{A}(T) - \mathbf{B}(T)\mathbf{C}\}\mathbf{x}(kT) + \mathbf{B}(T)r(kT) \\ &= \bar{\mathbf{A}}(T)\mathbf{x}(kT) + \mathbf{B}(T)r(kT) \end{aligned} \tag{3.4-6}$$

$$c(kT) = \mathbf{C}\mathbf{x}(kT) \tag{3.4-7}$$

A comparison of the open-loop dynamics of the plant under control (3.4-5) with its closed-loop form as given by (3.4-6) indicates that an essential difference lies in the makeup of the transition matrix.* The open-loop transition matrix $\mathbf{A}(T)$ has been transformed to $\mathbf{A}(T) - \mathbf{B}(T)\mathbf{C}$ by the feedback process. This feedback property may generate many desirable characteristics, such as a more rapidly responding system, a more stable system, etc. However, it is important that one investigate these characteristics carefully, as feedback can also have a destabilizing effect and can generate other undesirable characteristics.

This generalization approach may be extended to more complex systems (e.g., the system shown in Figure 3.4-1) in a straightforward manner.

3.5 The Discrete-state Analysis of Computer Control Systems

A computer control system is inherently a sampled-data system. Typically, it is of the form shown in Figure 3.5-1, where a digital computer is included in the forward loop of the system. As will be seen in later chapters, the presence of a digital computer opens up a wealth of considerably more sophis-

*The closed-loop terminology arises because the output signal $c(t)$ is fed back and subtracted from the input signal $r(t)$ to generate $e(t)$. With no feedback (open-loop) we would have $e(t) = r(t)$.

ticated design approaches than are possible from a more conventional approach. For this reason, it will be desirable to have computer control systems requiring the feedback of more variables than just the output. Although Figure 3.5-1 represents the basic structure of a computer control system, it is

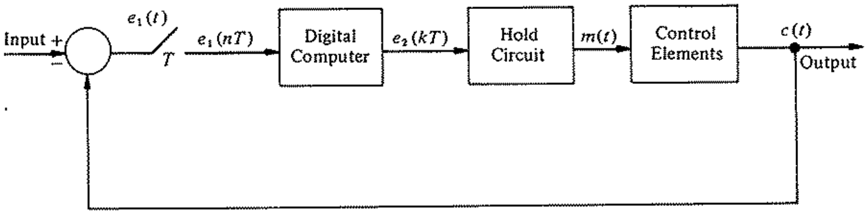


Figure 3.5-1. Schematic of a computer control system.

insufficient to indicate the sophisticated details of a computer control system in which the capabilities of the computer are fully exploited. Let the system shown above suffice as an introduction to the analysis of a system containing a digital computer as an active system element.

In the simplest application of a computer control system, the digital computer is employed to implement the linear recursion equation

$$\begin{aligned} e_2(kT) &= b_0e_1(kT) + b_1e_1[(k-1)T] + \cdots + b_ne_1[(k-n)T] \\ &\quad - a_1e_2[(k-1)T] - a_2e_2[(k-2)T] - \cdots - a_ne_2[(k-n)T] \end{aligned} \tag{3.5-1}$$

where $e_1(kT)$ and $e_2(kT)$ represent the input and output sequence of the digital computer at the discrete time kT.

It was shown in Chapter 2 how a linear recursion formula can be changed into a discrete state equation. If we follow those procedures, the discrete state equations in direct programming form corresponding to (3.5-1) are

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \\ \vdots \\ x_n[(k+1)T] \end{bmatrix} = \begin{bmatrix} -a_1 & -a_2 & \cdots & -a_n \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \\ \vdots \\ x_n(kT) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} e_1(kT) \tag{3.5-2}$$

$$e_2(kT) = [\hat{b}_1 \quad \hat{b}_2 \quad \ldots \quad \hat{b}_n] \begin{bmatrix} x_1(kT) \\ x_2(kT) \\ \vdots \\ x_n(kT) \end{bmatrix} + b_0e_1(kT) \tag{3.5-3}$$

where $\hat{b}_j = b_j - b_0a_j$ for $j = 1, 2, \ldots, n$.

With the recursion equation of the digital computer converted into state variable form, it is now an easy matter to proceed with the analysis of a typical computer control system.

EXAMPLE 3.5-1

Derive the closed-loop state equations for the system shown in Figure 3.5-2.

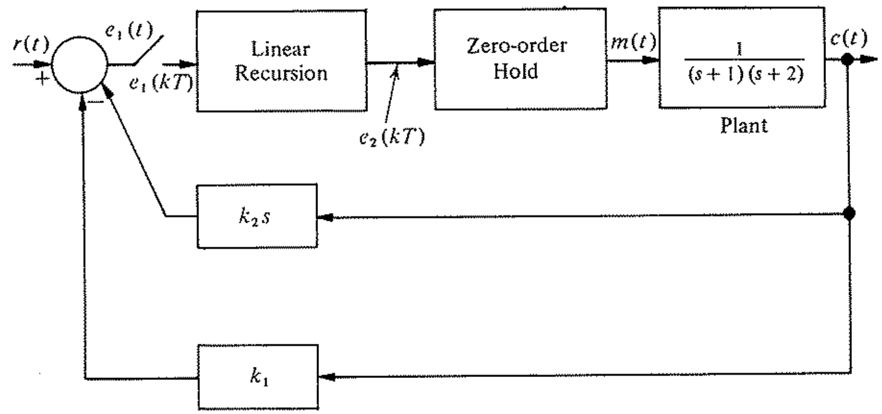


Figure 3.5-2. Computer control system.

