INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Université d'Ottawa • University of Ottawa

Reduction of the Quantization Error in Fuzzy Logic Controllers by Dithering

by
Jie Mao B.Sc.

A thesis submitted to the School of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Applied Science
in Electrical Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa
November 1998

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

0-612-36717-7

Canada

Abstract

Dithering technique has been used to reduce the quantization error for more than thirty years. This thesis reviews the theories of quantization and dithering, which explain the effect of dithering from a theoretical point of view. Applying dithering to digital Fuzzy Logic Controller (FLC) is the focus of this thesis. Two applications, FLC for Inverted Pendulum and FLC for Truck Backer-Upper, have been simulated and reported. The results of the simulation demonstrate the positive effect of input signal dithering on the resolution of the digital FLC output, especially for the case of the coarse quantization of the input signal.

Table of Contents

List of Figures

List of Terms

2-D	Two Dimensional
3-D	Three Dimensional
ADC	Analog to Digital Converter
CF	Characteristic Function
COG	Center of Gravity
EAI	External Authoring Interface
FAM	Fuzzy Associative Memories
FL	Fuzzy Logic
FLC	Fuzzy Logic Control
FT	Fourier Transform
GUI	Graphical User Interface
HTML	Hypertext Markup Language
I/O	Input/Output
MIQ	Machine Intelligence Quotient
PDF	Probability Density Function
QT	Quantizing Theorem
VRML	Virtual Reality Modeling Language

Acknowledgements

I would like to express my sincere gratitude to Dr. Emil Petriu, my academic supervisor, for his guidance, advice and continuous encouragement during the period of study. His knowledge and experience have been of tremendous assistance in the research and preparation of this thesis.

Secondly, I am grateful to many of my colleagues for the invaluable discussions during the Master program.

Finally, I would like to thank *Jian*, my husband, for his understanding, encouragement always, for his support and patience. My deepest gratitude is reserved for my parents, for their love and care throughout all stages of my study.

Chapter 1 Introduction

Signals encountered in real life are often in continuous time, that is, they are waveforms (or functions) on the real line. Their amplitude is usually continuous as well, meaning that it can take any real value in a certain range. Signals continuous in time and amplitude are called analog signals. The basic promise of analog systems comprises functional simplicity, high speed and overall low cost, as well as the ability to mimic natural phenomena with electrical variables and parameters. Analog signal processing inevitably has some limitations and drawbacks, such as accuracy limitations, limited repeatability, sensitivity to electrical noise, limited dynamic range of voltages and currents, lack of flexibility to specification changes in the processing functions, difficulty in implementing nonlinear and time-varying operations, and high cost and accuracy limitations of storage and retrieval of analog information.

Therefore in recent years, there has been a clear trend in the use of digital techniques for performing varied signal processing functions. Now we can find digital techniques almost in every area in our life, telecommunication, personal computer, cellular phone, etc. This thesis will address the special scope of the digital signal processing for fuzzy logic control.

Digital signals are both time and amplitude quantized. A digital signal may always be represented by a sequence of numbers in which each number has a finite

number of digits. To convert the analog signals to digital at the specified sampling intervals requires an analog-to-digital converter (ADC or A/D). It accepts an analog signal at its input and provides an n-bit binary signal at its output. The important characteristics of an ADC are the speed and accuracy with which it can perform a conversion. The former will place a limit on the number of conversions that will occur per second, limiting the upper value of analog signal frequency that can be retrieved from the sampled data and, in turn, the upper-frequency component of the analog signal being converted to digital. The latter (also the primary concern of an ADC) will affect the overall fidelity of the process, introducing extraneous components (or quantization error) into the output signal.

Since quantization error is inevitable, we need find some solutions to reduce the effect of the quantization error. The most straightforward way is to use the high-resolution quantizer, which means long conversion time (this is inappropriate for strict real-time system.) and expensive cost in money. The other way is to use the dithering technique. Dithering [35], [38] has been used in the past thirty years to reduce the effects of quantization error. In this technique, a signal called a dither is added to an input signal prior to quantization. In particular, suitably chosen random dither signals can cause the quantization error to be signal independent, uniformly distributed noise.

Chapter 2, "Quantization and Dithering", presents a detailed discussion on quantization and dithering theory.

Fuzzy sets were introduced in 1965 by Lotfi Zadeh [52], who is considered to be the father of Fuzzy Logic, as a new way to represent vagueness in everyday life. They are a generalization of conventional set theory, one of the basic structures underlying computational mathematics and models. During the past few years, we have witnessed a rapid growth in the number and variety of applications of FL, ranging from consumer electronics and industrial process control to decision support systems and financial trading.

