DIGITAL CONTROL THEORY
With Mobile Robot Applications

Won-Sang Ra
Handong Global University

HGU 한동대학교
HANDONG GLOBAL UNIVERSITY

CONTENTS

CHAPTER 1

INTRODUCTION TO DIGITAL CONTROL SYSTEMS

In recent years significant process has been made in the analysis and design of discrete-data and digital control systems. These systems have gained popularity and importance in industry due in part to the advances made in digital computers for controls and, more recently, in microprocessors and digital signal processors (DSP). Digital control systems differ from the conventional analog control systems because the signals in one or more parts of these systems are in the form of either pulse trains or numerical codes. The terms *sampled-data control systems, discrete-data control systems,* and *digital control systems* have all been used loosely and interchangeably in the control systems literature.

1.1 Basic Elements and Advantages of Digital Control Systems

Figure 1.1 shows the basic elements of a conventional closed-loop control system with sampled data. The sampler simply represents a device or operation that outputs a pulse train. No information is transmitted between two consecutive pulses. Figure 1.2 illustrates the input and output of a sampler. A continuous input signal $e(t)$ is sampled by the sampler, and the output is a sequence of pulse. In the illustrated

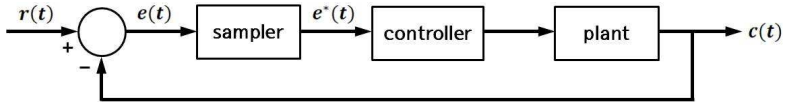

Figure 1.1 Closed-loop sampled data control system

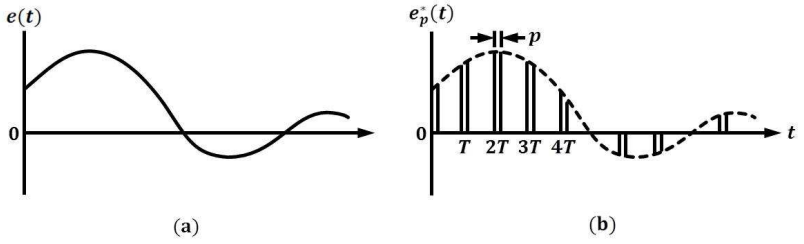

Figure 1.2 Input and output signals of a sampler

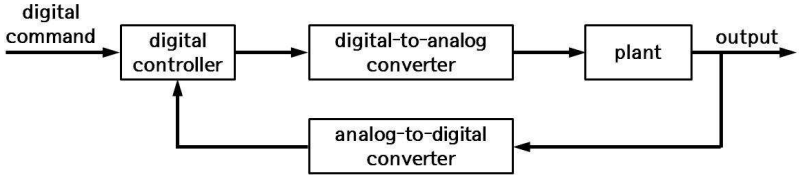

Figure 1.3 Digital control system

case, the sampler is assumed to have a uniform sampling rate. The magnitudes of the pulses at the sampling instants represent the values of the output signal $e(t)$ at the corresponding instants.

The block diagram of a typical digital control system is shown in Figure 1.3. The existence of digitally coded signals, such as binary-coded signals, in certain parts of the system requires the use of digital-to-analog (D/A) and analog-to-digital (A/D) converters. The digital computer block in Figure 1.3 can be a personal computer, an embedded computer with real-time operating system, a microprocessor, or a digital signal processor. To understand the merits and advantages of using sampling and digital data in control systems, one should ask, *Why sampled data and digital control?* or *What are the advantages and characteristics of sampled data and digital control?* We must first recognize that many physical systems have inherent sampling or their behavior can be described by discrete-data or digital models. For instance, in a radar tracking system the signals transmitted and received by the system are in the form of pulse trains. The scanning operation of the radar performs the function of a sampler, converting both the azimuth and the elevation information to sampled data. To address the advantages and characteristics of discrete-data systems, we must look at the characteristics of continuous-data control systems which have been in existence for many years.

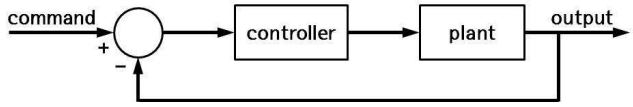

Figure 1.4 Analog control system

Figure 1.4 shows the block diagram containing the basic elements of a single-loop continuous-data control system. The controller is usually an electronic circuit that operates on an analog signal and outputs the same type of signal. The advantage with the analog controller is that the system operates in real time and is capable of a very high bandwidth. It is equivalent to having an infinite sampling frequency, so that the controller is effective at all times. The disadvantages of the analog controller are that its elements are usually hard-wired, so that their characteristics are fixed, making it more difficult to make design changes, and component aging and sensitivity to environmental changes can quite severe. Analog components are also more susceptible to noise problems.

Advantages of digital control systems

- Digital components are less susceptible to aging and environmental variations. They provide improved sensitivity to parameter variations.
- Digital devices are more reliable because they are less sensitive to noise and disturbance.
- Digital processors allow more flexibility in programming: changing a design does not require an alteration in the hardware.
- Digital processors are more compact and lightweight. Single-chip processors can be and very versatile and powerful for control applications.

Disadvantages of digital control systems

- Limitations on computing speed and signal resolution due to the finite wordlength of the digital processor. In contrast, analog controllers operate in real-time, and the resolution is theoretically infinite.
- The finite wordlength of the digital processor often causes instability of the closed-loop system in the form of limit cycles.
- The time delay occurs because of the limitation on computing speed. Furthermore, it may also cause instability in closed-loop systems for the lack of phase margin.

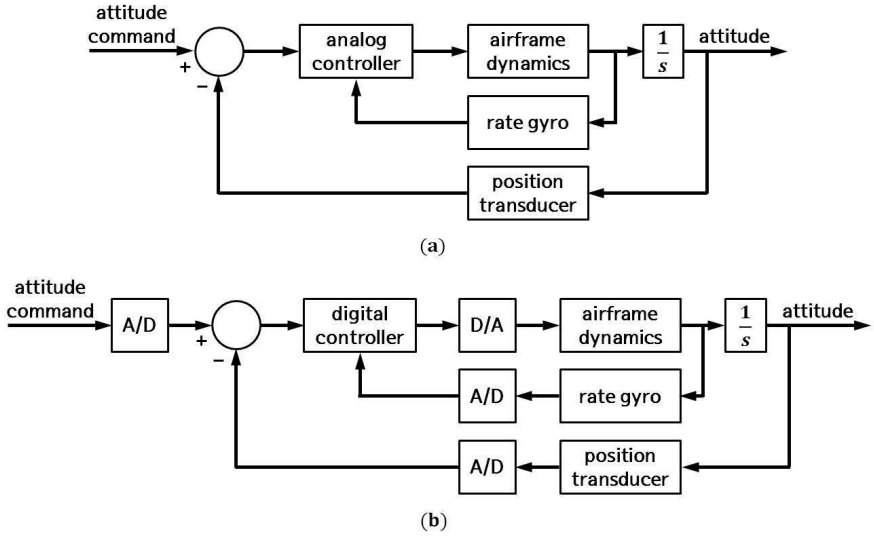

Figure 1.5 Analog and digital roll autopilots

1.2 Examples of Digital Control Systems

Figure 1.5 shows the block diagram of a simplified analog roll autopilot of an unmanned aerial vehicle. This is a continuous-data control system in which the signals are represented as functions of the continuous-time variable t. The objective of the control is that the attitude of the airframe follow the command signal. The rate loop is incorporated here for the improvement of system stability. Instead of using the analog controller as shown in Figure 1.5, a digital controller with the necessary A/D and D/A converters can be used for the same objective, as shown in Figure 1.5(b). Since all the components of the system other than the digital controller are still analog, the A/D and D/A converters are necessary for signal conversions. In Figure 1.5(b), the position and rate information are obtained by digital transducers, and the operations are represented on the block diagram by sample-and-hold(S/H) devices associated with the D/A. The A/D picks up the analog signal at some uniform sampling rate and the hold device maintains the value of the sampled signal until the next sample comes along.

CHAPTER 2

DIGITAL SIGNAL CONVERSION

Most digital control systems usually contain analog signals as well as digital signals. Therefore, the process of signal conversion is essential so that the digital and analog components can be interfaced in the same system. For instance, the output signal of an analog device, such as analog sensor, must undergo an analog-to-digital conversion before the signal can be processed by a digital controller. Sometimes the analog-to-digital conversion process may involve an encoding operation in that the signal is converted into some digital code. Similarly, the digitally coded signal from a digital controller or processor must be decoded by a digital-to-analog converter (D/A) before it can be sent to an analog device for processing. In fact, many microprocessors designed for control purposes have on-board analog-to-digital converters (A/D) for data conversion. The signal from a digital processor must be smoothed out after decoding by a signal-reconstruction device such as sample-and-hold (S/H) or low-pass filter before being sent to the analog process.

Since these components are important for signal processing and signal conditioning in digital control systems, they will be described and modeled in this chapter. The objective of the introductory treatment of A/D, D/A, and S/H is to establish the importance of digital signal processing and/or digital control systems. Furthermore, this chapter gives an answer to the question how these components are mathemati-

cally modeled for analysis and design of digital control systems.

The brief explanations about the above mentioned signal conversion devices are as follow:

Analog-to-digital converter (A/D) is a device which converters an analog signal to a digital-coded signal. That is, A/D quantizes and encodes a analog signal to produce the corresponding discrete-time signal (or digitally coded signal).

Digital-to-analog converter (D/A) performs the task of decoding on a digitally coded input. The output of the D/A is an analog signal, usually in the form of a voltage or a current.

Sample-and-hold device (S/H) is used for many purposes in digital control systems. The S/H makes a fast acquisition of an analog signal and then holds this signal at a constant value until the next acquisition (sample) is made. An S/H is often a part of an A/D.

2.1 Data Conversion and Quantization

Signals in digital computers are represented by digital codes or words. The information carried by the digital code is generally in the form of discrete bits (logic pulses of '0' and '1'). Since it is simple to distinguish just two states, 'on(1)' and 'off(0)', all modern digital computers are designed on the basis of the binary number system. A digital signal can be stored in a digital computer as a binary number of

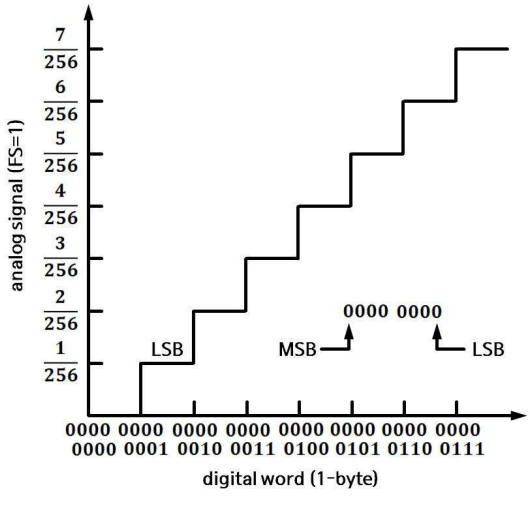

Figure 2.1 Relation between digital binary code and decimal number

zeros and ones. Each of the binary digits (0 or 1) is referred to as a *bit*. The bit itself, however, is too small to be considered as the basic unit of information. To resolve this problem, bits are strung together to form larger, more useful units; 8 bits placed together form a *byte*, and several of the 8-bit bytes are grouped together to form a *word*. In general, a word may be of almost any bit length, form 4 bits to 128 bits or more. The numerical value of the digital word or code then represents the magnitude of the information in the variable. Figure 2.1 illustrates the relation between *word*, *byte*, and *bit*. In this case, the length of the word illustrated is 1 byte or 8 bits. The accuracy of a digital computer in its ability to store and manipulate digital signals is indicated by its *wordlength*. For example, a microprocessor with an 8-bit word can store only numbers with 8-bit of accuracy in its memory. That is, digital signals in a microprocessor can be represented as *fixed-point numbers*.

In D/A and A/D conversions, the *most-significant bit* (MSB) and *least-significant bit* (LSB) and the weight of each in a digitally coded word are important in the understanding of the conversion process. An *n*-bit binary word defines 2^n-distinct levels of the digital signal. Thus the word provides a *resolution* of one part in 2^n levels. D/A and A/D make use of the binary code. In general, for an n-bit binary code and the given full-scale(FS), the MSB has a weight of (1/2 FS), but the LSB has a weight of 2^{-n} FS. *An n-bit binary word defines 2^n distinct states, thus the word provides a resolution of 2^{-n} FS.* For example, Figure 2.1 shows that a 8-bit binary code has the resolution of $2^{-4} = 1/256$ for FS$= 1$. The LSB in this case corresponds to the level of $1/256$. Therefore, from this relation, it is obvious that the resolution is closely related to the LSB. However, the knowledge on the FS is necessary to find the analog signal from the given digitally coded signal. Keep in mind that, *to improve the resolution of signal conversion, we should increase the number of bits for the same FS.*

If the number of bits in the digital word is finite, only a finite resolution can be attained by the A/D conversion; since the digital output can assume only a finite number of levels, the analog number is inevitably rounded off or truncated into a digital number. This approximation procedure is generally referred to as *quantization*. As shown in Figure 2.2 and 2.3, there are two kinds of quantization methods; round-off and truncation. In the figures, the parameter q, which is equal to the LSB, is known as the quantization level. Dotted lines represent the ideal outputs if there is no quantization characteristic is the *quantization error*. Thus, for the truncated quantization, the maximum quantization error is $\pm q$, whereas for the round-off quantization, the maximum error is $\pm q/2$. As an illustrative example, if the full-scale reference voltage of a 3-bit A/D conversion with round-off quantization is set at $10[V]$, referring to Figure 2.2, FS$= 10[V]$, and the quantization level is

$$q = \frac{1}{8}\text{FS} = 1.25[V]$$

Note that, in this case, the round-off quantization error will be within $\pm q/2 = \pm 0.625[V]$.

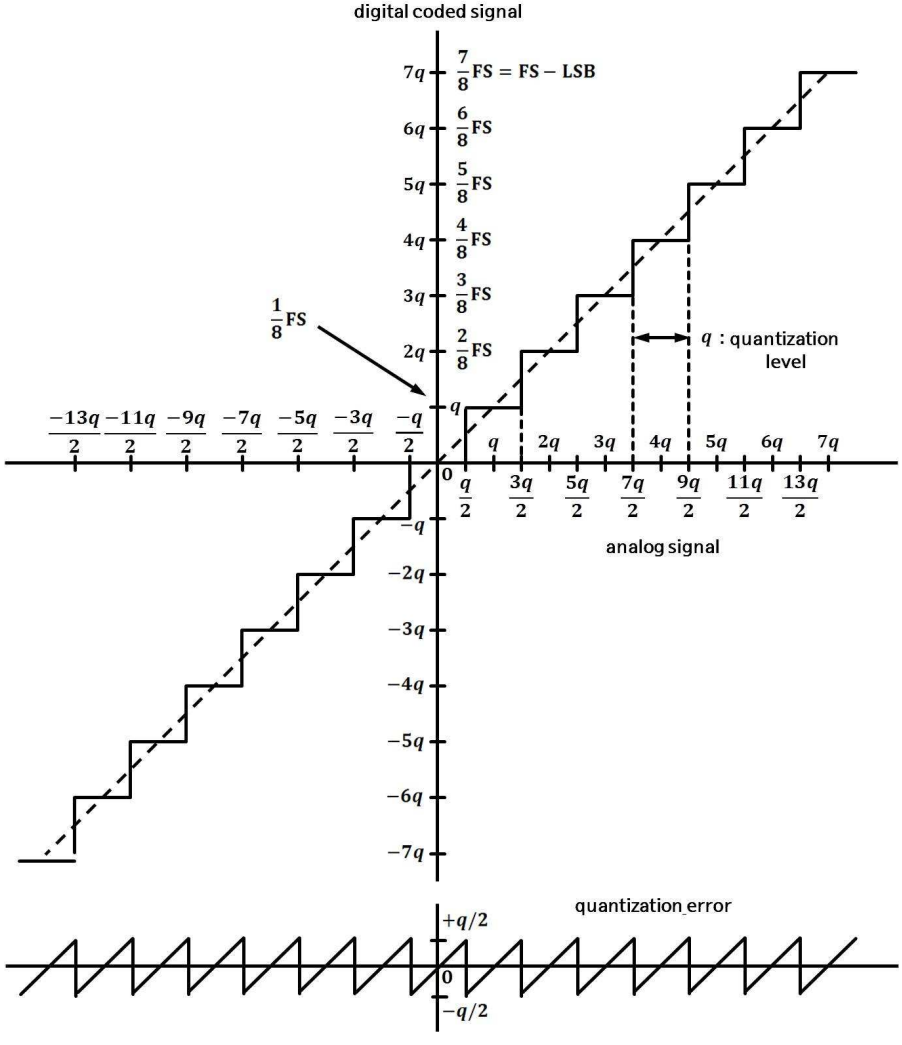

Figure 2.2 Input-output relationship of an A/D 3-bit round-off quantizer

2.2 Sample-and-Hold

Sample-and-hold devices are used extensively in digital control systems. A *sampler* is a device that convert an analog signal into a train of amplitude-modulated pulses or a digital signal. A *hold device* simply maintains or freezes the value of the digital signal for a prescribed time duration. In a majority of the practical digital operations, S/H operations are performed by a single unit.

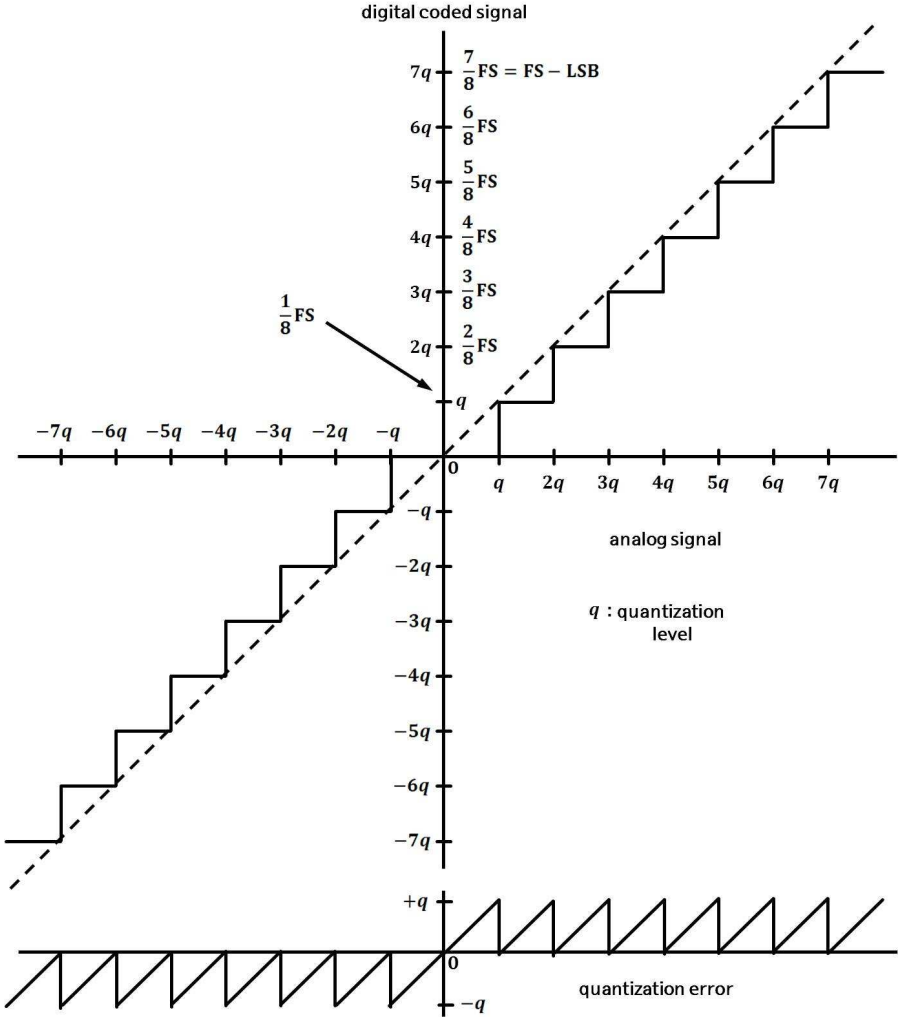

Figure 2.3 Input-output relationship of an A/D 3-bit truncation quantizer

The operation of a practical S/H device is conceptually illustrated by the circuit shown in Figure 2.4. The opening and closing of the switch or sampler are controller by a *clock* or a *command*. When the switch is closed, the S/H device *samples* and *tracks* the input signal $e_s(t)$. When the switch is opened, the output is held at the voltage that the capacitor is charged. Figure 2.5 illustrates typical input and output signals of the practical S/H. The time duration between the sample commands is called the sampling period T.