Let the linear recursion equation be given as

$$e_2(kT) = 1.2e_1(kT) - .4e_1[(k-1)T] - .25e_2[(k-1)T] \tag{3.5-4}$$

The corresponding state equations are

$$\begin{aligned} x_3[(k+1)T] &= -.25x_3(kT) + e_1(kT) \\ e_2(kT) &= -.1x_3(kT) + 1.2e_1(kT) \end{aligned} \tag{3.5-5}$$

The continuous state equations of the plant obtained by using the direct programming method are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -3 & -2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} m(t) \tag{3.5-6}$$
$$c(t) = x_2$$

Since $m(t)$ is a piecewise constant input, we can easily derive the discrete state equations of (3.5-6) to be

$$\begin{aligned} \begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} &= \begin{bmatrix} -e^{-T} + 2e^{-2T} & -2e^{-T} + 2e^{-2T} \\ e^{-T} - e^{-2T} & 2e^{-T} - e^{-2T} \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} \\ &\quad + \begin{bmatrix} e^{-T} - e^{-2T} \\ \frac{1}{2}e^{-T} + \frac{1}{2}e^{-2T} \end{bmatrix} m(kT) \end{aligned} \tag{3.5-7}$$

Two feedbacks paths are included in the system such that

$$e_1(t) = r(t) - k_1 c(t) - k_2 \dot{c}(t) \tag{3.5-8}$$

Since the state model adopted for the plant is a direct programming version, we have

$$\begin{aligned} c(t) &= x_2 \\ \dot{c}(t) &= x_1 \end{aligned} \tag{3.5-9}$$

At the sampling instants (3.5-8) yields

$$e_1(kT) = r(kT) - k_1 c(kT) - k_2 \dot{c}(kT)$$

or, using (3.5-9), we obtain

$$e_1(kT) = r(kT) - k_1 x_2(kT) - k_2 x_1(kT) \tag{3.5-10}$$

Also, we note that

$$m(kT) = e_2(kT) \tag{3.5-11}$$

Equations (3.5-5), (3.5-7), (3.5-10), and (3.5-11) may now be combined to establish the closed-loop discrete state equations. For convenience we shall write (3.5-7) in the form

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} m(k) \tag{3.5-12}$$

Using this notation, we find that the combined state equations are

$$\begin{bmatrix} x_1[(k+1)T] \\ x_2[(k+1)T] \\ x_3[(k+1)T] \end{bmatrix} = \begin{bmatrix} a_{11} - 1.2b_1k_2 & a_{12} - 1.2b_1k_2 & -0.1b_1 \\ a_{21} - 1.2b_2k_2 & a_{22} - 1.2b_2k_1 & -0.1b_2 \\ -k_2 & -k_1 & -0.25 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \\ x_3(kT) \end{bmatrix} + \begin{bmatrix} 1.2b_1 \\ 1.2b_2 \\ 1 \end{bmatrix} r(kT) \tag{3.5-13}$$

Equations (3.5-13) may be solved iteratively to calculate the response of the computer control system to any input $r(t)$. One should observe that the resultant controlled system is of third order, as indicated by (3.5-13). This is due to the fact that the system being controlled is second order and the digital computer is programmed to implement a first-order difference equation. In general, for the configuration shown in Figure 3.5-1, the order of the overall control system is equal to the sum of the orders of the system being controlled and the difference equation implemented by the digital computer.

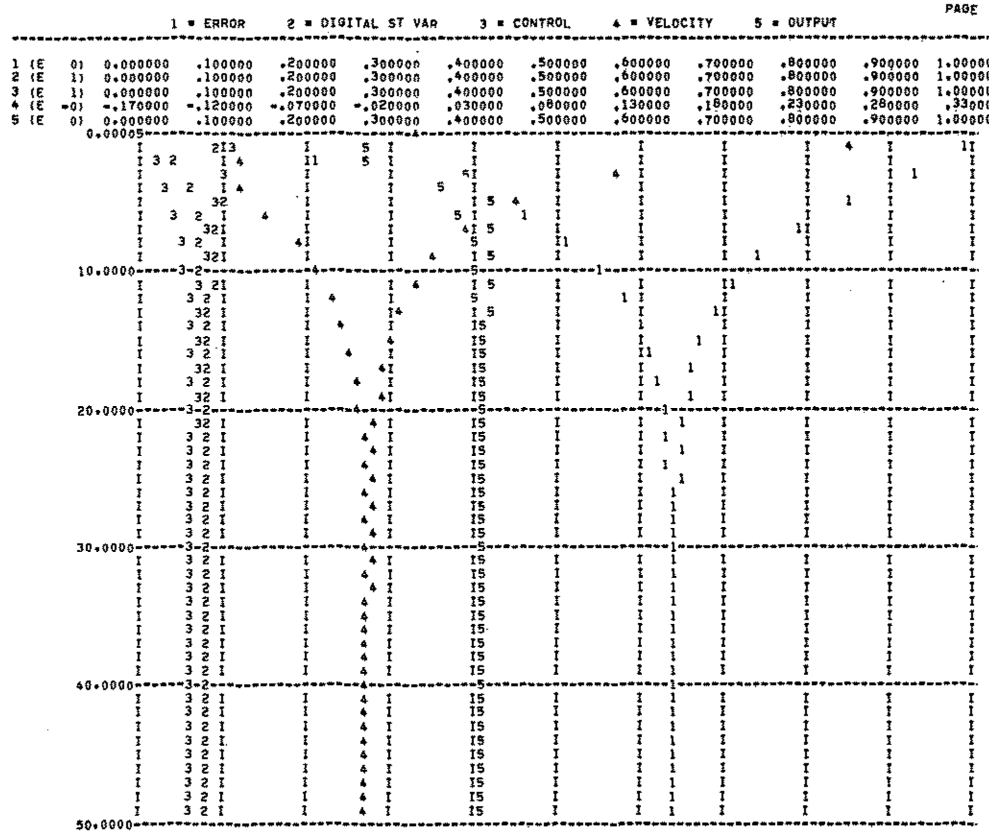


Figure 3.5-3.