The reason that the applications of FL have developed so rapidly, perhaps, is for its convenience. FL specification can be expressed very simply using natural language. The basis for FL is the basis for human communication and it is conceptually easy to understand.

Chapter 3, "Fuzzy Logic Control" , presents a detailed introduction to the theory and terminology of fuzzy disciplines, including fuzzy sets, fuzzy rules, fuzzy reasoning, and fuzzy inference systems.

The specific contribution made in this thesis is in the area of digital FLC application. Based on the theoretical studies on dithering technique and fuzzy logic control, this thesis is among the first to study the effects of dithering on digital FLC performance. Two applications using digital FLC with dithering technique are thoroughly studied and the results are validated using the MATLAB/Simulink. One application is the FLC for an inverted pendulum, and the other is the FLC for a truck backer-upper. The simulation results presented in Chapter 4, "Dithering for Fuzzy Logic Control --- Applications" have shown the improvement made by

adoption of dithering technique in digital FLC. Moreover, a virtual prototyping environment for the second application, fuzzy truck backer-upper, has been developed.

Chapter 2 Quantization and Dithering

Quantization occurs whenever physical quantities are represented numerically. As a mathematical operator, a quantizer may be defined as processing continuous signals to give a stepwise continuous output or as processing sampled signals to give a stepwise sampled output [54]. Our attention will be focused on the simple but important quantizer - uniform quantizer. It is apparent that the smaller the quantization step, the larger will be the numbers required to represent the same physical quantities and the greater will be the difficulty and expense in storing and processing these numbers. Often, a tradeoff has to be struck between accuracy and economy.

Dithering is a technique to reduce the effect of quantization error [38], [52] and [53]. From a statistical point of view, a proper dither signal added to the input signal before quantization can insure that the quantization theorems can be satisfied to a good approximation. This will lead to a predictable quantization error and a minimum loss of statistical data of the input signal.

In this chapter, we will review the theory of quantization, from which we can show the reason and importance for dithering. Also, the theoretical property of dither signals will be included. In addition, various dither forms will be introduced.

2.1 Theory of Quantization

Signals are represented in digital computers by sequences of finite bit-length numbers. Mapping of a continuous-time, continuous-amplitude signal to a sequence of computer numbers requires discretization both in time and amplitude: *sampling* and *quantization* must be performed. Sampling theory is well understood. It is a linear operation; therefore linear system theory can be applied to the analysis of it. In this section, we will focus on the concepts and theories of quantization. In addition, quantization error and how to reduce it are to be discussed.

2.1.1 Quantization and Quantizer

Sampling discretizes time, and quantization discretizes amplitude. They are two corner stones of the modern digital systems. But quantization is generally less well understood than sampling. The reason is that quantization is a nonlinear operation; therefore most people believe that standard tools of linear system theory cannot be applied to it. In fact, we will find out how linear system theory can be precisely used to analyze the effect of quantization on moments and other statistical properties of the signals.

The basic quantizer operation is depicted in the block diagram in Fig. 2.1.

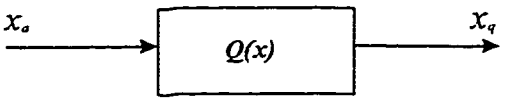


Figure 2.1 Quantizer symbol

In this thesis, only the uniform ideal quantizer with rounding quantization is considered. However, any other uniform quantization scheme can be studied with small changes in presented analysis [6]. A uniform quantizer maps an analog input into one of a collection of equally spaced output levels. The difference between two adjacent levels, Δ, is called *quantization step,* or *quantization interval.*

$$\Delta = \frac{x_{\max} - x_{\min}}{2^n} = \frac{x_{\max} - x_{\min}}{m} \tag{2.1}$$

x_{max}, x_{min} denote the maximum and minimum value of the input signal, respectively. n is the quantization resolution and m is the number of quantization output levels.

When the input is confined in the no-overload region $[-(m+1)\Delta/2, (m-1)\Delta/2]$, the analytical expression for quantizer input / output and error characteristics are, respectively

$$x_q = Q(x) = \left\lfloor \frac{x + \frac{\Delta}{2}}{\Delta} \right\rfloor \Delta \tag{2.2}$$

and

$$q(x) = Q(x) - x = \frac{\Delta}{2} - \Delta \left\langle \frac{x + \frac{\Delta}{2}}{\Delta} \right\rangle \tag{2.3}$$

where $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x, and $\langle x \rangle = x - \lfloor x \rfloor$ is the fractional part of x. Apparently, when the input is in the no-overload region, the maximum quantization error $q(x)$ is $\frac{\Delta}{2}$.