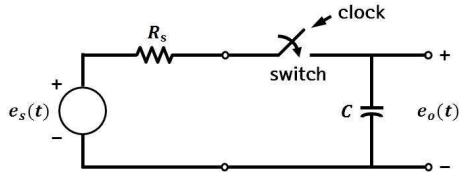

Figure 2.4 Simple circuit illustrating the sample-and-hold principle

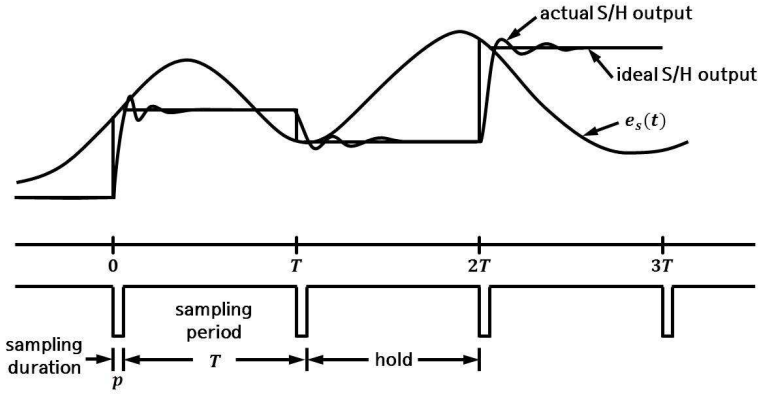

Figure 2.5 Input and output signals of an actual S/H device

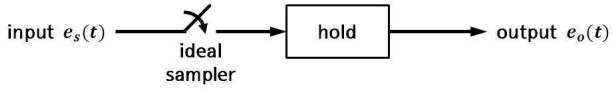

Figure 2.6 Ideal S/H device

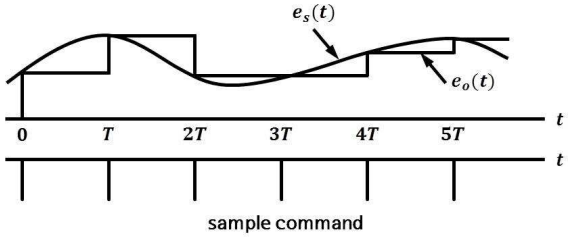

Figure 2.7 Input and output signals of an ideal S/H device

Without loss of generality, if the sampling duration is small enough, the S/H can be simply modeled by the block diagram shown in Figure 2.6. Figure 2.7 shows typical input and output waveform of this ideal S/H. We shall show later that the ideal sampler model leads to convenient mathematical modeling of the S/H operation.

2.3 Ideal Sampler

In general, the operation of a sampler may be regarded as one which converts a continuous-time signal into a digital signal.

The most common type of modulation required for actual S/H operation, as discussed in the preceding sections, is the *pulse-amplitude modulation* (PAM). Figure 2.8 shows the block diagram representation of a periodic sampler with finite sampling duration. The pulse or sampling duration is p[sec], and the sampling be considered if the value of p is not negligible when compared with the sampling period T. There are also physical systems that have natural properties, which need to be modeled by a *finite-pulsewidth sampler*. Consider that the input to the practical sampler is a continuous-time function $f(t)$. The output of the sampler, denoted as $f_p^*(t)$, is a train of finite-width pulses whose amplitudes are modulated by the input $f(t)$. Figure 2.9 shows an equivalent block diagram representation of the sampler as a pulse-amplitude modulator. The input $f(t)$ is considered to be multiplied by a carrier signal $p(t)$ which is a pulse train with unit amplitude. Figure 2.10 illustrates typical waveforms of the input signal $f(t)$ and the output signal $f_p^*(t)$ of the sampler in practice.

An ideal sampler is defined as a sampler which closes and opens instantaneously, every T seconds, for zero time duration. For such case, the carrier signal $p(t)$ in Figure 2.10 is replaced by the unit impulse train when the sampling duration is very small.

$$\delta_T(t) = \sum_{k=0}^{\infty} \delta(t - kT) \tag{2.1}$$

Therefore, the output of the ideal sampler is expressed as

$$f^*(t) = f(t)\delta_T(t) = \sum_{k=0}^{\infty} f(kT)\delta(t - kT) \tag{2.2}$$

where $f(t)$ is the input of the sampler and the sampling is assumed to begin at $t = 0$.

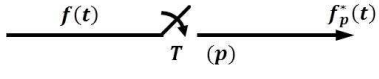

Figure 2.8 Uniform-rate sampler with finite sampling duration

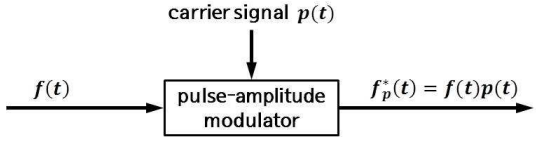

Figure 2.9 Pulse-amplitude modulator as a practical sampler

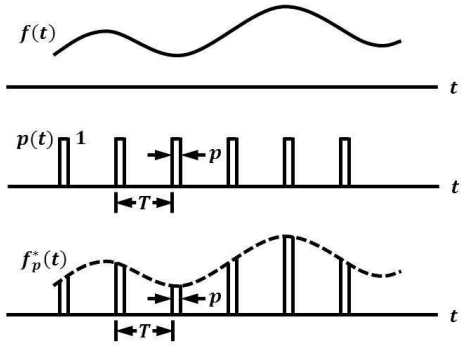

Figure 2.10 Input and output waveforms of a uniform-rate sampler

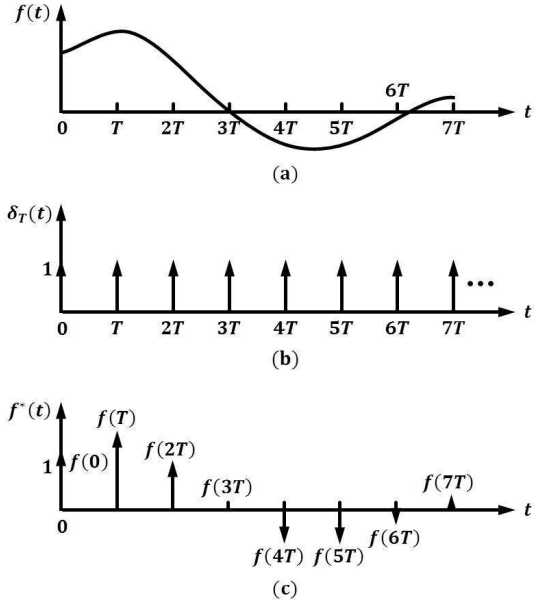

Figure 2.11 Input and output waveforms of the ideal sampler

Taking the Laplace transform on both sides of the above equation, we have

$$F^*(s) = \mathcal{L}\{f^*(t)\} = \sum_{k=0}^{\infty} f(kT)e^{-kTs}. \tag{2.3}$$

Typical input and output signals of an ideal sampler are illustrated in Figure 2.11. The output of the ideal sampler is shown to be a train of impulses with the respective areas (strengths) of the impulses equal to the magnitudes of the input signal at the corresponding sampling instants. Since an impulse function has zero pulsewidth and infinite pulse amplitude, in Figure 2.11 the impulses are depicted as arrows.

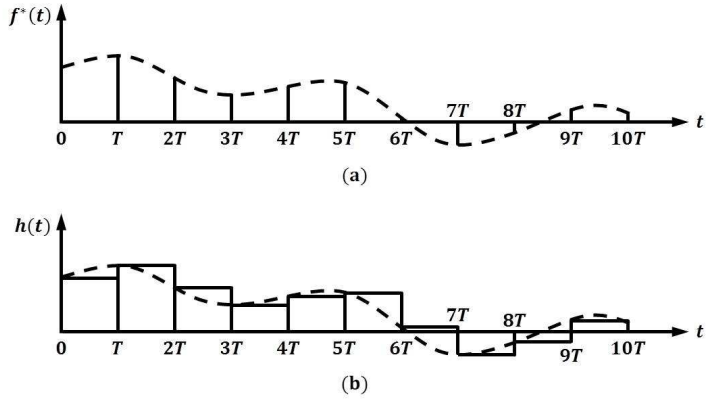

Figure 2.12 Zero-order-hold operation in time domain

2.4 Zero-Order Hold

The *zero-order hold* (ZOH) maintains the value of sampled data $f(kT)$ for $kT \leq t < (k+1)T$ until the next sample $f((k+1)T)$ arrives. The ZOH can be used for the hold portion of the S/H. Since the ZOH is a linear device, it satisfies the principle of superposition. The impulse response of ZOH is expressed as

$$g_{h0}(t) = u(t) - u(t-T) \tag{2.4}$$

where $u(t)$ indicates the unit-step function.
The transfer function of the ZOH is then obtained by taking the Laplace transform of (2.4).

$$G_{h0}(s) = \mathcal{L}\{g_{h0}(t)\} = \frac{1-e^{-Ts}}{s} \tag{2.5}$$

Figure 2.12 illustrates the waveform of the output of the ZOH, $h(t)$, when its input is a typical pulse sequence $f(kT)$ produced by the ideal sampler. The waveforms in Figure 2.12 clearly indicate that the accuracy of the ZOH as an extrapolating device depends greatly on the magnitude of the sampling frequency $\omega_s = \frac{2\pi}{T}$. As the sampling frequency increases to infinity or the sampling period T approaches zero, the output of the ZOH $h(t)$ approaches the continuous-time signal $f(t)$.

Since the ZOH is a data-reconstruction device, it is of interest to examine its frequency-domain characteristics. Replacing s by $j\omega$ in (2.5), we get

$$G_{h0}(j\omega) = \frac{1-e^{-j\omega T}}{j\omega} = T\frac{\sin(\omega T/2)}{(\omega T/2)}e^{-j(\omega T/2)} = \frac{2\pi}{\omega_s}\frac{\sin(\pi\omega/\omega_s)}{(\pi\omega/\omega_s)}e^{-j(\pi\omega/\omega_s)} \tag{2.6}$$

The magnitude and phase of $G_{h0}(j\omega)$ are computed as follows:

$$|G_{h0}(j\omega)| = \frac{2\pi}{\omega_s}\left|\frac{\sin(\pi\omega/\omega_s)}{(\pi\omega/\omega_s)}\right|, \quad \angle G_{h0}(j\omega) = \angle\sin(\pi\omega/\omega_s) - (\pi\omega/\omega_s). \tag{2.7}$$

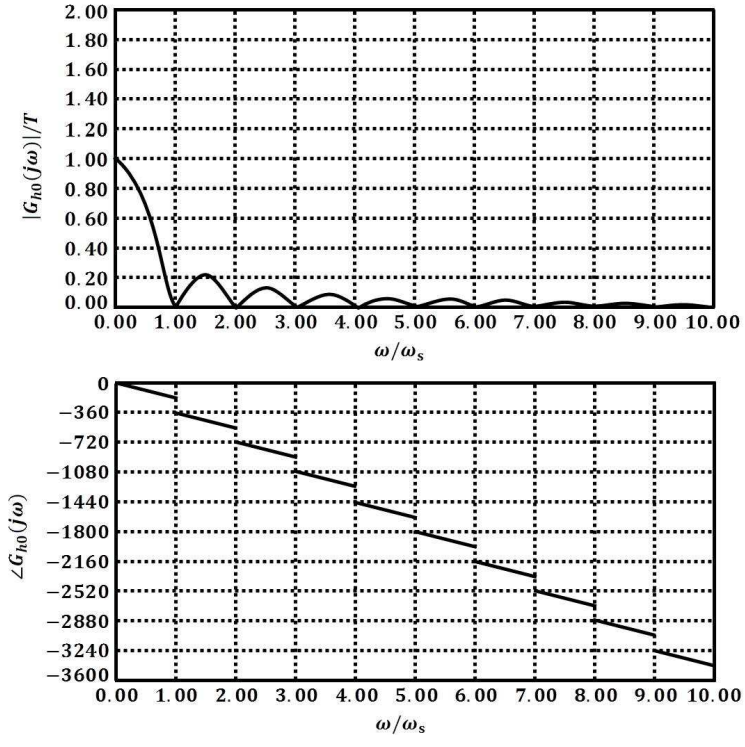

Figure 2.13 Gain and phase characteristics of the zero-order hold

The sign of $\sin(\pi\omega/\omega_s)$ may change for every integral value of $(\pi\omega/\omega_s)$. We can regard the change of sign from $+$ to $-$ as a phase change of $-180°$.

Based on these results, the bode plot of $G_{h0}(s)/T$ is depicted in Figure 2.13. The magnitude of $G_{h0}(s)/T$ is zero at $\omega = \omega_s$ and is 0.636 at $\omega = \omega_s/2$. The phase characteristic of $G_{h0}(s)$ is linear over the frequency intervals of $k\omega_s \leq \omega < (k+1)\omega_s,\ k = 0, 1, \cdots$, with jump discontinuities of $-180°$ at integral multiples of ω_s. Note that the frequency response of the ZOH is different from that of the ideal low-pass filter with a cut-off frequency at $\omega = \omega_s/2$. Although the ideal low-pass filter is known as the best interpolator in view of signal reconstruction, it cannot implemented in practice because its impulse response is anti-causal. On the contrary, the ZOH whose impulse response (2.4) is causal. This is the reason why we often use the ZOH as a viable alternative for signal reconstruction.

2.5 Digital-to-Analog Conversion

The *digital-to-analog* conversion, or simply decoding, consists of transforming the numerical information contained in a digitally coded signal into an equivalent analog signal. The basic elements of a D/A are portrayed by the block diagram

in Figure 2.14. The function of the logic circuit is to control the switching of the precision reference voltage or current source to the proper input terminals of the resistor network as a function of the digital value of each digital input bit.

Figure 2.15 illustrates a simple 3-bit binary D/A. The values of the summing resistors of the operational amplifier are weighted in a binary fashion. Each of these resisters is connected through an electronic switch to the reference voltage or the ground. When a binary 1 appears at the control logic circuit of a switch, it closes the switch and connects the resistor to the reference voltage. On the other hand, a binary 0 connects the resistor to ground. For the high-gain operational amplifier, the input impedance is very low so that the voltage at the summing points is practically zero (virtually grounded). Table 2.5 gives the output voltages of the D/A conversion corresponding to all the 3-bit binary words for the FS of $10[V]$.

If the similar property is applied for a n-bit D/A converter, the output voltage is written as

$$V_o = \left[\frac{a_0}{R} + \frac{a_1}{2R} + \cdots + \frac{a_{n-1}}{2^{n-1}R}\right] R_f V_r \tag{2.8}$$

where $a_0,\ a_1,\ \cdots,\ a_{n-1}$ are either 1 or 0 depending on the digital binary word which is to be converted.

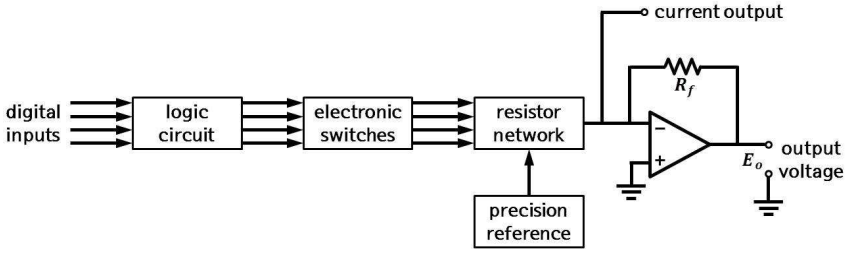

Figure 2.14 Basic elements of a D/A

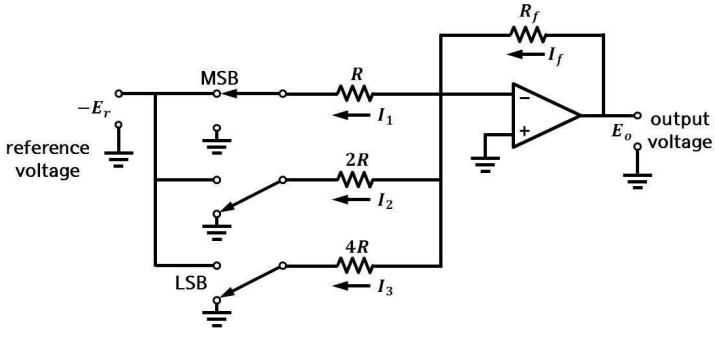

Figure 2.15 Basic elements of a D/A

Table 2.1 Output voltage of a 3-bit D/A

digital word $(a_0\ a_1\ a_2)$	output voltage	fraction of FS	FS $= V_r = 10[V]$ $R = R_f$
$(0,\ 0,\ 1)$	$\frac{1}{4}\frac{R_f}{R}V_r$	$\frac{1}{8}$FS = 1×LSB	1.25
$(0,\ 1,\ 0)$	$\frac{2}{4}\frac{R_f}{R}V_r$	$\frac{2}{8}$FS = 2×LSB	2.50
$(0,\ 1,\ 1)$	$\frac{3}{4}\frac{R_f}{R}V_r$	$\frac{3}{8}$FS = 3×LSB	3.75
$(1,\ 0,\ 0)$	$\frac{4}{4}\frac{R_f}{R}V_r$	$\frac{4}{8}$FS = 4×LSB	5.00
$(1,\ 0,\ 1)$	$\frac{5}{4}\frac{R_f}{R}V_r$	$\frac{5}{8}$FS = 5×LSB	6.25
$(1,\ 1,\ 0)$	$\frac{6}{4}\frac{R_f}{R}V_r$	$\frac{6}{8}$FS = 6×LSB	7.50
$(1,\ 1,\ 1)$	$\frac{7}{4}\frac{R_f}{R}V_r$	$\frac{7}{8}$FS = 7×LSB	8.75

2.6 Analog-to-Digital Conversion

The *analog-to-digital* conversion, or simply encoding, consists of converting the numerical information contained in an analog signal into a digitally coded word. A/D conversion is a more complex process than D/A conversion and requires more elaborate circuitry. In comparison to the D/A conversion, the A/D converter is generally more expensive and has slower response for the same conversion accuracy.

When a number is given as an input to an A/D,the converter performs the operations of *quantizing* and *encoding*. When a time-varying signal (voltage or current) is to be converted from analog to digital form, the A/D converter usually performs the following operations in succession: sample-and hold, quantization, and encoding.

The sampling operation is needed to sample the analog signal at xed periodic intervals. Theoretically, the holding operation is not needed; however, the A/Dconversion, the sampled signal is held until the conversion is completed. Figure 2.16 gives the block diagram representation of an A/D converter.

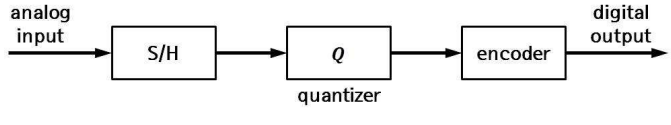

Figure 2.16 Block diagram representation of an A/D

CHAPTER 3

DISCRETE-TIME SYSTEM ANALYSIS USING $\mathcal{Z}$-TRANSFORM

3.1 $\mathcal{Z}$-Transform

The discrete-time counterpart to the Laplace transform is the $\mathcal{Z}$-transform, hence we can easily design and analyze the discrete-time control system using $\mathcal{Z}$-transform. General approach to derive the $\mathcal{Z}$-transform is to use the discrete-time Fourier transform (DTFT) and the sampling operation applied to a continuous-time signal but this approach is rather lengthy. This is because, in this section, we adopt a much shorter approach which introduces the $\mathcal{Z}$-transform as the Laplace transform of a continuous-time signal sampled by an ideal sampler and a *zero-order hold* (ZOH). For such case, the *sampling* of a continuous-time signal $x(t)$ in the discrete-time system leads to another continuous-time signal that has a staircase form, as presented in Figure 3.1. The signal $x_h(t)$ indicated by the dashed line in the figure can be considered as a *discrete-time signal* with its values defined at $x_h(t) = x(kT) \triangleq x[k]$.

As mentioned in the previous chapter, the term ZOH comes from the fact that this element can make a continuous-time staircase signal from a discrete-time signal by holding the value of a discrete-time signal, $x(kT)$, during the given sampling interval T. In practice, the ZOH element is often used for the conversion of a discrete-time signal into a continuous-time signal in many real situations and, without loss of

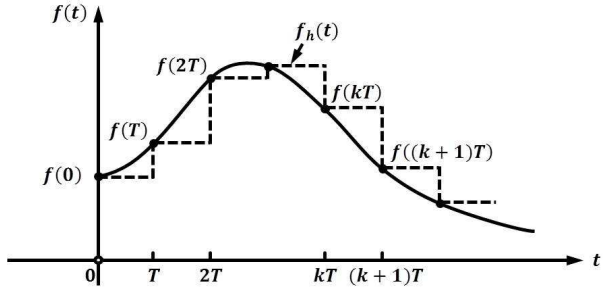

Figure 3.1 Sampled signal after discretization

generality, we use it for the purpose of deriving the $\mathcal{Z}$-transform from the Laplace transform. In the following, we will recall the transfer function of the ZOH element.