A computer program has been prepared to compute the response of this system to a step input. Solved iteratively are equations (3.5-5), (3.5-7), and (3.5-10). The results are plotted out in Figure 3.5-3 on the computer-generated printer plot on which the variables are identified as follows:

Plot 1: Error $= e_1(k)$
Plot 2: Digital state variable $= x_3(k)$
Plot 3: Control $= e_2(k)$
Plot 4: Velocity $= x_1(k)$
Plot 5: Position $= x_2(k)$

The program used to generate the printer plot is explained in Appendix 3A.

```
    DIMENSION  X(5),  Y(2)
    READ  (5,14)  T,TLIMIT,  NPT,  NV
    READ  (5,13)  (HEAD(K),  K  =  1,8)
 14 FORMAT  (2F10.4,2I10)
```

```
13 FORMAT (8A10)
   R = 1.
   E1 = EXP(-T)
   E2 = EXP(-2.*T)
   A11 = E1+2.*E2
   A12 = -2.*E1+2.*E2
   A21 = E1-E2
   A22 = 2.*E1-E2
   B1 = E1-E2
   B2 = .5*(E1+E2)
   TIME = 0.
   DO 23 K = 1,NV
23 X(K) = 0.
   REWIND 1
   WRITE (6,15)
15 FORMAT (1H1,8X,4HTIME,9X,5HERROR,8X,10HDIG ST VAR, 7X,7HCONTROL,
  X 7X,X7X,8HVELOCITY,8X,6HOUTPUT)
49 WRITE (6,11) TIME,(X(I),I = 1,NV)
11 FORMAT (7(5X,F10.6))
   WRITE (1,12) TIME, (X(I),I = 1,NV)
12 FORMAT (7O20)
   IF (TIME.GT.TLIMIT) GO TO 51
   X(1) = R-X(5)-2.*X(4)
   X(2) = .25*X(2)+X(1)
   X(3) = -.1*X(2)+1.2*X(1)
   Y(1) = A11*X(4)+A12*X(5)+B1*X(3)
   Y(2) = A21*X(4)+A22*X(5)+B2*X(3)
   X(4) = Y(1)
   X(5) = Y(2)
   TIME = TIME+T
   GO TO 49
51 CALL PLOTX
   CALL EXIT
   END
```

By incorporating a digital computer in a control system, the design engineer has the capability of generating control characteristics not usually obtainable using standard control elements. Even if the iterative processes that the digital computer carries out are restricted to be linear, a wide design choice is available. Linear iterative processes may be programmed to perform such operations as differentiation, integration, filtering, prediction, etc.

To give a simple demonstration of this, assume that the digital computer in Figure 3.5-1 has been programmed to numerically differentiate the function $e_1(t)$. We shall use the iteration process developed in Section 1.4, namely

$$e_2(kT) = \frac{1}{T}[e_1(kT) - e_1(kT - T)] \tag{3.5-14}$$

To illustrate that this iteration actually has properties ascribed to the differentiation operation, let $e_1(t) = tu(t)$ so that

$$e_1(kT) = \begin{cases} kT & \text{for } k \geq 0 \\ 0 & \text{for } k < 0 \end{cases}$$

Assume that $e_2(-T) = 0$; that is, the discrete system represented by (3.5-14) is initially at rest. Evaluating (3.5-14) for $k = 0, 1, 2, \ldots$, we have

$$\begin{aligned} e_2(0) &= 0 \\ e_2(T) &= \frac{1}{T}[T - 0] = 1 \\ e_2(2T) &= \frac{1}{T}[2T - T] = 1 \\ &\cdots\cdots\cdots \\ e(kT) &= \frac{1}{T}[kT - T] = 1 \quad \text{for } k \geq 1 \end{aligned}$$

Thus, the output of the digital computer will be a sequence of ones, which, when fed into the zero-order hold circuit shown in Figure 3.5-1, results in

$$m(t) = u(t - T) = \begin{cases} 1 & \text{for } t \geq T \\ 0 & \text{for } t < T \end{cases}$$

An ideal differentiating network would have produced the output $m(t) = (d/dt)\,e_1(t) = u(t)$. Thus, for the input $e_1(t) = tu(t)$, the given iteration process in conjunction with a zero-order hold has characteristics normally associated with the differential operator. Again, this was a very simple example to illustrate the fact that a digital computer may be used to implement characteristics usually associated with analog elements. In addition, the digital computer may be programmed to perform extremely complex nonlinear operations that are difficult, if not impossible, to implement by analog elements. In summary, the inclusion of a digital computer in a control system opens up new avenues of design not available with standard analog components.

3.6 Stability of Discrete Systems

The stability of linear feedback control systems depends predominantly on the gain of the control loop, on the poles and zeros of the controlled system, on the magnitude of transportation lags, and perhaps on several other less important physical characteristics. A criterion for the stability of continuous-time systems consists of testing whether the eigenvalues of the system matrix or the closed-loop poles all have negative real parts. In the analysis of discrete-time systems one other important design parameter enters into the consideration of stability; this is the sampling period T. In Example 3.4-1, it was demonstrated that a stable sampled-data system can be made unstable by increasing the sampling period beyond a certain point.

Here we wish to consider a stability criterion for discrete-time systems. To this effect we consider the state equations of typical linear discrete systems.

$$\mathbf{x}[(k+1)T] = \mathbf{A}\mathbf{x}(kT) + \mathbf{B}\mathbf{r}(kT) \tag{3.6-1}$$

By an iterative approach we have shown that the solution is given by

$$\mathbf{x}(kT) = \mathbf{A}^k\mathbf{x}(0) + \sum_{n=0}^{k-1} \mathbf{A}^{k-1-n}\,\mathbf{B}\mathbf{r}(nT) \tag{3.6-2}$$

For the purpose of a stability analysis it is necessary to consider only the homogeneous solution to (3.6-1); that is, for $\mathbf{r}(nT) = \mathbf{0}$,

$$\mathbf{x}(k) = \mathbf{A}^k\mathbf{x}(0) \tag{3.6-3}$$

By the use of (3.6-3) we can establish a stability criterion for a discrete system. The matrix $\mathbf{A}^k$ is the state transition matrix $\mathbf{A}$ raised to the kth power.

If we assume that the eigenvalues of the state transition matrix are all distinct, then $\mathbf{A}^k$ may be expressed as a series by means of the Sylvester expansion theorem (see Appendix 2). Thus, if $\mathbf{A}$ is an $n \times n$ matrix with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, then

$$\mathbf{A}^k = \sum_{i=1}^{n} \mathbf{A}_i \lambda_i^k \tag{3.6-4}$$

where the matrices $\mathbf{A}_i$ are the constituent matrices of $\mathbf{A}$. Substituting (3.6-4) into (3.6-3) yields

$$\mathbf{x}(k) = \sum_{i=1}^{n} \mathbf{A}_i \lambda_i^k \mathbf{x}(0) \tag{3.6-5}$$

From this equation it is apparent that the sequence of vectors

$$\{\mathbf{x}(0),\quad \mathbf{x}(1),\quad \ldots,\quad \mathbf{x}(k),\quad \ldots\}$$

can converge to zero, for arbitrary $\mathbf{x}(0)$, only if the terms λ_i^k converge individually to zero. Therefore, for a discrete system to be stable, the eigenvalues of the state transition matrix must satisfy the following condition

$$|\lambda_i| < 1 \quad \text{for } i = 1, 2, \ldots, n \tag{3.6-6}$$

Although we have considered here only the case where all eigenvalues are distinct, it can be shown that the stability criterion (3.6-6) is general and applies to systems with any degree of multiple eigenvalues.

EXAMPLE 3.6-1

Test the stability of the system of Example 3.4-1 for $T = 1$ and $T = 4$. The state transition matrices for the two cases were found to be

(a) $T = 1$

$$\mathbf{A} = \begin{bmatrix} .368 & -.632 \\ .632 & .632 \end{bmatrix}$$

The eigenvalues are determined from the relation

$$|\lambda\mathbf{I} - \mathbf{A}| = \begin{vmatrix} \lambda - .368 & .632 \\ -.632 & \lambda - .632 \end{vmatrix} = \lambda^2 - \lambda + .632 = 0$$

which yields

$$\lambda_{1,2} = .5 \pm j\sqrt{.382} = .5 \pm j.625$$
$$|\lambda_1| = |\lambda_2| = \sqrt{(.5)^2 + (.625)^2} = .796$$

Since the absolute value of each eigenvalue is less than unity, the system is stable.

(b) $T = 4$

$$\mathbf{A} = \begin{bmatrix} .0183 & -.98 \\ .98 & -2.02 \end{bmatrix}$$

$$\begin{vmatrix} \lambda - .0183 & -.98 \\ .98 & \lambda + 2.02 \end{vmatrix} = \lambda^2 + 2.002\lambda + .95$$

Clearly, $|\lambda_1|$ and $|\lambda_2|$ are greater than one, although only by a small margin; thus, the system is unstable. This example effectively demonstrates that the selection of the sampling period is a critical element in determining a sampled system's stability. It also verifies the conclusion that we reached in Example 3.4-1, namely, that this system was stable for $T = 1$ but unstable for $T = 4$.

3.6-1 Regions of Stability

We have established that the location of the roots of the characteristic equation of a system can be used as a criterion of stability. For the sake of comparison we summarize here the results as they apply to continuous- and discrete-time systems.

	Continuous Time	*Discrete Time*
System equation:	$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{G}\mathbf{r}(t)$ $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{r}(t)$	$\mathbf{x}[(k+1)T] = \mathbf{A}\mathbf{x}(kT) + \mathbf{B}\mathbf{r}(kT)$ $\mathbf{y}(kT) = \mathbf{C}\mathbf{x}(kT) + \mathbf{D}\mathbf{r}(kT)$
System matrix:	$\mathbf{F}$	$\mathbf{A}$
Characteristic equation:	$\lvert \lambda\mathbf{I} - \mathbf{F} \rvert = 0$	$\lvert \lambda\mathbf{I} - \mathbf{A} \rvert = 0$
Stability criterion:	$\text{Re}\,\{\lambda_i\} < 0$	$\lvert \lambda_i \rvert < 1$
Region of stability:		

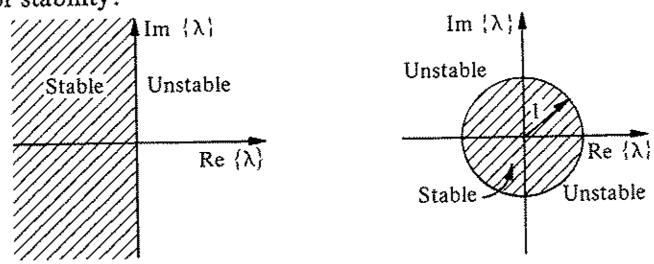


Figure 3.6-1. Regions of stability.