Lettting T denote the sampling period, the discrete-time signal $x_h(t)$ obtained from the original continuous-time signal $x(t)$ is given by

$$x_h(t) = \sum_{k=0}^{\infty} x(kT)[u(t-kT) - u(t-(k+1)T)]. \tag{3.1}$$

Applying the Laplace transform for $x_h(t)$, we have

$$X_h(s) = \mathcal{L}\{x_h(t)\} = \sum_{k=0}^{\infty} x(kT)\frac{e^{-skT} - e^{-s(k+1)T}}{s} = G_{h0}(s)X^*(s) \tag{3.2}$$

where $G_{h0}(s)$ is the transfer function of the ZOH defined in (2.5) and $X^*(s)$ means the Laplace transform of the sampled signal $x(kT)$,

$$\begin{aligned} X^*(s) &\triangleq \mathcal{L}\left\{\sum_{k=0}^{\infty} x(kT)\delta(t-kT)\right\} = \sum_{k=0}^{\infty} x(kT)\mathcal{L}\{\delta(t-kT)\} \\ &= \sum_{k=0}^{\infty} x(kT)e^{-skT} \end{aligned} \tag{3.3}$$

Therefore, from the above relation, we can define the $\mathcal{Z}$-transform as

$$X(z) \triangleq \mathcal{Z}\{x[k]\} = X^*(s)|_{s=\frac{1}{T}\ln(z)} = \sum_{k=0}^{\infty} x[k]z^{-k}, \;\; z \triangleq e^{sT}. \tag{3.4}$$

As a matter of fact, the above definition is obtained for causal signals, and it defines the *one-sided* or *unilateral $\mathcal{Z}$-transform*. We could have done the same derivations using a non-causal signal in which case we would have obtained the *two-sided* or *bilateral $\mathcal{Z}$-transform* defined by

$$X(z) \triangleq \sum_{k=-\infty}^{\infty} x[k]z^{-k}.X \tag{3.5}$$

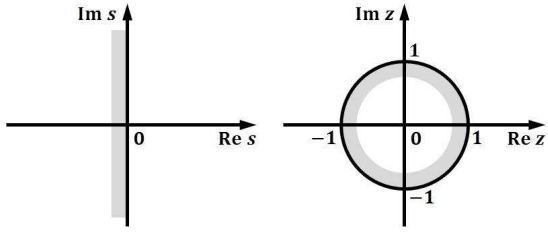

Figure 3.2 Mapping between the s-plane and z-plane

Let us discuss the implication of the operator $z = e^{sT}$ in detail. It is obvious that it maps the $j\omega$-axis on the s-plane into the unit circle on the z-plane, and it maps the open left half s-plane into the interior of the unit circle on the z-plane as shown in Figure 3.2. Indeed, there exists the periodicity as follows:

$$z = e^{sT} = e^{(\sigma + j\omega)T} \tag{3.6}$$

where $s = \sigma + j\omega$, if $\sigma = 0$ (the imaginary axis of the s-plane), then $|z| = |e^{j\omega T}| = 1$ (the unit circle on the z-plane). This means that the mapping is not one-to-one. In fact, for $\omega_s = \frac{2\pi}{T}$, all $s = 0,\ \pm j\omega_s,\ \pm 2\omega_s,\ \cdots$ are mapped into $z = 1$. If $\sigma < 0$ (left half s-plane), then $|z| = |e^{\sigma T}||e^{j\omega T}| = |e^{\sigma T}| < 1$ (interior of the unit circle on the z-plane). This establishes the mapping shown in Figure 3.2.

3.2 Region of Convergence

There are also a number of important relationships between the $\mathcal{Z}$-transform and the Fourier transform. To explore these relationships, let us express the complex variable z in polar form as follows:

$$z = \rho e^{j\phi}, \tag{3.7}$$

with ρ as the magnitude of z and ω as the phase of z.
In terms of ρ and ω, (3.3) becomes

$$X(\rho e^{j\omega}) = \sum_{n=0}^{\infty} x[n](\rho e^{j\omega})^{-n} \tag{3.8}$$

or, equivalently,

$$X(\rho e^{j\omega}) = \sum_{n=0}^{\infty} \left(x[n]\rho^{-n}\right) e^{-j\omega n} \tag{3.9}$$

From (3.9), it is clear that $X(\rho e^{j\omega})$ is the Fourier transform of the sequence $x[n]$ multiplied by a real exponential ρ^{-n}, that is,

$$X(\rho e^{j\omega}) = \mathcal{Z}\left\{x[n] = x_h(t)\right\} = \mathcal{L}\left\{x_h(t)\right\}|_{s=\frac{1}{T}\ln(z)} = \mathcal{F}\left\{x[n]\rho^{-n}\right\} \tag{3.10}$$

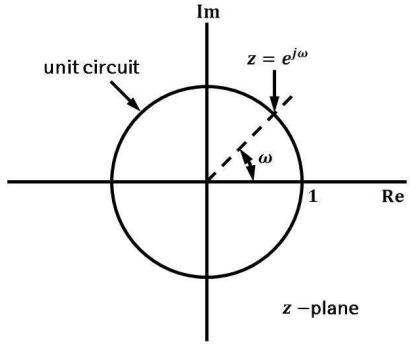

Figure 3.3 Complex z-plane

The exponential weighting ρ^{-n} may be decaying or growing with increasing n, depending on whether ρ is greater than or less than unity. We note in particular that for $\rho = 1$, or equivalently $|z| = 1$, the $\mathcal{Z}$-transform reduces to the Fourier transform, that is,

$$X(z)|_{z=e^{j\omega}} = \mathcal{F}\{x[n]\} \tag{3.11}$$

At this point, it is worth note that the z-transform is closely related to the Fourier transform for discrete-time signals, but with some important differences. In the continuous-time case, the Laplace transform reduces to the Fourier transform when the real part of the transform variable s is zero. Interpreted in terms of the s-plane, this means that the Laplace transform reduces to the Fourier transform on the imaginary axis (i.e., for $s = j\omega$). On the other hand, the z-transform reduces to the Fourier transform when the magnitude of the transform variable z is *unity* (i.e., for $z = e^{j\omega}$). Thus, the $\mathcal{Z}$-transform reduces to the Fourier transform on the contour in the complex z-plane corresponding to a circle with a radius of unity as indicated in Figure 3.3. This circle in the z-plane is referred to as the *unit circle*, and plays a role in the discussion of the z-transform similar to the role of the imaginary axis in the s-plane for the Laplace transform.

Because of this relationship between the z-transform and the Fourier transform, it is convenient at this point to make a simple change of notation in representing the discrete-time Fourier transform. Specifically, we will now denote the independent variable associated with the discrete-time Fourier transform as $e^{j\omega}$ rather than simply ω, to emphasize the fact that it is equal to the z-transform for $z = e^{j\omega}$. With this change in notation, we can also express (3.11) as

$$X(z)|_{z=e^{j\omega}} = \mathcal{F}\{x[n]\} = X(e^{j\omega}) \tag{3.12}$$

From (3.10), for convergence of the z-transform, we require that the Fourier transform of $x[n]\rho^{-n}$ should converge. For any specific sequence $x[n]$, we would expect this convergence for some values of ρ and not for others. In general, there is a range of values of z for which $X(z)$ converges. As with the Laplace transform, this range of values is referred to as the *region of convergence* (ROC). If the ROC includes the unit circle, then the Fourier transform also converges.

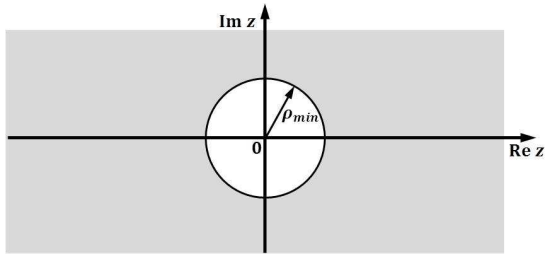

Figure 3.4 The ROC of the one-sided $\mathcal{Z}$-tansform

In summary, it is obvious that the $\mathcal{Z}$-transform is defined for discrete-time signals when the infinite series (3.4) is convergent. This will be the case if

$$\lim_{n\to\infty}\left(\sum_{k=0}^{n}\left|x[k]z^{-k}\right| \leq \sum_{k=0}^{n}|x[k]|\,|z|^{-k} = \sum_{k=0}^{n}|x[k]|\,\rho^{-k}\right) = C \leq \infty, \quad (3.13)$$

where C is a positive real constant, which may depend on ρ.
This condition means that $f[k]$ has a one-sided $\mathcal{Z}$-transform if

$$|z| > \rho_{min} \quad (3.14)$$

where ρ_{min} denotes the minimal element in the set of real positive numbers such that the convergence condition is satisfied. The set of complex numbers z satisfying inequality (3.14) defines the region of absolute convergence of the $\mathcal{Z}$-transform $F(z)$, which means that $F(z)$ is not defined outside of that region (see Figure 3.4). For such case, the ROC is often denoted by $\Re = \{z : |z| > \rho_{min}\}$.

To illustrate the z-transform and the associated ROC, let us consider the following examples.

EXAMPLE 3.1

Consider the signal $x[n] = a^n u[n]$. From (3.4), we have

$$X(z) = \sum_{n=-\infty}^{\infty} a^n u[n] z^{-n} = \sum_{n=0}^{\infty}\left(az^{-1}\right)^n,$$

hence, for convergence of $X(z)$, we require that $\sum_{n=0}^{\infty}\left|az^{-1}\right|^n < \infty$.
Thus, the ROC is the range of values of z for which $\left|az^{-1}\right| < 1$ or equivalently $|z| > |a|$. Then,

$$X(z) = \sum_{n=0}^{\infty}\left(az^{-1}\right)^n = \frac{1}{1-az^{-1}} = \frac{z}{z-a},\ |z| > |a|.$$

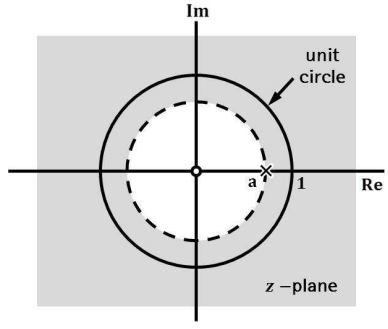

Figure 3.5 Pole-zero plot and ROC for Example 3.1

Consequently, the $\mathcal{Z}$-transform converges for any finite value of a. The Fourier transform of $x[n]$, on the other hand, only converges when $|a| < 1$. For $a = 1$, $x[n]$ becomes the unit step sequence whose $\mathcal{Z}$-transform is

$$X(z) = \frac{1}{1 - z^{-1}}, \quad |z| > 1.$$

We see that the $\mathcal{Z}$-transform in Example 3.1 is a rational function of z. Therefore, just as with rational Laplace transforms, it can be characterized by its zeros (the roots of the numerator polynomial) and its poles (the roots of the denominator polynomial). In Example 3.1, there is one zero, at $z = 0$, and one pole, at $z = a$. The pole-zero plot and its ROC are shown in Figure 3.5. For $|a| > 1$, the ROC does not include the unit circle, consistent with the fact that for these values of z, the Fourier transform of $x[n] = a^n u[n]$ does not converge.

EXAMPLE 3.2

Now let $z[n] = -a^n u[-n-1]$. Then,

$$X(z) = -\sum_{n=-\infty}^{\infty} a^n u[-n-1] z^{-n} = -\sum_{n=-\infty}^{-1} a^n z^{-n} = -\sum_{n=1}^{\infty} a^{-n} z^n = 1 - \sum_{n=0}^{\infty} (a^{-1} z)^n.$$

If $|a^{-1}z| < 1$ or equivalently $|z| < |a|$, the sum in the above equation converges and

$$X(z) = 1 - \frac{1}{1 - a^{-1}z} = \frac{1}{1 - az^{-1}} = \frac{z}{z - a}.$$

The pole-zero plot and region of convergence for this example are shown in Figure 3.6.

Comparing the results in Example 3.1 and 3.2, and Figures 3.5 and 3.6, we see that the algebraic expression for $X(z)$ and the corresponding pole-zero plot are identical in Examples 3.2. Also, in both examples, the sequences were exponentials and the

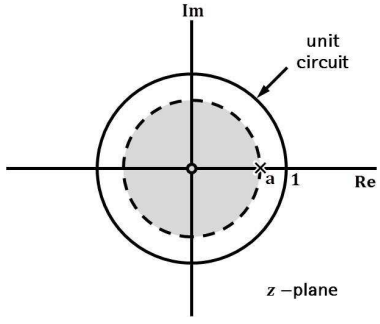

Figure 3.6 Pole-zero plot and ROC for Example 3.2

resulting $\mathcal{Z}$-transforms were rational. In fact, as further suggested by the next example, $X(z)$ will be rational whenever $x[n]$ is a linear combination of real or complex exponentials.

EXAMPLE 3.3

Let us consider a signal that is the sum of two real exponentials:

$$x[n] = \left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{3}\right)^n u[n].$$

The $\mathcal{Z}$-transform is then

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{\infty} \left[\left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{3}\right)^n u[n]\right] z^{-n} \\ &= \sum_{n=0}^{\infty} \left(\frac{1}{2}z^{-1}\right)^n + \sum_{n=0}^{\infty} \left(\frac{1}{3}z^{-1}\right)^n \\ &= \frac{1}{1-\frac{1}{2}z^{-1}} + \frac{1}{1-\frac{1}{3}z^{-1}} \\ &= \frac{z\left(2z-\frac{5}{6}\right)}{\left(z-\frac{1}{2}\right)\left(z-\frac{1}{3}\right)}. \end{aligned}$$

For convergence of $X(z)$, both sums in the above equation must converge, which requires that both $\left|\frac{1}{2}z^{-1}\right| < 1$ and $\left|\frac{1}{3}z^{-1}\right| < 1$ or equivalently $|z| > \frac{1}{2}$ and $|z| > \frac{1}{3}$. Thus, the resultant ROC is $|z| > \frac{1}{2}$.

The $\mathcal{Z}$-transform for this example can also be obtained using the results of Example 3.1. Specifically, from the definition of the $\mathcal{Z}$-transform (3.4), we see that the $\mathcal{Z}$-transform is linear. That is, if $z[n]$ is the sum of two terms, then $X(z)$ will be the sum of the $\mathcal{Z}$-transform of the individual terms and will converge

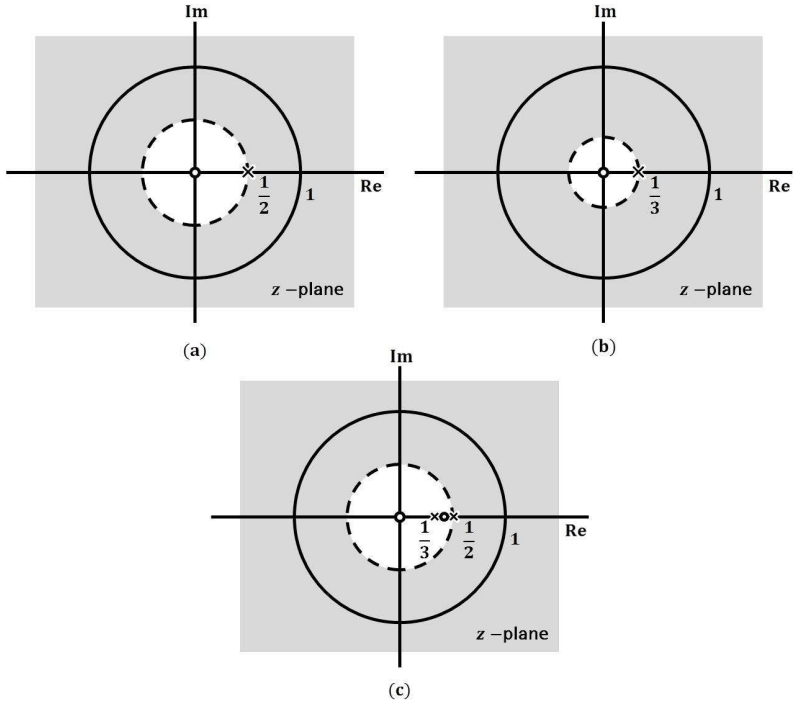

Figure 3.7 Pole-zero plot and ROC for Example 3.3

when both z-transforms converge. From Example 3.1, we have

$$\mathcal{Z}\left\{\left(\frac{1}{2}\right)^n u[n]\right\} = \frac{1}{1-\frac{1}{2}z^{-1}}, \quad |z| > \frac{1}{2},$$
$$\mathcal{Z}\left\{\left(\frac{1}{3}\right)^n u[n]\right\} = \frac{1}{1-\frac{1}{3}z^{-1}}, \quad |z| > \frac{1}{3}.$$

Hence,

$$\mathcal{Z}\left\{\left(\frac{1}{2}\right)^n u[n] + \left(\frac{1}{3}\right)^n u[n]\right\} = \frac{1}{1-\frac{1}{2}z^{-1}} + \frac{1}{1-\frac{1}{3}z^{-1}}, \quad |z| > \frac{1}{2},$$

as we had determined above. The pole-zero plot and ROC for the $\mathcal{Z}$-transform of each of the individual terms and for the combined signal are shown in Figure 3.7.

In each of the three examples above, we expressed the $\mathcal{Z}$-transform both as a ratio of polynomials in z and a ratio of polynomials in z^{-1}. From the form of the definition of the $\mathcal{Z}$-transform as given in (3.4), we see that for sequences that are zero for $n < 0$, $X(z)$ involves only negative powers of z. Thus, for the causal signals, it is particularly convenient for $X(z)$ to be expressed in terms of polynomials in z^{-1}

Table 3.1 $\mathcal{Z}$-transform pairs

DT signal	$\mathcal{Z}$-transform
$\delta[k]$	1
$u[k]$	$\dfrac{1}{1-z^{-1}}$
$ku[k]$	$\dfrac{z^{-1}}{(1-z^{-1})^2}$
$k^2u[k]$	$\dfrac{z^{-1}(1+z^{-1})}{(1-z^{-1})^3}$
$a^ku[k]$	$\dfrac{1}{1-az^{-1}}$
$\cos(\omega_0 kT)u[k]$	$\dfrac{1-z^{-1}\cos(\omega_0 T)}{1-2z^{-1}\cos(\omega_0 T)+z^{-2}}$
$\sin(\omega_0 kT)u[k]$	$\dfrac{z^{-1}\sin(\omega_0 T)}{1-2z^{-1}\cos(\omega_0 T)+z^{-2}}$
$a^k\cos(\omega_0 kT)u[k]$	$\dfrac{1-az^{-1}\cos(\omega_0 T)}{1-2az^{-1}\cos(\omega_0 T)+a^2z^{-2}}$
$a^k\sin(\omega_0 kT)u[k]$	$\dfrac{az^{-1}\sin(\omega_0 T)}{1-2az^{-1}\cos(\omega_0 T)+a^2z^{-2}}$

rather than z, and when appropriate we will use that form in our further discussions. When the $\mathcal{Z}$-transform is expressed in terms of factors of the form $(1-z^{-1})$, it should be remembered that such a factor introduces both a pole and a zero, as evidenced in the algebraic expressions for the foregoing examples.

3.3 Properties of $\mathcal{Z}$-Transform

Since the $\mathcal{Z}$-transform is, in general, an infinite sum of complex numbers, most of the properties of the $\mathcal{Z}$-transform will be proved by using the known (and very simple) properties of infinite sums.

Property 1: Linearity Let $\mathcal{Z}\{f_i[k]\} = F_i(z)$ with ROC $\Re_i$, $i = 1, 2, \cdots, n$. Then, for arbitrary constants $\alpha_1,\ \alpha_2, \cdots, \alpha_n$,

$$\mathcal{Z}\{\alpha_1 f_1[k] + \cdots + \alpha_n f_n[k]\} = \alpha_1 F_1(z) + \cdots + \alpha_n F_n(z). \tag{3.15}$$

The ROC of this linear combination of discrete-time signals is the intersection of the ROC of the individual signals, that is, it is equal to $\Re_1 \cap \cdots \cap \Re_n$.

Property 2: Right-shift in Time (Discrete Time Integration) If a discrete-time signal $f[k]u[k]$ is shifted by $m > 0$, the resulting signal $f[k-m]u[k-m]$ has the

following $\mathcal{Z}$-transform.

$$\mathcal{Z}\{f[k-m]u[k-m]\} = z^{-m}F(z), \tag{3.16}$$

where it has been assumed that $\mathcal{Z}\{f[k]u[k]\} = F(z)$.
Note that the ROC of the time shifted signal is the same as the ROC of the original signal.

Another variant of this property is written as

$$\mathcal{Z}\{f[k-m]u[k]\} = z^{-m}F(z) + \left(\sum_{i=1}^{m} f[-i]z^{i-m}\right). \tag{3.17}$$

proof. By the definition of the $\mathcal{Z}$-transform (3.4), we have

$$\mathcal{Z}\{f[k-m]u[k]\} = \sum_{k=0}^{\infty} f[k-m]u[k]z^{-k}. \tag{3.18}$$

Introducing a change of variables $k-m=i$ results in

$$\begin{aligned}\sum_{k=0}^{\infty} f[k-m]u[k]z^{-k} &= \sum_{i=-m}^{\infty} f[i]u[i+m]z^{-(i+m)} \\ &= z^{-m}\sum_{i=-m}^{\infty} f[i]z^{-i} \\ &= z^{-m}\left(\sum_{i=0}^{\infty} f[i]z^{-i} + \sum_{i=-m}^{-1} f[i]z^{-i}\right) \\ &= z^{-m}F(k) + z^{-m}\sum_{i=-m}^{-1} f[i]z^{-i}.\end{aligned} \tag{3.19}$$

In the last sum, we can replace the dummy variable of summation i by $-m$ and then rename m as i, which leads to the desired result. (Q.E.D.)