3.7 Analysis of a Digital Process Controller

The process control industry has a potentially large market for computer control technology. It is not unusual to find in a single process a substantial number of control loops, each one of which involves a single controlled variable such as temperature, flow, heat, etc. The process control industry has been using for many years the so-called PID controller, which provides proportional, integral, and derivative control action for a given loop. The configuration for this controller is illustrated in Figure 3.7-1, where it is shown as part of a typical control loop. Ideally, the controller is intended to provide the three control functions exactly as the transfer functions indicate. However, because of common physical limitations, neither the derivative nor the integral operation can be perfectly achieved.

Normally a PID controller is realized by analog means. We demonstrate now how one may replace the PID controller by a digital computer programmed to generate the functions of differentiation and integration numerically. The proposed digital control loop is shown in Figure 3.7-2.

The digital computer is programmed so that the output sequence is given by

$$e_2(k) = K_p e_{21}(k) + K_i e_{22}(k) + K_d e_{23}(k) \tag{3.7-1}$$

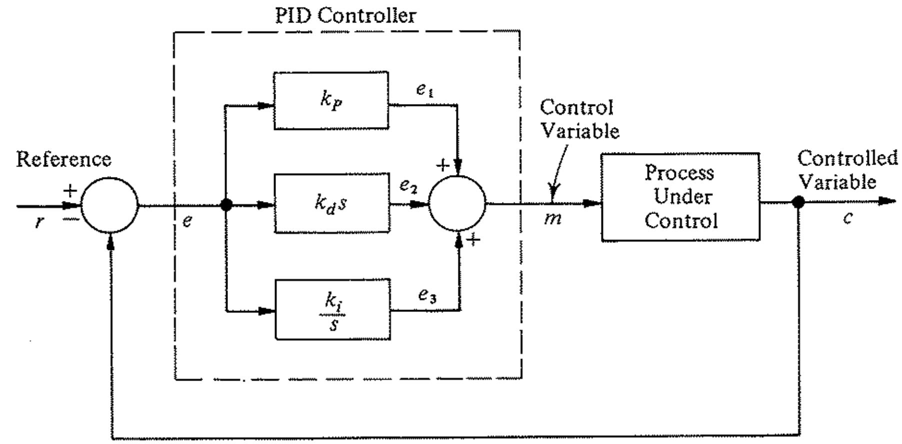


Figure 3.7-1. Typical PID control loop.

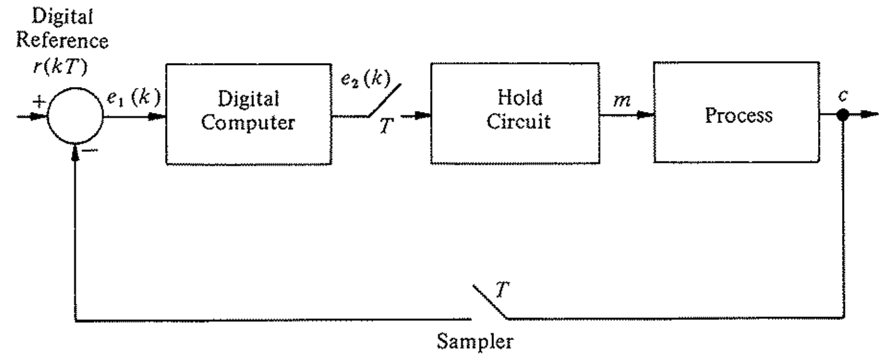


Figure 3.7-2. System with digital computer replacing PID controller.

The three components of the sum are generated as follows:

Proportional control

$$e_{21}(k) = e_1(k) \tag{3.7-2a}$$

Integral control

$$e_{22}(k) = e_{22}(k-1) + Te_1(k) \tag{3.7-2b}$$

Derivative control

$$e_{23}(k) = \frac{1}{T}[e_1(k) - e_1(k-1)] \tag{3.7-2c}$$

Other forms of numerical integration and differentiation could have been selected (e.g., see Chapter 9).

Although equations (3.7-2a) through (3.7-2c) may be readily programmed in their present form, it is desirable to obtain a discrete state model of the discrete PID controller in order to facilitate other analytical studies.

We recognize that the integral and derivative control are described by first-order linear difference equations of the general type

$$x_{\text{out}}(k) = b_0 x_{\text{in}}(k) + b_1 x_{\text{in}}(k-1) - a_1 x_{\text{out}}(k-1) \tag{3.7-3}$$

The state model according to the direct programming method corresponding to this equation is given by

$$\begin{aligned} x(k+1) &= -a_1 x(k) + r(k) \\ y(k) &= (b_1 - b_0 a_1)x(k) + b_0 r(k) \end{aligned} \tag{3.7-4}$$

Applying equation (3.7-4) to the integrator, we have $a_1 = -1$, $b_0 = T$, $b_1 = 0$, so

$$\begin{aligned} x_i(k+1) &= x_i(k) + e_1(k) \\ e_{22}(k) &= Tx_i(k) + Te_1(k) \end{aligned} \tag{3.7-5}$$

where the subscript i denotes integration. Similarly, for the differentiator we have $a_1 = 0$, $b_0 = 1/T$, $b_1 = -1/T$, so

$$\begin{aligned} x_d(k+1) &= e_1(k) \\ e_{22}(k) &= -\frac{1}{T}x_d(k) + \frac{1}{T}e_1(k) \end{aligned} \tag{3.7-6}$$

where the subscript d denotes differentiation.

The complete state model can now be assembled from equations (3.7-2a), (3.7-5), and (3.7-6).

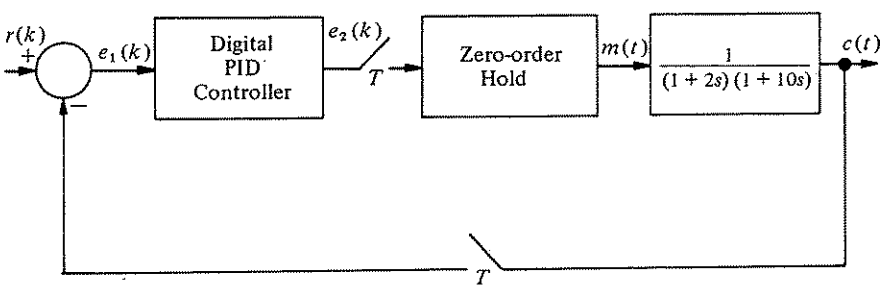


Figure 3.7-3. Digital control loop.

$$\begin{aligned} \begin{bmatrix} x_i(k+1) \\ x_d(k+1) \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_i(k) \\ x_d(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} e_1(k) \\ e_2(k) &= \begin{bmatrix} T & -\frac{1}{T} \end{bmatrix} \begin{bmatrix} x_i(k) \\ x_d(k) \end{bmatrix} + \left(K_p + K_i T + K_d \frac{1}{T}\right) e_1(k) \end{aligned} \tag{3.7-7}$$

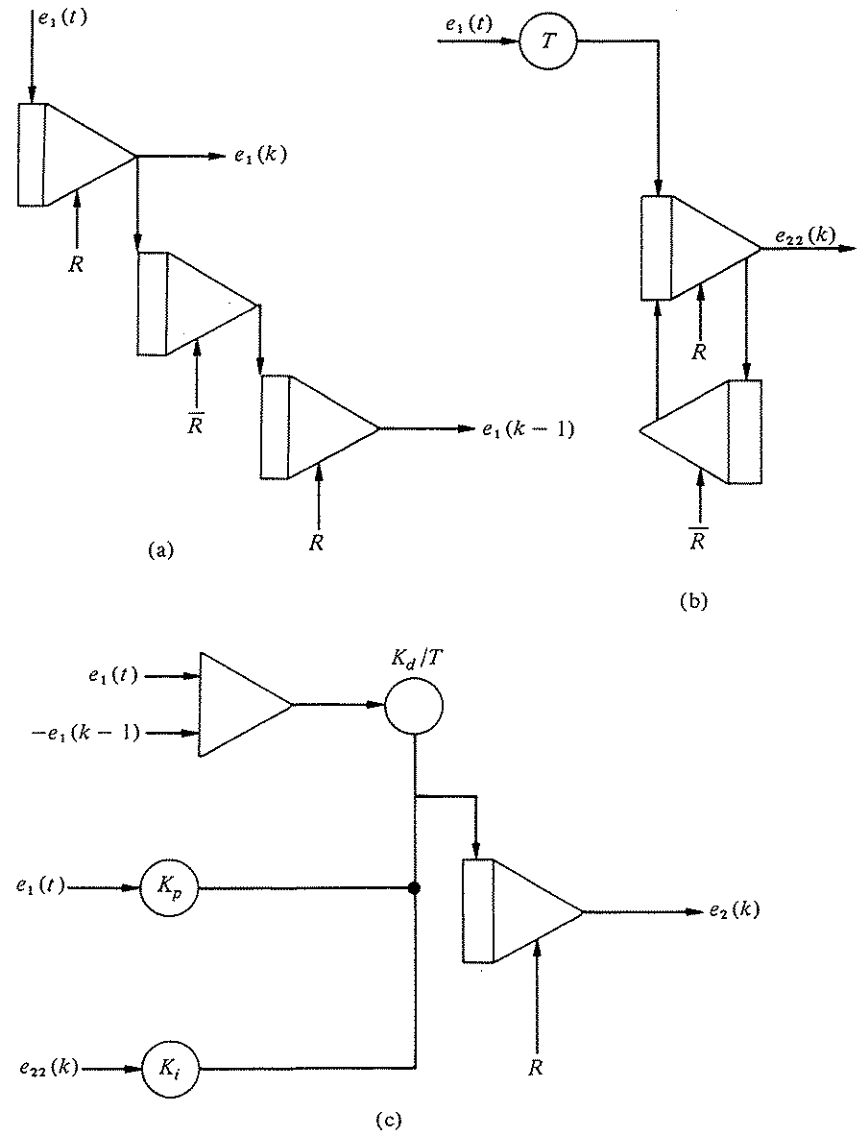


Figure 3.7-4. Analog-hybrid implementation of discrete PID controller.