Note that if $f[k]$ is a causal signal($f[k]=0,\ k<0$), then all terms corresponding to the signal samples for negative discrete-time instants drop out and we are left with

$$f[k-m]u[k] \leftrightarrow z^{-m}F(z),\ \ f[k]=0, \forall k<0. \tag{3.20}$$

This formula will play a very important role in the analysis of linear time-invariant discrete-time system expressed as a difference equation.

Property 3: Left-shift in Time (Discrete Time Differentiation) Left shifts in time, which in fact stand for the discrete-time derivatives, satisfies the following:

$$\mathcal{Z}\{f[k+m]u[k]\} = z^{m}F(z) - \sum_{i=0}^{m-1} f[i]z^{m-i} \tag{3.21}$$

proof. We can establish the proof of (3.21) rather easily. For the first discrete-time derivative, we have

$$\mathcal{Z}\{f[k+1]u[k]\} = \sum_{k=0}^{\infty} f[k+1]u[k]z^{-k}. \tag{3.22}$$

Introducing a change of variables $i = k+1$, we obtain

$$\begin{aligned}
\mathcal{Z}\{f[k+1]u[k]\} &= \sum_{i=1}^{\infty} f[i]u[i-1]z^{-(i-1)} = z\sum_{i=1}^{\infty} f[i]z^{-i} \\
&= z\left(\sum_{i=1}^{\infty} f[i]z^{-i} + f[0] - f[0]\right) \\
&= z\left(\sum_{i=0}^{\infty} f[i]z^{-i} - f[0]\right) \\
&= zF(z) - zf[0].
\end{aligned} \tag{3.23}$$

With the similar way, for the second discrete-time derivative, we have

$$\mathcal{Z}\{f[k+2]u[k]\} = \sum_{k=0}^{\infty} f[k+2]u[k]z^{-k}, \tag{3.24}$$

which, after a change of variables $i = k+2$, becomes

$$\begin{aligned}
\mathcal{Z}\{f[k+2]u[k]\} &= \sum_{i=2}^{\infty} f[i]u[i-2]z^{-(i-2)} = z^2\sum_{i=2}^{\infty} f[i]z^{-i} \\
&= z^2\left(\sum_{i=2}^{\infty} f[i]z^{-i} + z^{-1}f[1] + f[0] - z^{-1}f[1] - f[0]\right) \\
&= z^2\left(\sum_{i=0}^{\infty} f[i]z^{-i} - z^{-1}f[1] - f[0]\right) \\
&= z^2F(z) - zf[1] - z^2f[0].
\end{aligned} \tag{3.25}$$

In the sequel, it is easy to complete the general proof for the n^{th} derivative. (Q.E.D.)

It can be concluded that the derivative property of the $\mathcal{Z}$-transform is analogous to the corresponding property of the Laplace transform. Namely, the n^{th} derivative in discrete-time, which is represented by the left shift in time for n discrete-time instants, in the frequency domain corresponds to a multiplication by z^n; that is, *assuming that all initial conditions are zero*, we have

$$\mathcal{Z}\{f[k+n]u[k]\} = z^nF(z) \tag{3.26}$$

On the other hand, for the continuous-time derivatives, the following holds.

$$\mathcal{L}\left\{\frac{d^nf(t)}{dt^n}\right\} = s^nF(s), \quad f^{(i)}(0^{-1}) = 0, \quad (i = 0, 1, \cdots, n-1) \tag{3.27}$$

Property 4: Time Multiplication The time multiplication property states that

$$\begin{aligned} \mathcal{Z}\{kf[k]\} &= -z\frac{d}{dz}F(z), \\ \mathcal{Z}\{k^2 f[k]\} &= z\frac{d}{dz}F(z) + z^2\frac{d^2}{dz^2}F(z). \end{aligned} \tag{3.28}$$

The ROCs for the newly formed signals, $kf[k]$ and $k^2 f[k]$ are equal to the ROC of the original signal $f[k]$.

Property 5: Frequency Scaling The frequency scaling property of the $\mathcal{Z}$-transform is a consequence of the obvious fact that

$$\mathcal{Z}\{a^k f[k]\} = \sum_{k=0}^{\infty} a^k f[k] z^{-k} = \sum_{k=0}^{\infty} f[k]\left(\frac{z}{a}\right)^{-k} = F\left(\frac{z}{a}\right), \tag{3.29}$$

with the new ROC of equal to $|a|\Re$, where $\Re$ means the ROC of the original signal $f[k]$. This property is an intermediate step in establishing the modulation property. In addition, it can be used for finding the $\mathcal{Z}$-transform of signals multiplied by an exponential function.

Property 6: Modulation The modulation property is very useful for digital signal processing and/or communication systems. It is directly derived from the frequency scaling property. Representing the sine and cosine functions by Euler's formulas yields

$$\cos(\omega kT) = \frac{1}{2}\left(e^{j\omega kT} + e^{-j\omega kT}\right), \quad \sin(\omega kT) = \frac{1}{j2}\left(e^{j\omega kT} - e^{-j\omega kT}\right). \tag{3.30}$$

Using the frequency scaling property, we have

$$\begin{aligned} \mathcal{Z}\{f[k]\cos(\omega kT)\} &= \frac{1}{2}\left(F(e^{j\omega T}z) + F(e^{-j\omega T}z)\right), \\ \mathcal{Z}\{f[k]\sin(\omega kT)\} &= \frac{j}{2}\left(F(e^{j\omega T}z) - F(e^{-j\omega T}z)\right). \end{aligned} \tag{3.31}$$

The above relation constitutes the modulation property. We can also use this property to find $\mathcal{Z}$-transform of the cosine and sine signals.

Property 7: Convolution This is the most important property of the one-sided $\mathcal{Z}$-transform from the linear system theory point of view. It states that

$$\mathcal{Z}\{f_1[k] * f_2[k]\} = F_1(z)F_2(z). \tag{3.32}$$

The ROC of the convolved signal is equal to $\Re_1 \cap \Re_2$, where $\Re_i$ means the ROC of the signal $f_i[k]$. Since the one-sided $\mathcal{Z}$-transform is defined only for nonnegative values of k, we will use the following definition for the discrete-time convolution:

$$f_1[k] * f_2[k] = \sum_{m=0}^{\infty} f_1[m] f_2[k-m] \tag{3.33}$$

This definition will play a very important role in the analysis of discrete-time linear time-invariant systems.

Property 8: Initial Value Theorem From the definition of the $\mathcal{Z}$-transform, the expression for the signal initial value follows directly. This is because

$$F(z) = \mathcal{Z}\{f[k]\} = f[0] + z^{-1}f[1] + z^{-2}f[2] + \cdots \tag{3.34}$$

implies that

$$f[0] = \lim_{z\to\infty} F(z). \tag{3.35}$$

Using simple algebra, we can also recover the other signal values. For example, $f[1]$ is obtained using the following equation.

$$f[1] = \lim_{z\to\infty} \{z\,(F(z) - f[0])\} \tag{3.36}$$

It is left as an exercise for the readers to derive the expression for $f[n-1]$.

Property 9: Final Value Theorem Assuming that we know the $\mathcal{Z}$-transform of a signal, we are able to find the signal's steady-state value $f[\infty]$ without going back to the time-domain via the inverse $\mathcal{Z}$-transform and then taking the limitation $k \to \infty$. This can be achieved as follows:

$$f[\infty] = \lim_{k\to\infty} f[k] = \lim_{z\to 1} \left\{(1 - z^{-1})F(z)\right\} \tag{3.37}$$

Note that the final value theorem is applicable only to signals for which the limit at infinity exists, that is, asymptotically stable signals. For example, it cannot be applied to the sine and cosine signals because they have no limits at $k = \infty$.

We can determine whether the final value theorem is applicable to the given signal $f[k]$ by investigating that the complex variable function $(z-1)F(z)$ has no poles outside or on the unit circle (recall that the poles are the values at which $(p_j - 1)F(p_j) = \infty$). This condition will become clearer after we learn about the stability of discrete-time linear systems in Section 3.7.

EXAMPLE 3.4

Assume that the $\mathcal{Z}$-transform of a certain signal is given by

$$F(z) = \frac{z - 0.5}{z(z + 0.5)(z - 1)}.$$

The initial and final value theorems produce

$$f[0] = \lim_{z\to\infty} \{F(z)\} = 0,$$

$$f[\infty] = \lim_{z\to 1} \left\{(1 - z^{-1})F(z)\right\} = \frac{1}{3}.$$

Note that the final value theorem is applicable in this case because $(z-1)F(z)$ has two poles inside the unit circle (at 0 and -0.5). This means that the above answer is correct.

Table 3.2 Properties of the $\mathcal{Z}$-transform

DT signal	$\mathcal{Z}$-transform
$a_1 f_1[k] + a_2 f_2[k]$	$a_1 F_1(z) + a_2 F_2(z)$
$f[k-m]u[k-m]$	$z^{-m}F(z)$
$f[k-m]u[k]$	$z^{-m}F(z) + z^{-m+1}f[-1] + \cdots + z^{-1}f[-m+1] + f[-m]$
$f[k+m]u[k]$	$z^{m}F(z) - z^{m}f[0] - z^{m-1}f[1] - \cdots - zf[m-1]$
$kf[k]$	$-z\dfrac{d}{dz}F(z)$
$k^2 f[k]$	$z\dfrac{d}{dz}F(z) + z^2\dfrac{d^2}{dz^2}F(z)$
$a^k f[k]$	$F\left(\dfrac{z}{a}\right)$
$f[k]\cos(\omega_0 kT)$	$\dfrac{1}{2}\left(F(ze^{j\omega_0 T}) + F(ze^{-j\omega_0 T})\right)$
$f[k]\sin(\omega_0 kT)$	$\dfrac{j}{2}\left(F(ze^{j\omega_0 T}) - F(ze^{-j\omega_0 T})\right)$
$f_1[k] * f_2[k]$	$F_1(z)F_2(z)$
$\lim\limits_{k\to 0} f[k]$	$\lim\limits_{z\to\infty} F(z)$
$\lim\limits_{k\to\infty} f[k]$	$\lim\limits_{z\to 1}\left\{(1-z^{-1})F(z)\right\}$

For your convenience, the properties of $\mathcal{Z}$-transform are summarized in Table 3.2.

3.4 Inverse of the $\mathcal{Z}$-Transform

It is not straightforward to derive the inverse $\mathcal{Z}$-transform. Since it can be derived using complex variable contour integration, it might be helpful to study the basic concepts used in complex analysis. The definition formula for the inverse $\mathcal{Z}$-transform is given by the following complex variable contour integral.

$$f[k] \triangleq \frac{1}{2\pi j}\oint_{\Gamma} F(z)z^{k-1}dz, \tag{3.38}$$

where Γ is a circle of radius greater than ρ_{min} defined in (3.14) that encircles all singularities in the ROC. The most common singularities are the pole p_i of $F(z)z^{k-1}$, at which $F(z)z^{k-1} = \infty$. This integral can be evaluated using Cauchy's residue theorem. The result of this theorem is that a contour integral of a function of z which

is analytic inside the contour Γ, except at a finite number of isolated singularities p_i.

$$f[k] = \frac{1}{j2\pi} \oint_{\Gamma} F(z) z^{k-1} dz = \sum_i \mathrm{Re}s\{F(p_i) p_i^{k-1}\} \tag{3.39}$$

As possible ways to find the inverse $\mathcal{Z}$-transform (3.38), for rational functions (represented by ratios of two polynomials of z), two methods known as the long division expansion and partial fraction expansion are often employed.

Long Division Expansion The underlying idea of this method is to calculate $f[k]$ by expanding the function $F(z)$ into a series of powers of z^{-1}.

$$F(z) = f[0] + z^{-1} f[1] + z^{-2} f[2] + \cdots + z^{-k} f[k] + \cdots \tag{3.40}$$

The coefficients of this series are the values of $f[k]$ at $k = 0, 1, 2, \cdots$. Although this method is very simple, it might not be computationally efficient. Furthermore, for signals that have infinite time duration, the method is not capable of producing all of the signal values.

Partial Fraction Expansion Since both $\mathcal{Z}$-transforms and Laplace transforms are most often given in terms of ratios of two polynomials with complex variables, the problem of finding the inverse $\mathcal{Z}$-transform is exactly the same as the problem of finding the inverse Laplace transform. The main technique for finding the inverse Laplace transform, based on partial fraction expansion, is also valid in the case of the $\mathcal{Z}$-transform as well.

Examining the common $\mathcal{Z}$-transform pairs in Table 3.1, we notice that all of the $\mathcal{Z}$- transforms presented, except for $\delta[k]$, contain in the numerators a product of the variable z and some other quantity. This indicates that it will be wise to perform the partial fraction expansion of the function

$$\frac{1}{z} F(z) = \frac{1}{z} \frac{N(z)}{D(z)}. \tag{3.41}$$

Thus, given the $\mathcal{Z}$-transform $F(z)$, we first form

$$F_1(z) = \frac{F(z)}{z}. \tag{3.42}$$

The function $F_1(z)$ is then expanded in partial fractions. For example, for the case of distinct poles, we have

$$F_1(z) = \sum_{i=1}^{n} c_i \frac{1}{z - p_i}, \quad F(z) = \sum_{k=1}^{n} c_i \frac{z}{z - p_i}. \tag{3.43}$$

In this case, $F(z)$ has its inverse given by a discrete-time exponential signal (see Table 3.1), that is,

$$f[k] = \left(\sum_{i=1}^{n} c_i p_i^k \right) u[k] \tag{3.44}$$

EXAMPLE 3.5

Consider the case with multiple poles given by

$$F(z) = \frac{z(z+1)}{(z-0.5)^2(-1)}.$$

The required partial fraction expansion has the form

$$\begin{aligned} F_1(z) = \frac{F(z)}{z} &= \frac{z+1}{(z-0.5)^2(z-1)} \\ &= \frac{c_{11}}{(z-0.5)} + \frac{c_{12}}{(z-0.5)^2} + \frac{c_2}{(z-1)} \end{aligned}$$

where the coefficients c_{11} and c_{12} correspond to the multiple poles. As in the Laplace transform, these coefficients can be found by the following formula.

$$c_{1j} = \frac{1}{(r-j)!} \lim_{z \to p_1} \left\{ \frac{d^{r-j}}{dz^{r-j}} (z-p_1)^r \frac{F(z)}{z} \right\},$$

where r means the multiplicity of the pole p_1 in $F(z)$. In our problem, we have $r = 2$, hence

$$\begin{aligned} c_{12} &= \left. \frac{z+1}{z-1} \right|_{z=0.5} = -3, \\ c_{11} &= \left. \frac{d}{dz} \left(\frac{z+1}{z-1} \right) \right|_{z=0.5} = \left. \frac{-2}{(z-1)^2} \right|_{z=0.5} = -8. \end{aligned}$$

Meanwhile, the coefficient corresponding to the simple pole is

$$c_2 = \left. \frac{z+1}{(z-0.5)^2} \right|_{z=1} = 8.$$

The original function $F(z)$ has the expansion

$$F(z) = \frac{-8z}{(z-0.5)} + \frac{-3z}{(z-0.5)^2} + \frac{8z}{(z-1)}.$$

As a result, using Table 3.1 and Table 3.2, we obtain

$$f[k] = \left(-8(0.5)^k - 6k(0.5)^k + 8 \right) u[k].$$

EXAMPLE 3.6

Consider the infinite series

$$S_1 = \sum_{k=0}^{\infty} k a^k, \quad S_2 = \sum_{k=0}^{\infty} k^2 a^k.$$

The $\mathcal{Z}$-transform of the signals $ka^k u[k]$ and $k^2 a^k u[k]$ are respectively given by

$$F_1(z) = \frac{az}{(z-a)^2}, \quad F_2(z) = \frac{az(z+a)}{(z-a)^3},$$

with the ROC equal, in both cases, to $|z| > |a|$.
It can be observed that, for $|a| < 1$, the point $z = 1$ belongs to the corresponding ROC, hence

$$S_1 = \sum_{k=0}^{\infty} ka^k = F_1(1) = \frac{a}{(1-a)^2}, \quad |a| < 1,$$
$$S_2 = \sum_{k=0}^{\infty} k^2 a^k = F_2(1) = \frac{a(1+a)}{(1-a)^3}, \quad |a| < 1.$$

Meanwhile, for $a = 0.5$, these summations are equal to

$$S_1 = \sum_{k=0}^{\infty} k(0.5)^k = \frac{0.5}{(1-0.5)^2} = 2,$$
$$S_2 = \sum_{k=0}^{\infty} k^2(0.5)^k = \frac{0.5(1+0.5)}{(1-0.5)^2} = 6.$$

3.5 Response to Complex Exponentials

It is advantageous in the study of LTI systems to represent signals as linear combinations of basic signals that possess the following two properties:

1) The set of basic signals can be used to construct a broad and useful class of signals.

2) The response of an LTI system to each signal should be simple enough in structure to provide us with a convenient representation for the response of the system to any signal constructed as a linear combination of the basic signals.

Much of the importance of Fourier analysis results from the fact that both of these properties are provided by the set of complex exponential signals, e^{st} in continuous time and z^n in discrete time where s and z are complex numbers. Since, in subsequent sections, we will examine the first property in detail, we focus on the second property now. Especially, the second property provides motivation for the use of Fourier series and transforms in the analysis of LTI systems.

The importance of complex exponentials in the study of LTI systems stems form the fact that the response of an LTI system $\mathbb{L}\{\bullet\}$ with transfer function $H(\bullet)$ to a complex exponential input is the same complex exponential with only a change in its complex amplitude; that is,

$$\begin{aligned} \text{continuous time}: &\quad \mathbb{L}\{e^{st}\} = H(s)e^{st}, \\ \text{discrete time}: &\quad \mathbb{L}\{z^n\} = H(z)z^n. \end{aligned} \tag{3.45}$$

A signal for which the system output is a (possibly complex) constant times the input is referred to as an *eigenfunction* of the system, and the amplitude factor is referred to as the system's *eigenvalue*.

To show that complex exponentials are indeed eigenfunctions of LTI systems, let us consider a continuous-time LIT system with impulse response $h(t)$. For an input $x(t)$, we can determine the output through the use of the convolution integral, so that with $x(t) = e^{st}$.

$$y(t) = \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau = \int_{-\infty}^{\infty} h(\tau)e^{s(t-\tau)}d\tau = e^{st}\int_{-\infty}^{\infty} h(\tau)e^{-s\tau}d\tau \qquad (3.46)$$

Assuming that the integral on the right-hand side of the above equation converges, the response to e^{st} is of the form

$$y(t) = H(s)e^{st}, \qquad (3.47)$$

where $H(s)$ is a complex constant whose value depends on s and which is related to the system impulse response by

$$H(s) = \int_{-\infty}^{\infty} h(\tau)e^{-s\tau}d\tau. \qquad (3.48)$$

Hence, we have shown that complex exponentials are eigenfunctions of LTI systems. The constant $H(s)$ for a specific value of s is then the eigenvalues associated with the eigenfunction e^{st}.

Together with the superposition property, (3.47) implies that the representation of signals as a linear combination of complex exponentials leads to a convenient expression for the response of an LTI system. Specifically, if the input to a continuous-time LTI system is represented as a linear combination of complex exponentials,

$$x(t) = \sum_j a_j e^{s_j t}, \qquad (3.49)$$

then the output will be

$$y(t) = \sum_j a_j H(s_j)e^{s_j t}. \qquad (3.50)$$

In an exactly analogous manner, we can show that complex exponential sequences are eigenfunctions of discrete-time LTI system. That is, suppose that an LTI system with impulse response $h[n]$ has as its input the sequence

$$x[n] = z^n, \qquad (3.51)$$

where z is a complex variable. Then the output of the system can be determined from the convolution sum as

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] = \sum_{-\infty}^{\infty} h[k]z^{n-k} = z^n \sum_{k=-\infty}^{\infty} h[k]z^{-k}. \qquad (3.52)$$

From this expression, we see that, if the input $x[n]$ is the complex exponential given by (3.51), then, assuming that the summation on the right-hand side of (3.52) converges, the output is the same complex exponential multiplied by a constant that depends on the value of z. That is,

$$y[n] = H(z)z^n, \;\; H(z) = \sum_{k=-\infty}^{\infty} h[k]z^{-k}. \tag{3.53}$$

Consequently, as in the continuous-time case, complex exponentials are eigenfunctions of discrete-time LTI systems. The constant $H(z)$ for a specified value of z is the eigenvalue associated with the eigenfunction z^n.
Therefore, if the input to a discrete-time LTI system is represented as a linear combination of complex exponentials,

$$x[n] = \sum_j a_j z_j^n, \tag{3.54}$$

then the output will be

$$y[n] = \sum_j a_j H(z_j) z_j^n. \tag{3.55}$$

In other words, for both continuous time and discrete time, if the input to an LTI system is represented as a linear combination of complex exponentials, then the output can also be represented as a linear combination of the same complex exponential signals. Each coefficient in this representation of the output is obtained as the product of the corresponding coefficient a_j of the input and the system's eigenvalue $H(s_j)$ or $H(z_j)$ associated with the eigenfunction $e^{s_j t}$ or z_j^n, respectively. Although the variables s and z in the above equations may be arbitrary complex values in general, Fourier analysis involves restricting our attention to particular forms for these variables. In particular, in continuous time we focus on pure imaginary values of $s = j\omega$ and thus we consider only complex exponentials of the form $e^{j\omega t}$. Similarly, in discrete time we restrict the range of values of z to those of unit magnitude $z = e^{j\omega}$ so that we focus on complex exponentials of the form $e^{j\omega n}$.

EXAMPLE 3.7

Consider an LTI system for which the input $x(t)$ and output $y(t)$ are related by a time shift of 3.

$$y(t) = x(t-3)$$

If the input to this system is the complex exponential signal $x(t) = e^{j2t}$, then

$$y(t) = e^{j2(t-3)} = e^{-j6}e^{j2t}$$

The above result implies that the given system satisfies the eigenfunction property for the input e^{j2t}. The associated eigenvalue is $H(j2) = e^{-j6}$. It is straightforward to confirm (3.48) for this example. Specifically, the impulse response

of the system is $h(t) = \delta(t-3)$. Substituting it into (3.48) yields

$$H(s) = \int_{-\infty}^{\infty} \delta(\tau - 3) e^{s\tau} d\tau = e^{-3s},$$

so that $H(j2) = e^{-j6}$.

3.6 Discrete-Time Transfer Function

3.6.1 Transfer Function of a Discrete-Time System

Consider an LTI system with input $r[n]$ and output $y[n]$. If it is initially relaxed at $n = 0$, then $r[n] = y[n] = 0$, for $n < 0$. In this case, by the convolution theorem, the input and output can be described by

$$y[n] = \sum_{k=0}^{n} h[n-k] r[k]. \tag{3.56}$$

Applying the $\mathcal{Z}$-transform to $y[n]$ yields

$$Y(z) = \mathcal{Z}\{y[n]\} = \sum_{n=0}^{\infty} y[n] z^{-n} = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{\infty} h[n-k] r[k] \right) z^{-(n-k)} z^{-k}. \tag{3.57}$$

Interchanging the order of summations, introducing a new index $m = n - k$ and using the causality of the impulse response ($h[m] = 0$ for $m < 0$), we obtain

$$\begin{aligned} Y(z) &= \sum_{k=0}^{\infty} \left(\sum_{n=0}^{\infty} h[n-k] z^{-(n-k)} \right) r[k] z^{-k} \\ &= \sum_{k=0}^{\infty} \left(\sum_{m=-k}^{\infty} h[m] z^{-m} \right) r[k] z^{-k} \\ &= \left(\sum_{m=0}^{\infty} h[m] z^{-m} \right) \left(\sum_{k=0}^{\infty} r[k] z^{-k} \right) \\ &= H(z) R(z), \end{aligned} \tag{3.58}$$

where $R(z)$ and $Y(z)$ are the $\mathcal{Z}$-transform of the input and output, respectively. Moreover, $H(z)$ is called as the discrete-time transfer function which is defined as the $\mathcal{Z}$-transform of the impulse response.

$$H(z) = \mathcal{Z}\{h[n]\} = \sum_{n=0}^{\infty} h[n] z^{-n} \tag{3.59}$$

Thus, the $\mathcal{Z}$-transform converts the convolution (3.56) in time-domain into the algebraic multiplication (3.58) in z-domain. Because the convolution in (3.56) describes

only zero-state responses, whenever we use (3.58), the system is implicitly assumed to be initially relaxed. In the sequel, using (3.58), we can understand the meaning of transfer function in the following manner.

$$H(z) = \left.\frac{\mathcal{Z}\{\text{output}\}}{\mathcal{Z}\{\text{input}\}}\right|_{\text{all initial conditions are zero}} \tag{3.60}$$

Because the transfer function can be obtained from any input-output pair, the characteristics of an LTI system can be determined from any single pair of input and output.

3.6.2 Solving LTI Difference Equations

Every DT LTI lumped system can be expressed as a difference equation. It is interesting that we can readily find its solution and transfer function using $\mathcal{Z}$-transform.

EXAMPLE 3.8

Consider a DT LTI system described by the second-order difference equation

$$3y[k] + 2y[k-1] - y[k-2] = 2r[k-1] - 3r[k-2]. \tag{3.61}$$

As is well-known, its solution can be obtained by direct substitution. The solution, however, will not be in closed form, and it will be difficult to develop from the solution general properties of the equation. Now we apply the $\mathcal{Z}$-transform to study the equation. The equation is of second order; therefore, the response $y[k]$ depends on the input $r[k]$ and two initial conditions. To simplify discussion, we assume that $r[k] = 0$ for $k \leq 0$ and that the two initial conditions are $y[-1]$ and $y[-2]$. Using the one-step delay element z^{-1}, the application of the $\mathcal{Z}$-transform (3.17) and (3.21) for (3.61) yields

$$\begin{aligned} 3Y(z) + 2z^{-1}\big(Y(z) + z^1 y[-1]\big) - z^{-2}\left(Y(z) + z^1 y[-1] + z^2 y[-2]\right) \\ = 2z^{-1}R(z) - 3z^{-2}R(z) \end{aligned} \tag{3.62}$$

which can be rewritten as

$$Y(z) = \underbrace{\frac{-2y[-1] + z^{-1}y[-1] + y[-2]}{3 + 2z^{-1} - z^{-2}}}_{\text{zero-input response}} + \underbrace{\frac{2z^{-1} - 3z^{-2}}{3 + 2z^{-1} - z^{-2}} R(z)}_{\text{zero-state response}} \tag{3.63}$$

Note that the output in the transform domain consists of two parts: The first part is excited by the initial conditions and the second part is excited by the input. The former is the zero-input response, and the later is the zero-state response. This confirms the fact that the response of every DT LTI system can always be decomposed in this way.

If $y[-2] = 1$, $y[-1] = -2$, and $r[k]$ is a unit-step sequence. Then, we can obtain the closed-form solution to the given difference equation using $\mathcal{Z}$-transform pair.

$$\begin{aligned} Y(z) &= \frac{5 - 2z^{-1}}{3 + 2z^{-1} - z^{-2}} + \frac{2z^{-1} - 3z^{-2}}{3 + 2z^{-1} - z^{-2}} \frac{1}{1 - z^{-1}} \\ &= \frac{z(5z^2 - 4z - 2)}{(3z - 1)(z + 1)(z - 1)} \end{aligned} \tag{3.64}$$

To find its time response, we expand $Y(z)/z$ as

$$\frac{Y(z)}{z} = \frac{5z^2 - 4z - 2}{(3z-1)(z+1)(z-1)} = \frac{19}{8} \cdot \frac{1}{3z-1} + \frac{9}{8} \cdot \frac{1}{z+1} - \frac{1}{4} \cdot \frac{z}{z-1}, \tag{3.65}$$

and its inverse $\mathcal{Z}$-transform is

$$y[k] = \frac{19}{24} \left(\frac{1}{3}\right)^k + \frac{9}{8}(-1)^k - \frac{1}{4}(1)^k, \quad k = 0, 1, \cdots . \tag{3.66}$$

If all initial conditions are zero, $y[n] = u[n] = 0$, for $n < 0$, then the transfer function of the system becomes

$$H(z) = \frac{Y(z)}{R(z)} = \frac{2z^{-1} - 3z^{-2}}{3 + 2z^{-1} - z^{-2}}. \tag{3.67}$$

Let us compute the transfer function of the following difference equation.

$$\sum_{n=0}^{N} a_n y[k + n] = \sum_{m=0}^{M} b_m r[k + m] \tag{3.68}$$

If the given system is initially relaxed, we can simply use, as demonstrated in the preceding example, $\mathcal{Z}\{x[k + n] = z^n X(z)\}$, instead of (3.17) or (3.21), in computing the transfer function.

$$\left(\sum_{n=0}^{N} a_n z^n\right) Y(z) = \left(\sum_{m=0}^{M} b_m z^m\right) R(z) \tag{3.69}$$

Thus the transfer function of (3.68) is

$$H(z) = \frac{Y(z)}{R(z)} = \frac{\sum\limits_{m=0}^{M} \bar{b}_m z^m}{\sum\limits_{n=0}^{N} \bar{a}_n z^n} = \frac{\sum\limits_{m=0}^{M} b_m z^{-m}}{\sum\limits_{n=0}^{N} a_n z^{-n}}. \tag{3.70}$$

We call the first form as a *positive power transfer function* and the second form as a *negative power transfer function*, respectively. Either form can be easily transformed into the other form.

Unlike CT systems where we use exclusively positive-power transfer functions, we may encounter both forms in DT systems. Thus we discuss further the two forms. As in the CT case, the positive power form of $H(z)$ is improper if $N < M$, proper if $N \geq M$, strictly proper if $N > M$, and biproper if $N = M$. If the denominator and numerator of $H(z)$ has no common factors, then $H(z)$ has the degree of $\max(N, M)$. As an example, if $H(z)$ is proper, then its degree is N.

For the negative-power transfer function, we always assume $a_0 \neq 0,\ b_0 \neq 0$ and at least one of a_N and b_M are not zero. We now discuss its properness condition. Because the properness of a rational function has been defined for positive-power form, we must translate the condition into the negative power form. Recall that the properness of $H(z)$ can also be determined from $H(\infty)$. If $z = \infty$, then $z^{-1} = 0$ and $H(\infty)$ in the negative power form reduces to $\frac{b_M}{a_N}$. Thus, the negative power form becomes improper if $a_N = 0$ and $b_M \neq 0$, proper if $a_N \neq 0$, biproper if $a_N \neq 0,\ b_M \neq 0$, and strictly proper if $a_N \neq 0, b_M = 0$.

3.6.3 Poles and Zeros

3.6.3.1 Significance of Poles and Zeros This subsection introduces the concepts of poles and zeros of DT transfer functions. Consider a proper rational function $H(z) = N(z)/D(z)$ where $D(z)$ and $N(z)$ are polynomials of z with real coefficients. A finite real or complex number λ is called a zero of $H(z)$ if $H(\lambda) = 0$. It is a pole if $|H(\lambda)| = \infty$. If $N(z)$ and $D(z)$ are coprime, then all roots of $N(z)$ are the zeros of $H(z)$ and all roots of $D(z)$ are the poles of $H(z)$.

EXAMPLE 3.9

Consider the transfer function

$$H(z) = \frac{8z^3 - 24z - 16}{2z^5 + 20z^4 + 98z^3 + 268z^2 + 376z + 208}.$$

To find its zeros and poles, we can apply the MATLAB function `roots` to its numerator and denominator. We can also apply the function `tf2zp` as

```
num=[8 0 -24 -16]; den=[2 20 98 268 376 208];
[zeros,poles,K] = tf2zp(num,den);
```

which yields

```
zeros=[-1 -1 2]; poles=[-2 -2 -2 -2-j3 -2+j3]; K=4;.
```

Thus, $H(z)$ can be expressed as follows:

$$H(z) = \frac{4(z+1)^2(z-2)}{(z+2)^3(z+2+j3)(z+2-j3)}$$

It has a simple zero at 2 and simple poles at $-2 \pm j3$. A repeated zero is found at -1 with multiplicity 2 and a repeated pole lies at -2 with multiplicity 3. If

all coefficients of $H(z)$ are real, complex conjugate poles and zeros must appear in pairs. Note that $H(z=\infty)=0$, but we do not consider $z=\infty$ as a zero in usual because we conventionally take only finite zeros into account. We see that the discussion for the poles and zeros of a CT transfer function $H(s)$ can be directly applied to DT transfer function $H(z)$. However, it should be pointed out that the DC gain of a DT transfer function is calculated in different way from that of a CT transfer function. In other words, in the above example, K is not the DC gain of a DT transfer function. This is because there exists the nonlinear mapping relation between s and $z=e^{sT}$. For more explanations will be given in the subsequent subsections.

EXAMPLE 3.10

We next consider negative power transfer function such as

$$H(z)=\frac{4z^{-1}+6z^{-2}}{2+5z^{-1}+3z^{-2}}=\frac{4z^{-1}(1+1.5z^{-1})}{(2+z^{-1})(1+3z^{-1})}$$

Setting $1+1.5z^{-1}=0,\ z^{-1}=-1/1.5$ or $z=-1.5$, we have $H(-1.5)=0$. Thus -1.5 is a zero of $H(z)$. Likewise, we can verify that -1.2 and -3 are poles. Note that setting $z^{-1}=0$ or $z=1/0=\infty$, we have $H(\infty)=0$. But $z=\infty$ is not zero because we consider only finite zeros. Thus zeros and poles can also be obtained from negative power transfer functions.

Similar with the CT case, the poles and zeros of a DT transfer function are plotted on the complex z-plane with crosses and circles. However, unlike the CT case in which the s- plane is divided into the left and right half planes by the $j\omega$-axis, the complex z-plane is divided into the unit circle, its interior, and its exterior. This is expected in view of the mapping discussed in Figure 3.2.

In order to discuss about the effect of the pole location, let us start with real poles. Assuming α be real and positive, then the inverse $\mathcal{Z}$-transform of $\frac{z}{z(z-\alpha)}$ is $\alpha^n u[n]$. The time response $\alpha^n u[n]$ grows unbounded if $\alpha>1$, is a step sequence if $\alpha=1$, and vanishes if $\alpha<1$. If α is a repeated pole such as $\frac{z}{z(z-\alpha)^2}$, then its inverse $\mathcal{Z}$-transform is $n\alpha^n u[n]$. It grow unbounded if $\alpha\geq 1$. If $\alpha<1$, it approaches 0 because

$$\lim_{n\to\infty}\frac{(n+1)\alpha^{n+1}}{n\alpha^n}=\alpha<1 \tag{3.71}$$

following Cauchy's ratio test.
Analogous to the above example, we can also show that $n^k\alpha^n u[n]$ approaches 0 as $n\to\infty$ for any $\alpha<1$ and any positive integer k.

Next we consider the pole at $\alpha e^{j\pi}=-\alpha$ where α is real and positive. The pole is located on the negative real axis. The inverse $\mathcal{Z}$-transform of $\frac{z}{z-\alpha}$ is $(-\alpha)^n u[n]$. Thus its value changes sign at every sampling instant, and the time sequence contains

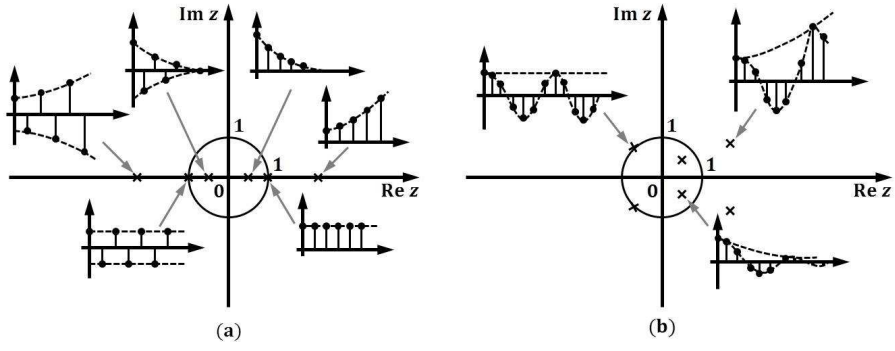

Figure 3.8 (a) Responses of real poles (b) Responses of complex conjugate poles

the highest-frequency component π. In order to understand the frequency component contained in $(-\alpha)^n u[n]$, let us assume $T = 1$ for the sake of convenience. In this case, the Nyquist frequency range becomes $(-\frac{\pi}{T}, \frac{\pi}{T}] = (-\pi, \pi]$ hence π is the highest frequency. The time sequence $(-\alpha)^n u[n]$ grows alternatively to infinity if $\alpha > 1$, is 1 or -1 if $\alpha = 1$, and vanishes alternatively if $\alpha < 1$ as shown in Figure 3.8(a). If the real and negative pole is repeated with multiplicity 2, then its time response $n(-\alpha)^n u[n]$ grows unbounded if $\alpha \geq 1$ and vanishes if $\alpha < 1$.

Let us consider $\alpha e^{j\omega_0}$ where α is real and positive and $0 < \omega_0 < \pi$. The complex pole together with its complex conjugate $\alpha^{-j\omega_0}$ generate the response $\alpha^n \sin(\omega_0 n)u[n]$ as shown in Figure 3.8(b). The response grows unbounded if $\alpha > 1$, is a sinusoidal sequence with frequency ω_0 if $\alpha = 1$, and vanishes if $\alpha < 1$. If the pair of complex conjugate poles is repeated, then its time response grows unbounded if $\alpha \geq 1$ and vanishes if $\alpha < 1$. In conclusion, the time response of a pole, simple or repeated, real or complex, approached zero if and only if the pole lies inside the unit circle.

We summarize the preceding discussion in the following:

pole location	steady-state response
interior of the unit circle, simple or repeated	0
exterior of the unit circle, simple or repeated	∞ or $-\infty$
unit circle, simple	constant or sustained ocsillations
unit circle, repeated	∞ or $-\infty$

EXAMPLE 3.11

Consider the following four DT transfer functions:

$$H_1(z) = 1.9712/D(z),$$
$$H_2(z) = 19.712(z - 0.9)/D(z),$$

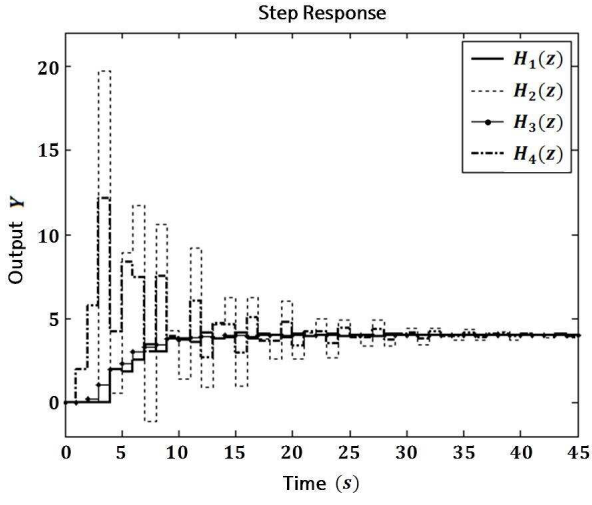

Figure 3.9 Step responses of the transfer function in Example 3.11

$$H_3(z) = 0.219(z^2 + 4z + 4)/D(z),$$
$$H_4(z) = 1.9712(z^3 + 2z^2 + 3z - 5)/D(z),$$

where the characteristic polynomial $D(z)$ is

$$\begin{aligned} D(z) &= z^4 + 0.07z^3 - 0.354z^2 - 0.5148z + 0.2916 \\ &= (z - 0.6)^2(z - 0.9e^{j2.354})(z - 0.9e^{-j2.354}). \end{aligned}$$

They have the same set of poles and the DC gain $H_i(1) = 4$. Even though they have different zeros, their step response can be expressed in terms of

$$Y(z) = H_i R(z) = H_i(z)\frac{1}{1 - z^{-1}}$$

which can be expanded as

$$\frac{Y(z)}{z} = \frac{k_1}{z - 0.6} + \frac{k_2}{(z - 0.6)^2} + \frac{\bar{k}_3}{z - 0.9e^{j2.354}} + \frac{\bar{k}_3^*}{z - 0.9e^{-j2.354}} + \frac{k_5}{z - 1}$$

with $k_5 = H_i(1) = 4$.

Thus their step responses in the time domain are all of the form,

$$y[n] = \Big(k_1(0.6)^n + k_1 n(0.6)^n + k_3(0.9)^n \cos(2.354n + k_4) + k_5\Big)u[n].$$

This form is determined solely by the poles of $H_i(z)$ and $R(z)$. The step responses of the transfer function $H_i(z)$ are depicted in Figure 3.9. They are obtained by using the MATLAB script.

```
den = [ 1 0.07 -0.354 -0.5148 0.2916 ];

num1 = 1.9712;
num2 = 19.712*[1 -0.9];
num3 = 0.2190*[1 4 4];
num4 = 1.9712*[1 2 3 -5];

Nf = 45;% final step

y1 = dstep(num1, den, Nf);
y2 = dstep(num2, den, Nf);
y3 = dstep(num3, den, Nf);
y4 = dstep(num4, den, Nf);

figure,  stairs(y1, '-');
hold on, stairs(y2, ':');
hold on, stairs(y3, '-');
hold on, stairs(y4, '-.');
```

Form the figure, even though all transfer functions have the same poles, their responses are all different due to different set of k_i. As is well-known, while the poles dictate the general form of responses, zeros affect only the coefficients k_i. Thus we conclude that zeros play a lesser role than poles in determining responses of systems.