Consider now the application of this digital controller to the control loop shown in Figure 3.7-3. In order economically to justify the employment of a digital computer for the control of a process, it would seem reasonable that the computer control a great number of loops on a time-shared basis. This is possible if the computer is switched through a multiplexor periodically to each of the control loops. The computer's high speed will permit the execution of the program for this control loop in less than 100 microseconds. It is apparent that, for example, if the sampling period is about one second, easily 1000 such loops can be handled by a single computer without encountering time problems.

In the application of a digital computer as a PID controller, the quantities K_p, K_i, K_d, and T must be determined. It has been the practice in the process control industry to adjust the gain constants on the job until a desirable response is obtained. Alternatively, a simulation of the entire system may be performed which permits parameter adjustment.

A hybrid computer is ideally suited for this task. The plant may be simulated on the analog computer, while the general-purpose digital computer is

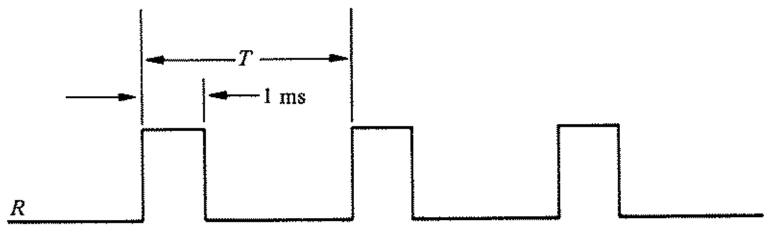


Figure 3.7-5. Timing signals for R.

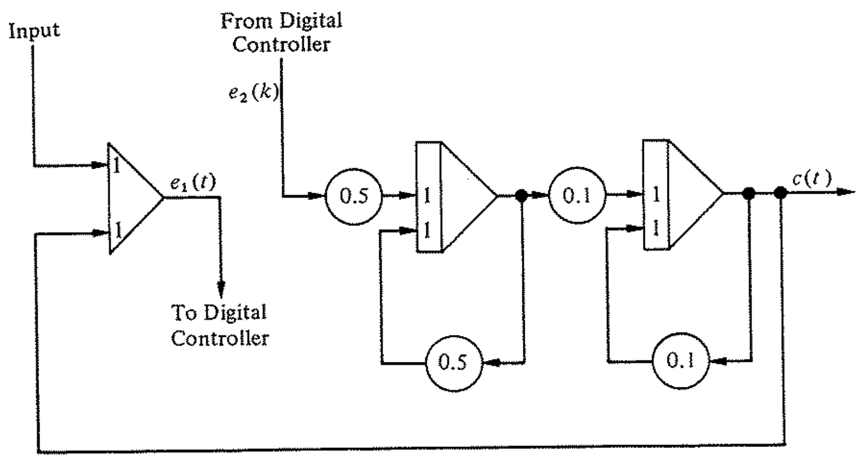


Figure 3.7-6. Simulation of process.