3.7 Stability of a Discrete-Time System

This section introduces the concept of stability for DT systems. If a DT system is not stable, its response excited by any input will grow unbounded. Thus every DT system designed to process signals must be stable. Let us give a formal definition on the stability of DT systems.

Definition 3.1 *A DT system is BIBO (bounded-input bounded- output) stable or, simply, stable, if every bounded input sequence excites a bounded output sequence. Otherwise, the system is said to be unstable.*

A signal is bounded if it does not grow to ∞ or $-\infty$. In other words, a signal $f[n]$ is bounded if there exists a constant M_1 such that $|f[n]| \leq M_1 < \infty$ for all n. As in the CT case, it is unrealistic to determine the stability of a system using Definition 3.1 because there are infinitely many bounded inputs to be checked. However, if we can find a bounded input that excites an unbounded output, then we can conclude that the system is not stable. In fact, the stability of a DT system can be determined from its mathematical descriptions without applying any input. In other words, stability is

an inherent property of a system and is independent of applied inputs. The output of a stable system excited by any bounded input must be bounded; its output excited by an unbounded input is generally unbounded.

Definition 3.2 *A DT LTI system with impulse response $h[n]$ is BIBO stable if and only if $h[n]$ is absolutely summable in $[0, \infty)$. That is, for a constant M,*

$$\sum_{n=0}^{\infty} |h[n]| \leq M < \infty.$$

A necessary condition for $h[n]$ to be absolutely summable is that $h[n] \to 0$ as $n \to \infty$. However, the condition is not sufficient. For example, consider $h[0] = 0$ and $h[n] = 1/n$ for $n \geq 1$. It approaches 0 as $n \to \infty$. In order to confirm this, let us compute

$$S = \sum_{n=0}^{\infty} |h[n]| = \sum_{n=1}^{\infty} \frac{1}{n} = (1) + \left(\frac{1}{2}\right) + \left(\frac{1}{3} + \frac{1}{4}\right) + \left(\frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8}\right) + \cdots . \quad (3.72)$$

The sum inside every pair of parentheses is $\frac{1}{2}$ or larger, thus we have

$$S > 1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} \cdots = \infty, \quad (3.73)$$

and the sequence is not absolutely summable. Thus a DT system with sequence $1/n$ for $n \geq 1$ is not stable even though its impulse response approaches zero. Note that the $\mathcal{Z}$-transform of $1/n$ is an irrational function of z, thus the DT system is not a lumped system.

The sequence $1/n$ is not absolutely summable because it does not approach zero fast enough. If a sequence approaches 0 sufficiently fast, such as $b^n u[n]$ or $n^k b^n u[n]$ for $|b| < 1$ and any positive integer k, then it is absolutely summable. Indeed, for $|b| < 1$, we have

$$\sum_{n=0}^{\infty} |b|^n = \frac{1}{1 - |b|} < \infty,$$

and

$$\sum_{n=0}^{\infty} n^k |b|^n < \infty,$$

following Cauchy's ratio test

$$\lim_{n \to \infty} \frac{(n+1)^k |b|^{n+1}}{n^k |b|^n} = |b| < 1.$$

This is so because, as $n \to \infty$, $n^k |b|^n$ decreases in the ratio of $|b|^n$, which decreases exponentially to zero for $|b| < 1$. It approaches zero much faster than $1/n$.

Theorem 3.1 *A DT LTI lumped system with proper rational transfer function $H(z)$ is asymptotically stable if and only if every pole of $H(z)$ has a magnitude less than 1 or, equivalently, all poles of $H(z)$ lie inside the unit circle on the z-plane.*

If $H(z)$ has one or more poles lying outside the unit circle, then its impulse response grows unbounded and is not absolutely summable. If it has poles on the unit circle, then its impulse response will not approach 0 and is not absolutely summable. Thus if $H(z)$ has one or more poles on or outside the unit circle, then the system is not asymptotically stable. On the other hand, if every pole of $H(z)$ has a magnitude less than 1, then its time response approaches zero exponentially and is absolutely summable. Thus we conclude that the system is asymptotically stable. Note that the stability of a system is independent of the zeros of its transfer function. The zeros can be located outside, on, or inside the unit circle.

EXAMPLE 3.12

Consider a DT system with transfer function

$$H(z) = \frac{(z+2)(z-10)}{(z-0.9)(z+0.95)(z+0.9+j0.7)(z+0.9-j0.7)}.$$

Its two real pols have magnitudes less than 1. On the contrary, the magnitude of the complex poles is computed by

$$\sqrt{(0.9)^2 + (0.7)^2} = 1.14 > 1.$$

Thus the system is not stable.

EXAMPLE 3.13

Consider a DT system with N^{th}-order finite impulse response (FIR). That is, it has an impulse response $h[n]$ for $n = 0, 1, \cdots, N$, and $h[n] = 0$ for $n > N$. Therefore, we get

$$\sum_{n=0}^{\infty} |h[n]| = \sum_{n=0}^{N} |h[n]| < \infty.$$

Thus, every FIR filter is stable.
This can also be concluded from its transfer function

$$H(z) = \sum_{n=0}^{N} h[n]z^{-n} = \frac{h[0]z^N + h[1]z^{N-1} + \cdots + h[N-1]z^1 + h[N]}{z^N}.$$

Since all its poles are located at $z = 0$ inside of the unit circle, the FIR system is stable.

The stability of $H(z)$ can be checked based on Jury's test result without computing its poles. The Jury's test can be understood as a DT version of the well-known Routh-Herwitz test for CT systems. To do this, firstly, we need to define *a polynomial to be DT stable if all its roots have magnitude less than* 1. As mentioned earlier, the Jury's test might be an efficient way to check whether the polynomial is DT stable or not. As an example, we use the following polynomial of degree 5 to illustrate the procedure:

$$D(z) = a_0 z^5 + a_1 z^4 + a_2 z^3 + a_3 z^2 + a_4 z^1 + a_5, \;\; a_0 > 0 \tag{3.74}$$

where the leading coefficient a_0 is assumed to be positive without loss of generality. For the polynomial $D(z)$, the Jury's table is shown in the following.

a_0	a_1	a_2	a_3	a_4	a_5	
a_5	a_4	a_3	a_2	a_1	a_0	$k_1 = a_5/a_0$
b_0	b_1	b_2	b_3	b_4	0	(1^{st} a_i row) $-$ k_1(2^{nd} a_i row)
b_4	b_3	b_2	b_1	b_0		$k_2 = b_4/b_0$
c_0	c_1	c_2	c_3	0		(1^{st} b_i row) $-$ k_2(2^{nd} b_i row)
c_3	c_2	c_1	c_0			$k_3 = c_3/c_0$
d_0	d_1	d_2	0			($1^{st} - c_i$ row) $-$ k_3(2^{nd} c_i row)
d_2	d_1	d_0				$k_4 = d_3/d_0$
e_0	e_1	0				(1^{st} d_i row) $-$ k_4(2^{nd} d_i row)
e_1	e_0					$k_5 = e_1/e_0$
f_0	0					(1^{st} e_i row) - k_5(2^{nd} e_i row)

The first row is filled out simply using the coefficients of $D(z)$ arranged in the descending power of z. The second row is the reversal of the first row. We compute $k_1 = a_5/a_0$, the ratio of the last entries of the first two rows. The first b_i row is obtained by subtracting from the first a_i row the product of the second a_i row and k_1. Note that the last entry of the first b_i row is automatically zero and is discarded in the subsequent discussion. We then reverse the order of b_i to form the second b_i row and compute $k_2 = b_4/b_0$. The first b_i row subtracting the product of the second b_i row and k_2 yields the first c_i row. We repeat the process until the table is completed. We call b_0, c_0, d_0, e_0 and f_0 the *leading coefficients*. If $D(z)$ has degree N, then the table has N subsequent leading coefficients.

Theorem 3.2 *A polynomial with a positive leading coecient is DT stable if and only if every subsequent leading coefficient is positive. If any subsequent leading coefficient is 0 or negative, then the polynomial is not DT stable.*

The proof of this theorem is beyond the scope of this text, hence it is omitted.

EXAMPLE 3.14

For the polynomial $D(z) = 2z^3 - 0.2z^2 - 0.24z - 0.08$, we can form the Jury's table as follows:

2	-0.2	-0.24	-0.08	
-0.08	-0.24	-0.2	2	$k_1 = -0.004$
1.9968	-0.2096	-0.2480	0	
-0.2480	-0.2096	1.9968		$k_2 = -0.124$
1.966	-0.236	0		
-0.236	1.996			$k_3 = -0.12$

The three subsequent leading coefficients are all positive. Thus the polynomial $D(z)$ is DT stable.

3.8 Time-Response of a Discrete-Time System

3.8.1 Time Constant

Before discussing about the time-response of a DT system, let us consider the following example.

EXAMPLE 3.15

Consider the transfer function $H_1(z)$ in Example 3.11. Its step response was computed as, for $n \geq 0$,

$$y[n] = k_1(0.6)^n + k_1 n(0.6)^n + k_3(0.9)^n \cos(2.354n + k_4) + H_1(1), \quad \forall n \geq 0.$$

Because the system is asymptotically stable, the response approaches the steady-state as $n \to \infty$.

$$y_{ss}[n] = \lim_{n \to \infty} y[n] = H_1(1) = 4$$

Mathematically speaking, it takes an infinite amount of time for the response to reach steady-state. However, in practice, we often consider the response to have reached steady-state if the response reaches and remains within $\pm 1\%$ of its steady-state value.

At this point, a natural question may arise. *How fast will the response reach the steady-state?* This question is extremely important in designing control systems. Giving an answer to this question, let us define the transient response $y_{tr} = y_n - y_{ss}[n]$. For the transfer function $H_1(z)$ with unit step input, the transient response becomes

$$y_{tr}[n] = \Big(k_1(0.6)^n + k_2 n(0.6)^n + k_3(0.9)^n \cos(2.35n + k_4)\Big) u[n].$$

Clearly the faster the transient response approaches zero, the faster the total response reaches the steady-state.

As mentioned in the previous section, the form of the transient response $y_{tr}[n]$ is dictated only by the poles of $H_1(z)$. Because $H_1(z)$ is stable, all terms in $y_{tr}[n]$ approach zero as $n \to \infty$. The smaller the magnitude of a pole, the faster its time response approaches zero. Thus the time for the transient response to approach zero is dictated by the pole that has the largest magnitude.

Now, consider the concept of time constant for the sequence $b^n u[n] = \mathcal{Z}\left\{\frac{z}{z(z-b)}\right\}$ with $|b| < 1$. The DT signal $b^n u[n]$ decreases to less than 1% of its peak magnitude in five time constants. We can extend the concept to the general case. Let $H(z)$ be a stable proper rational function and let $|b|$ be the largest magnitude of all poles. Because $H(z)$ is stable, we have $|b| < 1$. We define the *time constant* of $H(z)$ as

$$t_c = \frac{-1}{\ln |b|}. \tag{3.75}$$

Note that t_c is positive because $|b| < 1$. Then, generally, the transient response of $H(z)$ will decrease. In Example 3.11, all transfer functions have the same time constant $\frac{-1}{\ln 0.0} = 9.47$ and, as shown in Figure 3.9, their responses reach steady-state in around $(5 \times 9.47 = 47.35) \approx 47$ samples.

It is important to mention that the rule of five time constants should be used only as a guide. It is possible to construct examples whose transient responses will not decrease to less than 1% of their peak magnitudes in five time constants for the effect of zeros. However, it is generally true that *the smaller the time constant, the faster the system response*.

3.8.2 FIR and IIR Systems

DT LTI systems can be classified as FIR (finite impulse response) and IIR (infinite impulse response) systems. We discuss their transfer functions in this subsection. For your reference, all CT LTI systems, excluding memoryless systems, are IIR.Thus we do not classify CT LTI systems as FIR or IIR systems.

Consider a N^{th}-order FIR system with impulse response $h[n]$ for $n = 0, \cdots, N$. Note that it has length of $N + 1$ hence its transfer function becomes

$$\begin{aligned} H(z) &= h[0] + h[1]z^{-1} + \cdots + h[N]z^{-N} \\ &= \frac{h[0]z^N + h[1]z^{N-1} + \cdots + h[N]}{z^N}. \end{aligned} \tag{3.76}$$

The above transfer function has degree N and all its N poles are located at $z = 0$. Every FIR system has poles only at the origin of the z-plane. Conversely, if transfer function has poles only at $z = 0$, then it describes an FIR system.

EXAMPLE 3.16

Consider a N-points moving average filter. It has impulse response $h[n] = 1/N$ for $n = 0, 1, \cdots, (N-1)$. Thus its transfer function is

$$H(z) = \frac{1}{N} \sum_{n=0}^{N-1} z^{-n} = \frac{1}{Nz^{N-1}} \sum_{n=0}^{N-1} z^n.$$

It is a biproper rational function with degree $(N-1)$.
We can rewrite the transfer function as

$$H(z) = \frac{Y(z)}{R(z)} = \frac{1 - z^N}{Nz^{N-1}(1-z)} = \frac{1}{N} \cdot \frac{z^N - 1}{z^{N-1}(z-1)}.$$

Because its numerator and denominator have the common factor $z = 1$, $z = 1$ is not a pole of the transfer function. Therefore, the assertion that an FIR system has poles only at $z = 0$ is still valid.
From the above equation, we have

$$z^N Y(z) - z^{N-1} Y(z) = \frac{1}{N} \left(z^N R(z) - R(z) \right).$$

Thus, its time-domain description is

$$y[n+N] - y[n+(N-1)] = \frac{1}{N} \left(r[n+N] - r[n] \right).$$

It can be transformed into the delayed form as

$$y[n] - y[n-1] = \frac{1}{N} \left(r[n] - r[n-N] \right),$$

which can be implemented recursively.

While every DT FIR system has poles only at $z = 0$, every DT IIR system has at least one pole other than $z = 0$. The reason is as follows. The inverse $\mathcal{Z}$-transform of $z/(z-a)$ is $a^n u[n]$, which has, if $z \neq 0$, infinitely many nonzero entries. If $H(z)$ has one or more poles other than $z = 0$, then its impulse response (inverse $\mathcal{Z}$-transform of $H(z)$) has infinitely many nonzero entries. Thus $H(z)$ describes an IIR filter.

3.9 Frequency Response of a Discrete-Time System

In order to consider the implication of stability, we introduce the concept of frequency responses. Consider the values of DT transfer function $H(z)$ along the

unit circle on the z-plane. Now, we define these values, that is, $H(e^{j\omega T})$ for all ω as the *frequency response*. In general, $H(e^{j\omega T})$ is complex-valued and can be expressed in polar form as

$$H(e^{j\omega T}) = A(\omega)e^{j\theta(\omega)}, \tag{3.77}$$

where T means the sampling period, $A(\omega)$ and $\theta(\omega)$ are real-valued functions of ω and $A(\omega) \geq 0$. We call $A(\omega)$ the *magnitude response* and $\theta(\omega)$ the *phase response*, respectively.

EXAMPLE 3.17

Given the DT transfer function $H(z) = \frac{z+1}{10z-8}$ and $T = 1$, its frequency response is

$$H(e^{j\omega}) = \frac{e^{j\omega}+1}{10e^{j\omega}-8}.$$

Therefore, we can have

$$\begin{aligned} H(z=e^{j\omega})\big|_{\omega=0} &= H(z=1) \quad = 1 \cdot e^{j0}, \\ H(z=e^{j\omega})\big|_{\omega=\frac{\pi}{2}} &= H(z=j) \quad = 0.11e^{-j1.46}, \\ H(z=e^{j\omega})\big|_{\omega=\pi} &= H(z=-1) = 0, \end{aligned}$$

Other than these ω, the frequency response $H(e^{j\omega})$ is always complex. Fortunately, the MATLAB function `freqz`, where the last character `z` stands for the $\mathcal{Z}$-transform, carries out the computation. To compute the frequency response of $H(z)$ from $\omega = -10$ to 10 with increment 0.01,

```
num = [1 1]; den=[10 -8];
w = -10:0.01:10;
Hz = freqz(num,den,w);
figure, plot(w, abs(Hz), w, angle(Hz), ':');
```

The result is shown in Figure 3.10. We see that the frequency response is periodic with period 2π. This is expected because the Nyquist frequency range is $[-\pi, \pi)$ or $(-\pi, \pi]$ for $T = 1$. It also follows from

$$e^{j\omega T} = e^{j(\omega T + 2\pi)}.$$

If all coefficients of $H(z)$ are real, then we have $H(e^{j\omega T}) = [H(e^{-j\omega T})]^*$, which implies

$$A(\omega) = A(-\omega) \text{ (even)}, \quad \theta(\omega) = -\theta(-\omega) \text{ (odd)}. \tag{3.78}$$

In other words, if $H(z)$ has only real coecients, then its magnitude response is even and its phase response is odd. Thus we often plot frequency responses only in the positive frequency range $(0, \pi]$.

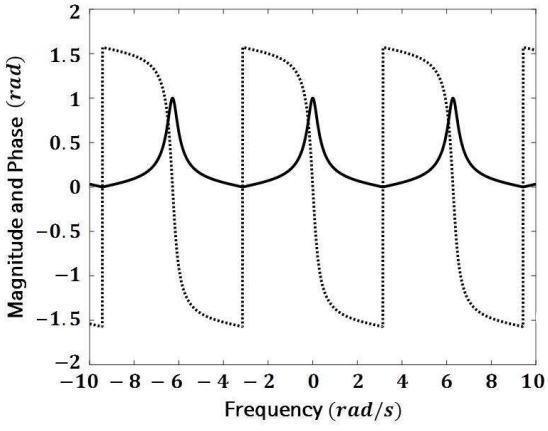

Figure 3.10 Frequency response of Example 3.17: $|H(z)|$(solid line), $\angle H(z)$(dotted line)

We now discuss the implication of stability and physical meaning of frequency responses. Consider a DT system with transfer function $H(z)$ and $T = 1$. Let us apply to it the input $r[n] = ae^{j\omega_0 n}u[n]$. If the system is not asymptotically stable, then the output will grow unbounded or maintain an oscillation with a frequency different from ω_0. However, if the system is stable, then the output approaches, as we show next, $aH(e^{j\omega_0})e^{j\omega_0 n}u[n]$ as $n \to \infty$.

Since $\mathcal{Z}\{r[n] = ae^{j\omega_0 n}u[n]\} = \frac{az}{z-e^{j\omega_0}}$, the output of the system satisfies

$$\begin{aligned} Y(z) = H(z)R(z) &= H(z)\frac{az}{z-e^{j\omega_0}} \\ &= k_1\frac{z}{z-e^{j\omega_0}} + \text{terms due to poles of } H(z), \end{aligned} \tag{3.79}$$

where $k_1 = aH(z)|_{z=e^{j\omega_0}} = aH(e^{j\omega_0})$. The above equation implies that

$$y[n] = aH(e^{j\omega_0})e^{j\omega_0 n}u[n] + \text{response due to poles of } H(z). \tag{3.80}$$

If $H(z)$ is stable, then all responses due to its poles approach 0 as $n \to \infty$. Then, the steady-state response becomes

$$\begin{aligned} y_{ss}[n] &= \lim_{n\to\infty} y[n] \\ &= aH(e^{j\omega_0})e^{j\omega_0 n}u[n] \\ &= aA(\omega_0)e^{j\theta(\omega_0)}e^{j\omega_0 n}u[n] \\ &= aA(\omega_0)\Big(\cos(\omega_0 n + \theta(\omega_0)) + j\sin(\omega_0 n + \theta(\omega_0))\Big)u[n]. \end{aligned} \tag{3.81}$$

For BIBO stable DT system with transfer function $H(z)$ with $T = 1$, some special cases of the above results are listed below:

$$\begin{aligned} r[n] = au[n] &\rightarrow y_{ss}[n] = aH(1)u[n], \\ r[n] = a\sin(\omega_0 n)u[n] &\rightarrow y_{ss}[n] = a|H(e^{j\omega_0 n})|\sin(\omega_0 n + \angle H(e^{j\omega_0}))u[n], \\ r[n] = a\cos(\omega_0 n)u[n] &\rightarrow y_{ss}[n] = a|H(e^{j\omega_0 n})|\cos(\omega_0 n + \angle H(e^{j\omega_0}))u[n] \end{aligned} \quad (3.82)$$

EXAMPLE 3.18

Consider a system with transfer function $H(z) = \frac{a+1}{10z+8}$ with $T = 1$. The system has pole at $z = -0.8$ which has a magnitude less than 1, hence it is asymptotically stable. Let us compute the steady-state response of the system excited by

$$\begin{aligned} r[n] &= 2u[n] + \sin(6.38n)u[n] + 0.2\cos(3n)u[n] \\ &= 2u[n] + \sin(0.1n)u[n] + 0.2\cos(3n)u[n], \end{aligned} \quad (3.83)$$

where $(2 + \sin(0.1n))u[n]$ and $0.2\cos(3n)u[n]$ will be considered as a desired signal and a unwanted noise, respectively.