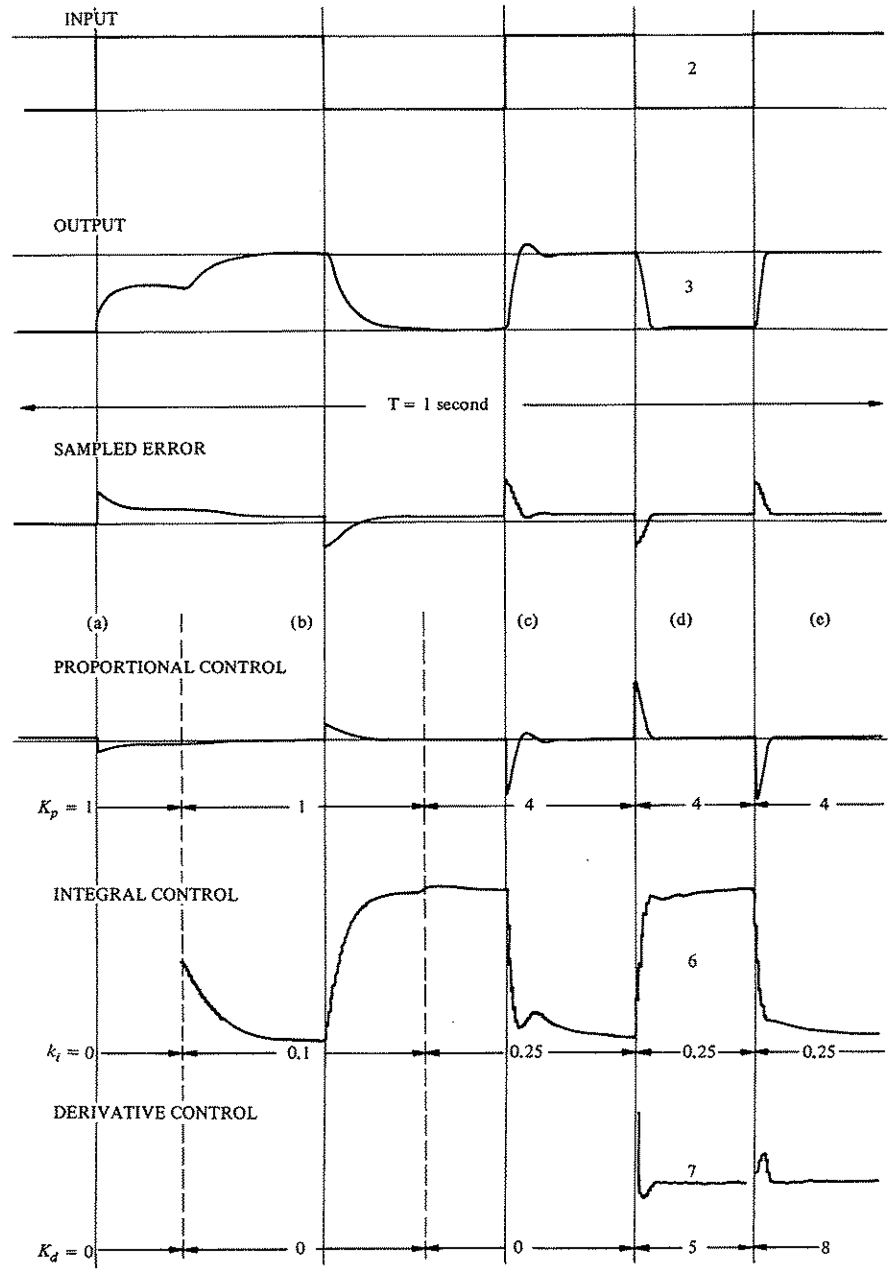


Figure 3.7-7.

available to program the PID controller. The hybrid computer also has the necessary data converter to transform signals from analog to digital form and vice versa. The programming of this problem by a hybrid computer is explained in Chapter 9.

One may also use an analog computer with discrete operation capability, called an analog/hybrid computer. This type of computer is equipped with digital logic timing networks with whose support integrators may be operated as discrete memory devices. Such a computer is used here to simulate the system.

The digital PID controller is simulated as shown in Figure 3.7-4. Figure 3.7-4(a) contains the diagram for the generation of $e_1(k)$ and $e_1(k-1)$ by use of two track-store units that are timed in a complementary fashion. Figure 3.7-4(b) shows the circuit that implements the integrator. By means of an analog accumulator, the integrator difference equation is generated. Finally, Figure 3.7-4(c) shows the generation of the output by summing the various signals making up equation (3.7-1).

The timing signal R is of the shape shown in Figure 3.7-5. It controls the RESET mode. It is understood that the OPERATE mode is controlled by the complement of RESET. The process to be controlled is programmed according to the diagram of Figure 3.7-6.

Step responses for various control settings are shown in Figure 3.7-7 in the form of a six-channel recording. Part (a) shows the response of the system for a sampling rate of one second and only proportional control. Since the process is a type 0 process and $K_p = 1$, the steady-state error is equal to 50 percent of the input. In part (b) the result of introducing integral control is an elimination of the steady-state error; a complete response for the settings ($T = 1$, $K_p = 1$, $K_i = .1$) is shown. In part (c) the gain settings for K_p and K_i have been increased to 4 and .25, respectively; the response now exhibits a definite overshoot. This overshoot is eliminated by the introduction of derivative control for which two runs are shown with gain settings of $K_d = 5$ and $K_d = 8$.

3.8 Response of Sampled Systems between Sampling Instants

It is evident by now that the analytical techniques developed in the preceding section for the analysis of sampled-data systems provide information on the system variables only at the sampling instants. Thus, even though the system to be analyzed has a continuous-time output such as a computer control system, the output is known only at discrete times synchronous with the sampling period. Quite frequently, however, it is important to know the time

history of the continuous-time variables for more than the sampling instants. This is actually quite easily accomplished. Consider the response of a linear system via state variable techniques. Given the arbitrary initial conditions $x(t_0)$ at t_0, we may write

$$\mathbf{x}(t) = \mathbf{A}(t - t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{A}(t - t_1)\mathbf{B}\mathbf{x}_{\text{in}}(t_1)dt_1 \tag{3.8-1}$$

where $\mathbf{x}_{\text{in}}(t)$ is the input to the system.

If this linear system represents the continuous time part of a sampled system such as the one shown in Figure 3.3-2, then we identify $\mathbf{x}_{\text{in}}(t)$ with $m(t)$ and

$$m(kT + \tau) = r(kT), \qquad 0 \leq \tau < T \tag{3.8-2}$$

where $r(kT)$ is the discrete input to the zero-order hold.

Letting $t_0 = kT$ and $t = kT + \tau$, $0 \leq \tau < T$, we write (3.8-1) as

$$\mathbf{x}(kT + \tau) = \mathbf{A}(\tau)\mathbf{x}(kT) + \int_{kT}^{kT+\tau} \mathbf{A}(kT + \tau - t_1)\mathbf{B}r(kT)dt_1 \tag{3.8-3}$$

The integral was shown in Chapter 2 to simplify to

$$\int_{kT}^{kT+\tau} \mathbf{A}(kT + \tau - t_1)\mathbf{B}r(kT)dt_1 = \left\{\int_0^{\tau} \mathbf{A}(t_1)\mathbf{B}dt_1\right\}r(kT) = \mathbf{B}(\tau)r(kT)$$

Therefore, (3.8-3) reduces to

$$\mathbf{x}(kT + \tau) = \mathbf{A}(\tau)\mathbf{x}(kT) + \mathbf{B}(\tau)r(kT), \qquad 0 \leq \tau < T \tag{3.8-4}$$

This equation may be used to calculate the response of any continuous time part in a sampled system during the sampling interval, provided that the state variables are known at the beginning of the interval and that the input is held constant during the interval. This equation may be best put to work by using computer techniques to evaluate it. This is shown in Chapter 9, Section 9.2.

REFERENCES

1. Dorf, R. C., *Time-Domain Analysis and Design of Control Systems*, Addison-Wesley, Reading, Mass., 1964.
2. Kuo, B. C., *Linear Networks and Systems*, McGraw-Hill, New York, 1967.
3. Monroe, A. J., *Digital Processes for Sampled-Data Systems*, Wiley, New York, 1963.