Since $H(1) = 1,\ H(e^{j0.1}) = 0.9e^{-j0.4}$ and $H(e^{j3}) = 0.008e^{-j1.6}$, applying the concept of frequency response to $r[n]$ yields

$$y_{ss}[n] = 2 + 0.9\sin(0.1n - 0.4) + 0.0016\cos(3n - 1.6).$$

We see that the noise is successfully eliminated by the system. The system passed low-frequency signals and suppresses high-frequency signals. Therefore, this system can be regarded as a low-pass filter.

Note that, in order to define the frequency response, the condition of stability is essential. If it is not the case, then we cannot find the frequency response from experiments. In the above example, it is assumed that $T = 1$, thus the Nyquist frequency range is $(-\pi, \pi]$. Because the frequency $6.38(rad/s)$ is outside the Nyquist frequency range, we have subtracted $2\pi = 6.28(rad/s)$ from it to bring it inside the range as $6.38 - 6.28 = 0.1(rad/s)$.

EXAMPLE 3.19

Consider a system whose transfer function is $H_1(z) = \frac{z+1}{-8z+10}$ with $T = 1$. Note that this system is unstable because it has a pole at $z = 1.25$. Since $H_1(e^{j0.1} = 0.91e^{j0.42})$, if we apply the input $r[n] = \sin(0.1n)u[n]$, the output can be computed as

$$y[n] = -0.375(1.25)^n u[n] + 0.91\sin(0.1n + 0.42)u[n].$$

The second term is buried by the first term determined by the unstable pole as $n \rightarrow \infty$. Thus the output grows unbounded and the frequency response is meaningless for this unstable system.

CHAPTER 4

LABORATORY PRACTICES: MOBILE ROBOT APPLICATION

4.1 Overview of a Mobile Robot Hardware

Recently, mobile robots have been received much interest by many engineers because, instead of human beings, it can carry out various difficult or dangerous missions such as exploring, surveillance, search and rescue, transportation, and so on. It is well-known that the guidance, navigation and control are core technologies for developing advanced mobile robots.

Based on this understanding, in our laboratory practices, we will design the digital signal processing algorithms and the digital controllers for the mobile robot with a 1-axis ultrasonic gimballed seeker. As shown in Figure 4.1, the gimballed seeker measures the bearing and range information between the robot and the target emitting a ultrasonic signal. Using these measurements, the robot controller guides the robot towards the target. In this point of view, the gimballed seeker is crucial for improving the overall performance of mobile robot systems. Note that the gimballed seeker contains double control loops to control the gimbal head as depicted in Figure 4.2. The inner loop involved with the inertial stabilizing controller aims to decouple the gimbal assembly from the disturbance mainly caused by the angular motion of the robot. On the other hand, the outer loop with the angle tracking controller enables

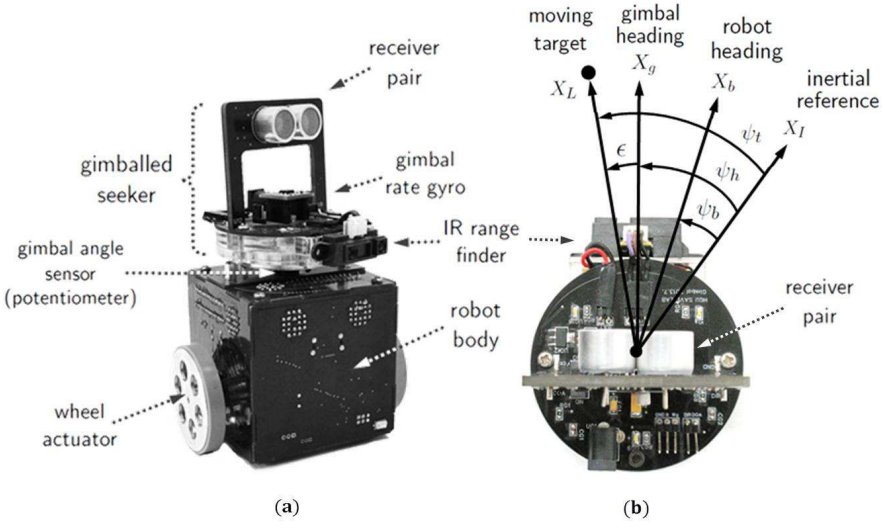

Figure 4.1 Mobile robot with 1-axis ultrasonic gimballed seeker

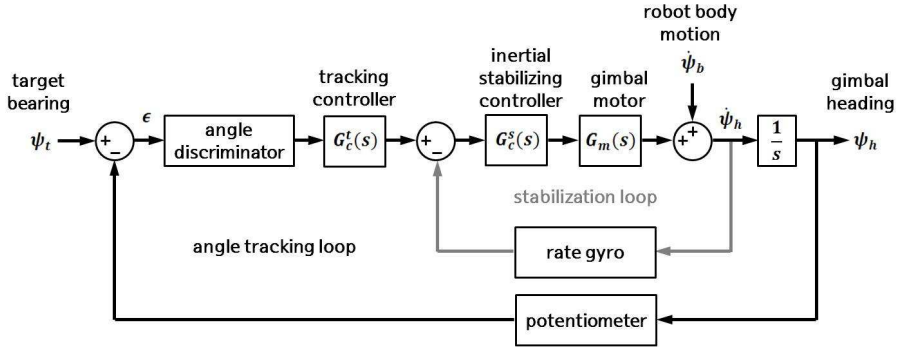

Figure 4.2 Control loops for the gimballed seeker

the gimbal to head towards the moving target in order for minimizing the boresignt error (BSE) ϵ provided by angle discriminator.

The technical issues covered by laboratory practices are summarized as follows:

1) Efficient software architecture for implementing the real-time digital system
2) Digital signal processing and filter design techniques to process sensor measurements used for feedback information of a digital controller
3) Control system design and analysis based on digital equivalence
4) Recursive least-squares estimation approach to parametric system identification in discrete-time domain

5) Systematic design and analysis of a digital control system to meet various specifications such as rising time, overshoot, steady-state error, relative stability margin, disturbance rejection performance and so on.
6) Bandwidth assignment technique for designing a control system with multiple feedback loops in view of classical control

4.2 Real-Time Software Architecture for Digital Systems

Purpose of the Experiment

In this experiment, we will learn about the desirable software architecture for real-time implementation of a control system. Once the concept of *real-time* processing is redefined in discrete-time domain, the *task scheduling* is emphasized as a matter of developing a real-time control software. We will write our own real-time ANSI-C program on a digital computer equips with a multi-purpose data acquisition board (DAQ) for signal interface.

Basic Concept of Real-Time Digital Control

Generally, the term, *real-time*, means that an event happens *instantly* at specific time. It is natural to apply this concept of real-time processing for analog control systems because the additional time is not required when the analog controller calculates and produces the control command. Unlike analog control systems, the sample and hold operation of input/output signals is inevitable to generate the control command of a digital controller. Bearing this distinctive feature in mind, the definition of *real-time* processing in digital control systems need to be relaxed to a case that the whole control tasks including signal acquisition, execution of a control algorithm and generation of control command are completed within one sampling period. This idea comes from the fact that the time increases with discrete manner by the sampling period in digital control systems.

In many cases, the *task scheduling* is often introduced as an effective way for developing a real-time control software. To do this, we should check the required time for completing each task beforehand and then allocate the appropriate time for each task so that all tasks do not overlap within one sample period. If the control software fails to complete all required tasks in every single sampling period, the control performance might be severely degraded or the digital control system may become unstable in the worst case. Therefore, the task scheduling is compulsory for implementing real-time digital control systems.

Software Architecture for Real-Time Digital Control

There are two kinds of real-time control software: hardware-timed or software-timed. Most digital controllers make use of the hardware-timed control software whose the operation is synchronized with a precise external timer or clock. However, if an external timer is unavailable, the viable alternative is to implement the

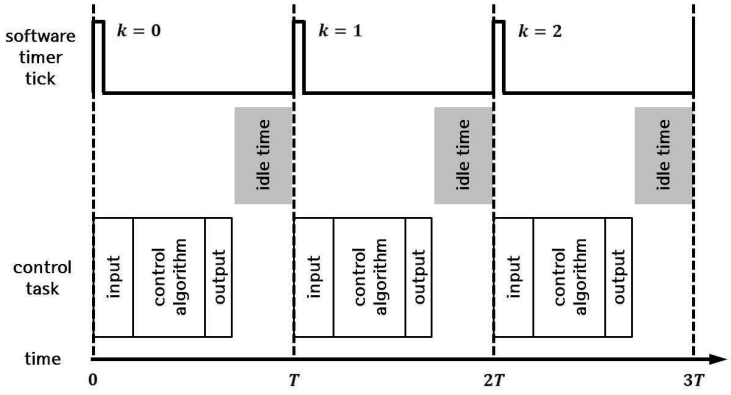

Figure 4.3 Timing diagram for software-timed digital control systems

control software using a virtual software timer. Actually, it is just a program which controls the timing sequence by exploiting the time offered by the operating system of a digital computer. This approach is often called the software-timed control. In this experiment, we will adopt the software-timed control scheme because it is convenient for rapid development and verification of a control software on our personal computer.

The timing diagram for control systems with a software timer is illustrated in Figure 4.3. When a control cycle starts, the digital control system acquires the feedback information measured by sensors using the DAQ which acts as an A/D. In sequence, a control algorithm calculates the control command. Then, the command is transferred to an actuator using the DAQ as a D/A. After completing these control tasks, the control software should let the time pass until one sampling period elapses. If this procedure is done, a software time tick is generated and the time is updated by the sampling period T.

For example, to check whether a new control cycle has come or not, you could use the following function, `CheckWindowsTime(void)` under MS Windows environments.

```
#include <windows.h>
#include <time.h>

/* check the windows time in [ms] */
double CheckWindowsTime(void)
{
    LARGE_INTEGER  liCount, liFreq;
    QueryPerformanceCounter  (&liCount);
    QueryPerformanceFrequency(&liFreq);
    return( (liCount.QuadPart/((double)(liFreq.QuadPart)))*1000.0 );
}
```

Surely, the similar task scheduling technique can be applied for the hardware-timed digital control system. The only difference is to use the time tick generated by the external clock, not the software timer.

Note that, different from the general event-driven application softwares, most real-time control softwares executes the predetermined functions or logics in a sequential manner. The following pseudo-code outlines the typical structure of a real-time control software.

```
Global_Variable_Declaration
  – Define the required variables such as the sampling time, the control gains, etc.
  – Declare the memory buffers which can be used for debugging of the program
void main(void)
{
    Program_Initialization;
    – It is strongly recommended to initialize all the variables beforehand
    – Configure all input and output channels needed for signal interface
    do{
        Import_Data;
        – Sensor measurements are acquired through input channels
        – Input data is preprocessed for unit conversion or calibration
        – Digital filters can be also applied for the measurements
        Digital_Controller;
        – Implement digital controllers in the form of difference equations
        – The control command and logic are recursively updated
        – The internal variables are stored in memory buffer for analysis
        Export_Data;
        – Control commands are postprocessed to interface with actuators
        – The resultant output signals are transferred via output channels
        Time_Management
        – Idle the time until the new sampling instance arrives
        – On a sampling instance, update the time tick and simulation time
    } while( Check_Stop_Condition );
    Data_Recording;
    – Write the data stored in memory buffers on the file for performance analysis
    – Do not print out and/or write the data on the file during tasks are running
    Program_Termination;
    – Stop and clear all input and output channels
    – Set free allocated memories including file pointers .
}
```

Experiment #1.1. *ANSI C Program for Setting and Activating a DAQ*

1) Create a new console application in Visual C.

   Firstly, check whether the *Measurement&Automation Explore* (MAX) software is installed on your computer. Search the files, `NIDAQmx.h` and `NIDAQmx.lib`, in your hard disk drive and copy them to the project directory. Include the header file at the beginning of a source file and add the library file in your project.

2) Write your own ANSI C program containing the following functions.

   - Through channel `AI0`, it acquires an analog sinusoidal input with magnitude $2[V]$ and frequency $20[Hz]$ generated by a function generator.
   - It provides two analog outputs through `AO0` and `AO1`, simultaneously.
     - `AO0` : sinusoidal signal measured through the channel `AI0`
     - `AO1` : DC signal with magnitude $3[V]$
   - If the user inputs the enter key, stop the analog output task and clear it.
   - It can calculate the maximum time required for each task, analog input and output, and print out it on the screen when the program is terminated.

   For your programming, it might be helpful to use the following functions. For more details, please refer to the lecture note provided in class or the reference found in `\National Instrument\NI-DAQ\ NI-DAQmx C Reference`.

   - channel creation : `DAQmxCreateTask`
   - channel configuration : `DAQmxAOVoltageChan`
   - starting the channel : `DAQmxStartTask`
   - analog output : `DAQmxWriteAnalogScalarF64`
   - analog input : `DAQmxReadAnalogScalarF64`
   - channel clearance : `DAQmxStopTask`, `DAQmxClearTask`
   - error diagnosis : `DAQmxErrChk` (optional)
   - OS time measurement : `GetWindowsTime`
   - standard I/O : `getchar, getch`
   - console I/O : `kbhit`

3) Run your program and, using your oscilloscope, check whether the analog output is properly generated. Compare the phase shift between the input and output signals measured at `AI0` and `AO0`.

Discussion and Analysis

1) Consider the reason why the phase shift is observed. Analyze the result based on the task execution time calculated by your program.

2) According to the experimental results, what is the fastest sampling time when you design a real-time system acquiring two analog inputs and generating one analog output without an additional signal processing or control algorithm?

Experiment #1.2. *Real-Time Check Using a Measurement Equipment*

1) Modify the console application in Experiment #1.1 by implementing a software timer whose sampling period `SAMPLING_TIME` is adjustable. To make a software timer in your program, you can use the function `GetWindowsTime` within the sentence `while/break`. Basically, a software timer should provide the following information which is necessary for implementing a real-time digital control system later.

   - time tick : `count`
   - simulation time : `Time=SAMPLING_TIME*(double)(count)`

2) Please note that the discrete-time generated by a software timer may contain an error, but this error should not be accumulated as time goes on. Moreover, this error should be small enough, hence it can be negligible compared to the sampling period. To prove that the software timer works properly, why don't you plot the error of the simulation time?

3) Using a function generator, generate a sinusoidal signal with magnitude $2[V]$ and frequency $5[Hz]$ and insert it into the analog input terminal of a DAQ.

4) With the sampling time 10[ms], execute your console application to acquire the analog input signal through `AI0` and directly output it through `AO1`.

5) Measure the input and output signals by using an oscilloscope. Observe the waveforms of input and output signals. Is there any distortion in the output waveform? How much time delay occurs in the output signal?

6) Repeat 4)$\sim$5) for the sampling time $8[ms], 6[ms], 4[ms]$ and $2[ms]$.

7) Plot the time-delay with respect to the sampling time.

8) For the sinusoidal inputs with frequency $50[Hz]$ and $250[Hz]$, repeat 4)$\sim$7).

Discussion and Analysis

1) Using the experimental results, analyze the relations among the frequency of an input signal, the sampling period, and the phase delay of a output signal.

2) If the DAQ output waveform is distorted, explain the reason in view of real-time digital processing.

Experiment #1.3. *Real-Time Check Using a Program*

1) Referring to the following directions, modify the console application in Experiment #1.2.

   - Set the sampling time as `SAMPLING_TIME` = $5[ms]$ and the final time as `FINAL_TIME` = $10[sec]$.
   - Declare the memory buffers `bufTime[N_DATA], bufInput[N_DATA], bufOutput[N_DATA]`, and `bufChkTaskTime` where `N_DATA` means the number of time ticks generated until `FINAL_TIME`.
   - Create the digital output channel `P0.0` as well as the analog input `AI0` and output channels `AO0`.
   - As in Experiment #1.2, acquire the sinusoidal input with frequency $5[Hz]$.
   - If the acquired value of an analog input is positive, the discrete output signal should be logic high (1) and, otherwise, logic low (0).
   - Count the number of changes from 0 to 1. Once more than 5 changes occur, start to save the acquired input, generated digital output and the simulation time into the corresponding memory buffers.
   - On a new sampling instance, increase the time tick `count` by 1 and update the simulation time `Time`. If every sampling period, check the time required for performing input/output tasks and the algorithm. Save it in `bufChkTaskTime`.
   - If the simulation time, `Time`, exceeds the final time, `FINAL_TIME`, the tasks should be automatically terminated.
   - After terminating the program, write all data stored in memory buffer on the file, `Result.txt`, as a text format.

2) Using Matlab, plot the input signal stored in the file. Does the signal have the frequency component of $5[Hz]$? In addition, plot the whole task execution time which is stored in `bufChkTaskTime`.

3) Repeat 1)$\sim$2) for `SAMPLING_TIME` = $2[ms]$.

Discussion and Analysis

Through the results of Experiment #1.3, explain the importance of task scheduling (or setting the adequate sampling period). Which problems may occur if the task is scheduled improperly? If possible, give an example to support your explanation.

4.3 Sampling and DT Spectra

Purpose of the Experiment

This experiment aims to show the sampling effects on the spectrum of a DT signal. The fast Fourier transform (FFT) algorithm will be implemented by programming and used for analyzing the spectrum of a sampled signal. Based on the experimental results, we can have a good grasp on the physical meaning of the sampling in view of signal reconstruction. Furthermore, as the second best way to avoid the frequency aliasing, the anti-aliasing filter will be taken into account.

Experiment #2.1. *Aliasing Effect*

Let us examine the effect of frequency aliasing. Theoretically speaking, the aliasing effect occurs whenever a signal is sampled at a rate less than twice the maximum frequency component contained in the given signal. This phenomenon is easily demonstrated using the under-sampled single sinusoid. More complicated situation may be observed if we sample a general signal with a vast number of frequency components.

Now, in order to confirm the aliasing effect, construct the experimental environment as shown in Figure 4.4. For implementing the anti-aliasing filter and the smoothing filter, use the analog circuit depicted in Figure 4.5.

1) Set the sampling rate of your DAQ as $200[Hz]$. Using its analog output channel, generate the sinusoidal signal $x_1(t) = A_1 \cos(2\pi f_1 t) + b$ with $A_1 = 1[V_{pp}]$, $f_1 = 10[Hz]$ and $b = 1[V]$.

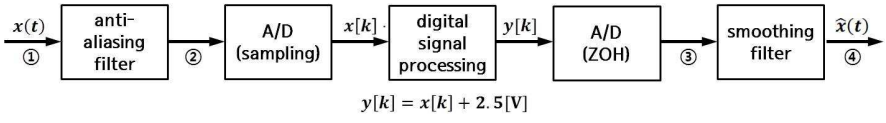

Figure 4.4 Configuration of an experiment to test the frequency aliasing effect

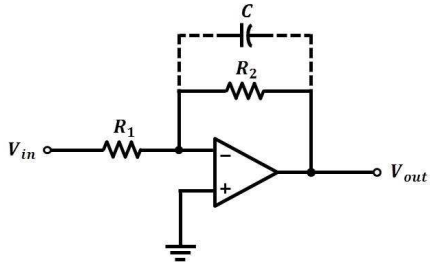

Figure 4.5 Analog filter

2) Using the function generator, generate a signal $x_2(t) = A_2 \cos(2\pi f_2 t)$ whose magnitude is $A_2 = 1[V_{pp}]$ and frequency is $f_2 = 105[Hz]$.

3) Using the adder circuit, construct the signal $x(t) = x_1(t) + x_2(t)$ at node ①. And then, pass it through the anti-aliasing filter.

4) Take out the capacitor from the anti-aliasing filter circuit. That is, the anti-aliasing filter is inactivated in this case. Set the sampling rate as $200[Hz]$ and acquire it from the analog input channel of a DAQ.

5) Add the offset voltage $2.5[V]$ to the acquired signal $x[k]$ and output the digital signal $y[k]$ through the analog output channel of a DAQ.

6) In order to make the resulting stairwise digital signal $y[k]$ smooth, set the cut-off frequency of the smoothing filter as $30[Hz]$

7) Plot the spectra of the signals measured at node ①, ②, ③ and ④. In order to obtain the spectrum of a signal, you can use the FFT function embedded in your oscilloscope. In the spectrum, limit its maximum frequency range to $2f_s$. Observe the frequency component of the signals and analyze your experimental results. What effects do you observe in the waveforms at each node?

8) Connect the capacitor to the anti-aliasing filter circuit. This activates the anti-aliasing filter Set the cut-off frequency of the anti-aliasing filter as $30[Hz]$. Repeat the above experiments from 3)$\sim$5).

Experiment #2.2. *Fast Fourier Transform and DT Spectra*

In this part, we will learn how to use the Matlab embedded FFT function to find the DT spectrum of a discrete-time signal. In addition, we will also make use of the IFFT algorithm in recovering the discrete-time signal from its spectrum.

1) Consider the discrete-time signal

$$x[k] = \begin{cases} -1 & , \ k = 0, 1, 2, 3 \\ 0 & , \ \text{otherwise} \end{cases}$$

and find its DTFT analytically. Recall the DTFT theory summarized in the lecture note.

2) Using the MATLAB function `Xw = fft(x,N)`, find the spectrum of the above signal where $N = 2^2, 2^3, 2^4, 2^5$. Also, utilize the MATLAB function `hatx = ifft(Xw,N)` to recover the original discrete-time signal from the spectrum `Xw`. Plot the magnitude and phase spectrum. As well, plot the original and recovered signals. Briefly comment on the results.

3) Implement the 1024-point FFT algorithm as a C function. Solve the problem in 2) using your own C program. It might be helpful to find and look through

the reference entitled *Numerical Recipe in C* in your university library together with the lecture note. Verify your C program by comparing the result with that obtained by MATLAB embedded function, `fft`.

4) Repeat 1) and 2) for the following discrete-time signal

$$x[k] = \begin{cases} 1 & , \ k = 0, 1, \cdots, 11 \\ 0 & , \ \text{otherwise} \end{cases}$$

5) Consider the signal defined by

$$x[k] = \begin{cases} [\,1 \ \ 2 \ \ 3 \ \ 4 \ \ 5 \ \ 4 \ \ 3 \ \ 2 \ \ 1\,] & , \ k = 0, 1, \cdots, 8 \\ 0 & , \ \text{otherwise} \end{cases}$$

Repeat 1) and 2) with this signal. Comment on the results obtained.

6) Reiterate **Experiments #2.1** using Matlab embedded FFT function and your own FFT function written by C. Calculate the spectrum at each node and compare the results with those of the oscilloscope.

4.4 Digital Filter Design Using Digital Equivalence

Purpose of the Experiment

This experiment is intended to go over the basics of digital equivalence of the continuous-time system and to realize the digital systems using this concept. To this end, we will design a few second-order digital filters and implement them by converting their discrete-time transfer function into the delayed-form difference equation. Though this experiment, we will deduce that the similar procedure known as emulation can be also applied to design a digital controller under the assumption that the sampling rate is fast enough. As well, we will have deeper understandings on the importance of the sampling frequency selection in digital system design.

Design Methodologies of a Digital Controller

From the existing open literature, we see that there are three different ways to design a digital controller. The first technique is called emulation design used by industries in general. In this approach, the controller design is done in the continuous-time domain followed by discretization to produce a discrete-time controller for digital implementation. Direct discrete-time design is the second technique, where a discrete-time controller is designed in discrete-time domain directly using an approximate discrete-time model of the given plant and using the pole-placement technique. This approach shows its potential in improving the overall performance of the sampled-data control systems. The third technique is called sampled-data design which tries to take into account the inter-sample behavior in the design to improve even the detailed control performance, but it is sill recognized as an open problem.

The use of the emulation technique was initiated by treating discrete-time systems in a continuous-time framework. As mentioned earlier, emulation is regarded as the simplest method of controller design for sampled-data systems, and is of inferior to the other two methods in terms of stability and performance. Nevertheless, it merits are emphasized in practice because the performance degradation due to the sampling becomes ignorable nowadays because of the recent developments in digital hardwares.

Emulation is very simple because it makes use of the control system design and analysis tools formulated in continuous-time domain. It consists of a threestep design procedure; continuous-time design, controller discretization, and digital implementation. The detailed procedure is described below.

Firstly, a continuous-time controller design is carried out. The obtained continuous-time controller achieves a set of performance and robustness criteria for the closed-loop continuous-time system. At this stage, the sampling is completely ignored. The second step is to discretize the continuous-time controller. Discretization is in fact a type of approximation. In general, it requires a sufficiently fast sampling because the discrete-time model becomes a good approximation of the continuous-time model typically only for small sampling periods. Discrete-time representation of the controller is usually obtained using some numerical integration methods or using pole-zero matching from continuous-time to discrete-time domain. For numerical integration, Euler and Tustin methods are commonly used, while in the matched pole-zero method the extrapolation from the relationship between the s-plane and the z-plane is investigated. Finally, the controller is implemented in digital domain. This step assumes that a sampler and hold device is placed between the plant and the controller. For its standing assumption, the most important issue in the implementation step of an emulation design is the selection of the sampling period. It is necessary to determine the sampling period at this stage, so that the property satisfied by the continuous-time system can be preserved.

In this technical aspect, it is common that the sampling frequency ω_s is chosen 20$\sim$30 times larger than the bandwidth ω_{BW} of the continuous-time control system. Provided that the continuous-time system does not have in-dominant poles, the sampling frequency must be $\omega_s \geq 2\omega_{BW}$ to meet the Nyquist sampling theorem. However, even for this case, many engineers try to set $\omega_s \geq 5\omega_{BW}$ to make the digital system close to the original analog system.

Experiment #3.1. *Digital Equivalence of an Analog Filter*

Let us design a digital filter using its digital equivalence which can be found by the Matlab function `butter`.

1) Design analog filters with parameters summarized in Table 4.1. Referring to the following MATLAB script, you can readily design the filters.

```
[CT_LPF_NUM, CT_LPF_DEN] = butter(N,Wc,'low','s');
Hs = tf(CT_LPF_NUM, CT_LPF_DEN); % CT transfer function
```

```
% digital equivalence using Tustins method
% generally used for bandpass or notch filter

Hz = c2d(Hs,Ts,'tustin'); % DT transfer function
```

or

```
% digital equivalence using frequency prewarping
% often used for bandpass or notch filter

opt=c2dOptions('Method','tustin','PrewarpFrequency',Wc);

Hz = c2d(Hs,Ts,opt); % DT transfer function
```

2) Draw the bode plot of each filter using its CT and DT transfer functions. In addition, check the step responses of the analog and digital filters using SIMULINK. Note that you should select the fixed-step Runge-Kutta method with step size $T_{sim} = 1[ms]$ in SIMULINK for simulation. If necessary, use appropriate blocks in SIMULINK library such as `zero-order hold`, `sample and hold`, `up/down sample` for interfacing the digital filter with analog signals.

3) Construct an input signal generation block in SIMULINK.

$$r(t) = s(t) + n(t),$$

where $s(t) = \sin(2\pi f_{in} t)$ is the signal and $n(t)$ is the zero-mean band-limited white noise. The frequency of the input signal is set as $f_{in} = 2[Hz]$. The noise power in band-limited white noise model is set as $\sigma^2 T_{sim}$, where the standard deviation of the noise is $\sigma = 0.2$.

4) Insert the above input signal into the analog and digital filters designed in the above. Save the input/output data using `From Workspace` and `To Workspace` blocks. Plot and analyze your experimental results.

5) Repeat 2)∼4) for $f_{in} = 35[Hz], 100[Hz]$.

Discussion and Analysis

What is the desirable value of sampling rate compared to the system bandwidth? Discuss your conclusion based on the above experimental results.

Table 4.1 Filter design parameters

filter	critical frequency	value
lowpass filter	ω_c^{LPF}	case 1) $20[Hz]$, case 2) $70[Hz]$
highpass filter	ω_c^{HPF}	case 1) $20[Hz]$, case 2) $70[Hz]$
bandpass filter	ω_{c1}^{BPF}, ω_{c2}^{BPF}	$30[Hz]$, $40[Hz]$
bandstop filter	ω_{c1}^{BSF}, ω_{c2}^{BSF}	$30[Hz]$, $40[Hz]$

Experiment #3.2. *Implementation of Digital Filters Using Matlab and C*

The output of a digital filter is recursively calculated by using the difference equation corresponding to the given DT transfer function. Now, let us write a Matlab script to implement a digital filter and then port it on the digital computer as the form of a C function.

1) Write the phase-variable form for the DT transfer function of a digital lowpass filter derived in **Experiment #3.1**.

2) Define the state variables $\mathbf{X}[k] = [\, x_1[k] \;\; x_2[k] \,]^T$ required for implementing the transfer function. Then, derive the difference equation for the DT transfer function.

3) Make a Matlab function to recursively calculate the output $y[k]$ to the input $r[k]$.

$$y[k] = \texttt{LowPassFilter}\,(\mathbf{X}[k], \mathbf{X}[k-1], r[k])$$

4) Apply the input signals defined in the above experiment for your own Matlab function and obtain the result. Compare the result obtained by using the Matlab script with the SIMULINK result in **Experiment #3.1**.

5) Implement the same filter using C. Test the function and compare the result with that of 4).

6) Repeat 1)$\sim$5) for a digital bandpass filter designed in the previous.

4.5 Recursive Least Squares Approach to Parametric Identification of a Digital System

Purpose of the Experiment

In this experiment, the recursive least squares estimation theory will be introduced to model the direct current (DC) motor attached to a gimbal assembly. Noticing that the transfer function requires the standing assumption on the LTI property of the given system, we first linearize the input-output characteristics of a motor by incorporating an extra routine in your real-time control software. Since the DT transfer function can be converted to the LTI difference equation with unknown coefficients, this motivates us to formulate the motor identification simply as the estimation problem of these coefficients using time responses of your gimbal motor measured by a rate gyro. This approach is somewhat different from the previous scheme because it directly makes use of not the frequency response but the time response. Moreover, it provides us the DT transfer function, hence the digital conversion of a CT transfer function is not necessary.

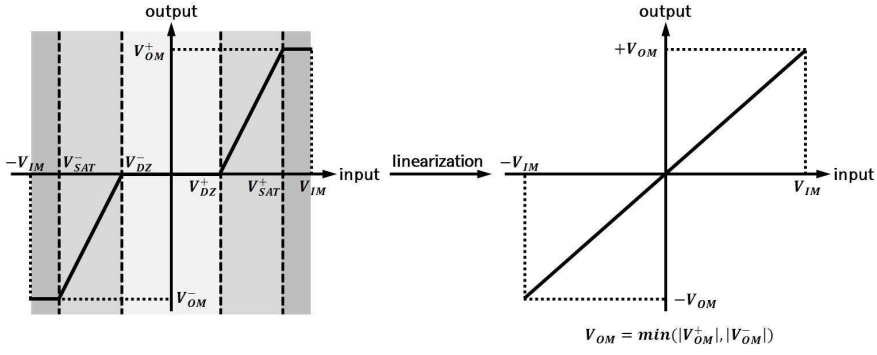

Figure 4.6 Linearization of input-output characteristics

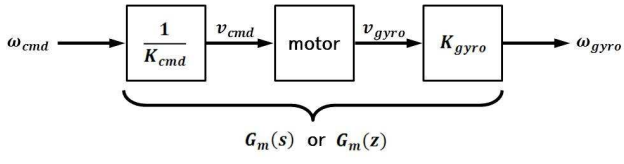

Figure 4.7 Motor transfer function including unit conversion factors

DC Motor Linearization

Every physical system retains nonlinearities and this is the same for the DC motor used in this experiment. Provided that there is no slew-rate limit, the conventional input-output characteristics of a DC motor is described in Figure 4.6. The DC motor operation region consists of the dead-zone, linear, and saturation regions. The ultimate goal of the linearization is to find an appropriate mapping function to operate the given motor in its linear region.

$$V_{ss} = f(V_{cmd}),$$

where the voltage command V_{cmd} generated from the control algorithm has the value within $-V_{IM} < V_{cmd} < V_{IM}$. On the contrary, the voltage V_{ss} issued to the motor is within $V_{SAT}^{-} < V_{ss} < V_{DZ}^{-}$ or $V_{DZ}^{+} < V_{ss} < V_{SAT}^{+}$.
To design The linearizing function requires the dead-zone voltage V_{DZ} and the saturation voltage V_{SAT} as well. However, practically speaking, the dead-zone could not be perfectly eliminated even after linearization. This is because, if the dead-zone is removed excessively, it results in more complicate nonlinear behavior of a DC motor around the neutral point $V_{cmd} = 0$.

As is well-known, a DC motor can be described as a second-order system considering its eletro-mechanical dynamics. Assuming that a motor driver and a rate gyro are much faster than the DC motor, its transfer function can be modeled by exploiting its response to the command V_{cmd} using the gyro measurement V_{gyro}. For

designing a velocity servo loop using a DC motor, as shown in Figure 4.7, it is more convenient to model the motor transfer function from the angular velocity command ω_{cmd} to the angular velocity response ω_{gyro}. To do this, the unit conversion factors from $[V]$ to $[deg/s]$, K_{cmd} and K_{gyro}, should be found in advance. This can be done by experiments and/or by theoretical analysis.

Experiment #4.1. *Linearization of a Gimbal Motor*

1) Write your C program to insert the following voltage command into the motor driver and measure the gyro output using DAQ.

$$V_{cmd} = \begin{cases} 0 & , \quad 0(sec) \le t < 1(sec) \\ A & , \quad 1(sec) \le t < 2(sec) \\ 0 & , \quad 2(sec) \le t < 3(sec) \\ -A & , \quad 3(sec) \le t < 4(sec) \\ 0 & , \quad 4(sec) \le t < 5(sec) \end{cases}$$

   Repeat the experiment from $A = 0[V]$ to $A = 5[V]$ with the interval of $0.1[V]$.

2) Calculate the DC value of gyro output by taking the average at steady-state.

3) Since $V_{cmd} = V_{ss}$ in this case, you can draw the $V_{ss} - V_{gyro}$ curve using experimental results. Analyze the operational regions of a DC motor in CW and CCW directions, respectively.

4) Design a function, $V_{ss} = f(V_{cmd})$ so the gimbal motor will behave linearly for almost entire range of the voltage command, $-5[V] < V_{cmd} < 5[V]$.

5) Implement the function for motor linearization in your C program and repeat 1)∼3). Observe the change in $V_{cmd} - V_{gyro}$ curve. Does the linearization work as expected?

6) Confirm the unit conversion factor K_{gyro} specified in the datasheet provided by the rate gyro maker. Obtain the steady-state value of angular velocity to the constant voltage command input using DAQ. Draw the $V_{cmd} - \omega_{gyro}$ curve for $-5[V] < V_{cmd} < 5[V]$ with the interval of $0.5[V]$.

7) Figure out the remaining conversion factor K_{cmd} by analyzing the curve obtained in 8). If necessary, use the basic data fitting function in the menu of Matlab figure (`Figure >> Tools >> Basic Fitting`).

8) Reflect the unit conversion factors in your program. And then, find the $\omega_{cmd} - \omega_{gyro}$ curve.

9) Observe the transient responses. Is there a slew-rate limit, $max(|\dot{\omega}_{gyro}|)$?

Recursive Least Squares Estimation

Let us assume that the DC motor dynamics is described by the following discrete-time transfer function.

$$G_m(z) = \frac{W(z)}{R(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

As stated above, the ultimate goal of parametric identification is to estimate the unknown coefficients, a_i and b_i, used in $G_m(z)$. Using the unit delay operator z^{-1}, the input-output relation is expressed as the form of difference equation.

$$w_j + a_1 w_{j-1} + a_2 w_{j-2} = b_0 r_j + b_1 r_{j-1} + b_2 r_{j-2}$$

Obviously, it can be rewritten as follows:

$$y_j = h_j \mathbf{x},$$

where

$$y_j \triangleq w_{j+2},\ \ h_j \triangleq \begin{bmatrix} r_{j+2} & r_{j+1} & r_j & -w_{j+1} & -w_j \end{bmatrix},\ \ \mathbf{x} \triangleq \begin{bmatrix} b_0 & b_1 & b_2 & a_1 & a_2 \end{bmatrix}^T.$$

Considering the measurement noise contaminated in the gyro output, the following approximate linear regression model[1] is used for system identification.

$$y_j = h_j x + e_j,$$

where e_j is introduced to reflect the measurement error and is assumed to be normally distributed zero-mean white noise with variance $R_j = R$, namely $e_j \sim \mathcal{N}(0, R_j)$. Accumulating the above relation from $j = 0$ to $j = k$ gives us the vector-valued linear regression model.

$$y^k = H^k \mathbf{x} + e^k,$$

where $e^k \sim \mathcal{N}(0^{k\times 1}, R^k)$ and it has been defined that

$$y^k \triangleq \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_k \end{bmatrix},\ \ H^k \triangleq \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_k \end{bmatrix},\ \ e^k \triangleq \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_N \end{bmatrix},\ \ R^k \triangleq \begin{bmatrix} R_0 & 0 & \cdots & 0 \\ 0 & R_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_k \end{bmatrix} = R \cdot I^{k\times k}.$$

[1]This approximation is not mathematically rigorous. To tackle this issue, more complicated measurement model and estimation theory, so-called non-conservative robust Kalman filtering, should be applied for the system identification. Fortunately, if the gyro measurement noise variance is small enough, the system identification result could be acceptable in spite of using this approximate model and the standard least squares estimator.

The least squares estimate is a unique minimizing solution to the following quadratic cost function. In other words, the least squares estimate $\hat{\mathbf{x}}_k$ minimizes the effect(energy) of the measurement error.

$$J_k = \left(y^k - H^k\mathbf{x}\right)^T (R^k)^{-1} \left(y^k - H^k\mathbf{x}\right) = (e^k)^T (R^k)^{-1} e^k = \sum_{j=0}^{N} \frac{1}{R} e_j^2$$

The condition for a unique minimum of the quadratic function is summarized as follows:

$$\frac{\partial J_k}{\partial \mathbf{x}} = -2(H^k)^T (R^k)^{-1}(y^k - H^k\mathbf{x}) = 0, \;\; \frac{\partial^2 J}{\partial \mathbf{x}^2} = 2(H^k)^T (R^k)^{-1} H^k > 0$$

Therefore, from the above condition, the batch form of the least squares estimate $\hat{\mathbf{x}}_k$ is defined as

$$\hat{\mathbf{x}}_k = P_k (H^k)^T (R^k)^{-1} y^k, \;\; P_k^{-1} = (H^k)^T (R^k)^{-1} H^k$$

Note that, to compute this batch solution, the required memory increases exponentially because the block matrices H^k, R^k and the vector y^k are getting bigger as time goes.

In order to solve this problem, the recursive form of a least squares estimate is used in practice.

$$P_k^{-1} = P_{k-1}^{-1} + H_k^T R^{-1} H_k,$$
$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + K_{f,k}(y_k - H_k \hat{\mathbf{x}}_{k-1}),$$

where the Kalman gain $K_{f,k}$ is defined as $K_{f,k} \triangleq P_k H_k^T R_k^{-1}$. Meanwhile, the physical meaning of the estimation error covariance matrix P_k is understood as the uncertainty contained in the estimate $\hat{\mathbf{x}}_k$.

Using this recursive formulae, if the current measurement y_k is acquired, we can recursively update the new values of the estimate $\hat{\mathbf{x}}_k$ and the (error covariance) matrix P_k from the previous data $\hat{\mathbf{x}}_{k-1}$ and P_{k-1}. This means that the recursive least squares estimator is computationally efficient compared to the batch solution. Hence, it is more suitable for real-time implementation on the low-cost digital computer with limited memory capacity.

Experiment #4.2. *Motor Modeling Using Least Squares Estimator*

1) Write your C program to insert the sinusoidal command into the motor driver and measure the gyro output using DAQ.

$$\omega_{cmd} = A \sin(2\pi f_{in} t),$$

where the magnitude A is chosen by considering the operating condition of a gimballed seeker. It should not be too small or too large. Also, it satisfies the condition related to the slew-rate limit.

$$A < \frac{1}{G_m(z=1)} \cdot \frac{\texttt{SLEW RATE LIMIT}}{2\pi \max(f_{in})}$$

2) Check whether there exists f_{in} which causes 90° phase shift of ω_{gyro} with respect to ω_{cmd}. Determine the order of a motor and roughly estimate the bandwidth of $\bar{\omega}_m$.

3) To obtain the frequency response, select the equally spaced frequencies between $\frac{1}{10}\bar{\omega}_m \sim \frac{1}{3}\bar{\omega}_m$. The total number of test frequencies are 10. Issue the sinusoidal command and collect the frequency response for each test frequency. Pay attention if the responses have the same number of samples.

4) Estimate the motor transfer function $G_m(z)$ by applying the Matlab function `invfreqz` for the frequency responses.

5) Write a Matlab script to implement the batch and the recursive least squares estimator, respectively.

6) Using your Matlab functions, estimate the unknown coefficients contained in the DT transfer function $G_m(z)$. The recursive solution gives the same estimate with the batch one?

7) Validate the motor model obtained in 4) and 6). Which one is more accurate? Calculate the important parameters regarding motor dynamics, the DC gain and the time constant if $G_m(z)$ is a 1^{st} order system or the natural frequency and damping ratio if $G_m(z)$ is a 2^{nd} order system.

Discussion and Analysis

1) Show the detailed procedure to derive the recursive form of a least squares estimator.

2) In order to implement the recursive least squares estimator, it is very important to determine the initial guess $\mathbf{x}_0$, the initial estimation error covariance matrix P_0 and the variance of a measurement error R. Explain your idea to determine these values in the above experiment.

3) Discuss about the effects of P_0 and R on the convergence property of a least squares estimate $\hat{\mathbf{x}}_k$.