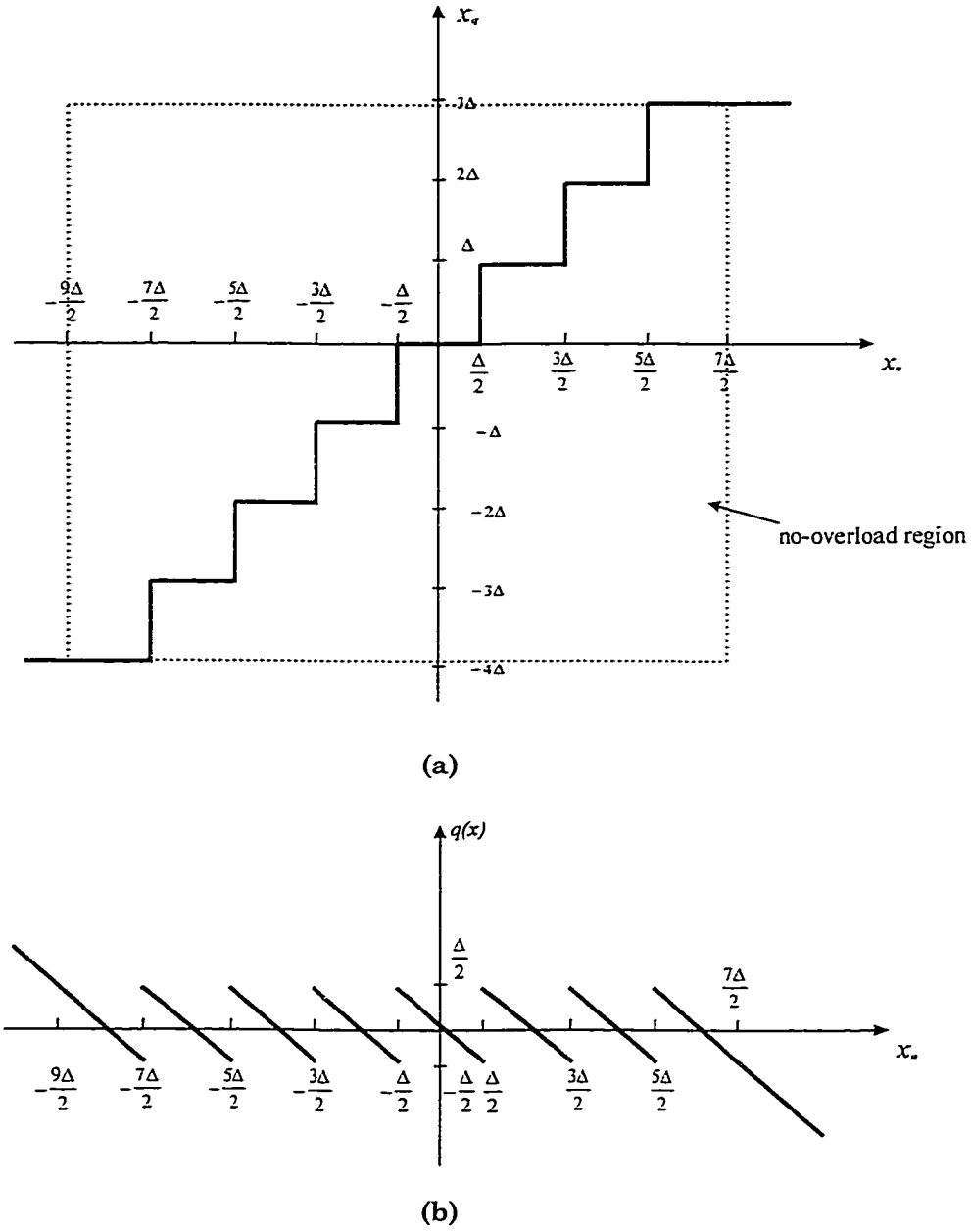


Figure 2.2 Uniform quantizer (adapted from [6])

(a) quantization characteristics (b) quantization error

Fig. 2.2(a) shows the input/output characteristic of a typical uniform 8-level quantizer and Fig. 2.2(b) shows the quantizer error characteristic *e*, corresponding to (a). x_a is the analog input and x_q is the quantizer output. For simplicity, we will use x to denote the analog input instead of x_a, in the rest of this thesis.

Now let us look at a simple example. Assume that we have *y=cos(x)* as the original input signal. We quantize this signal with different resolutions, 2 bit and 6bit.

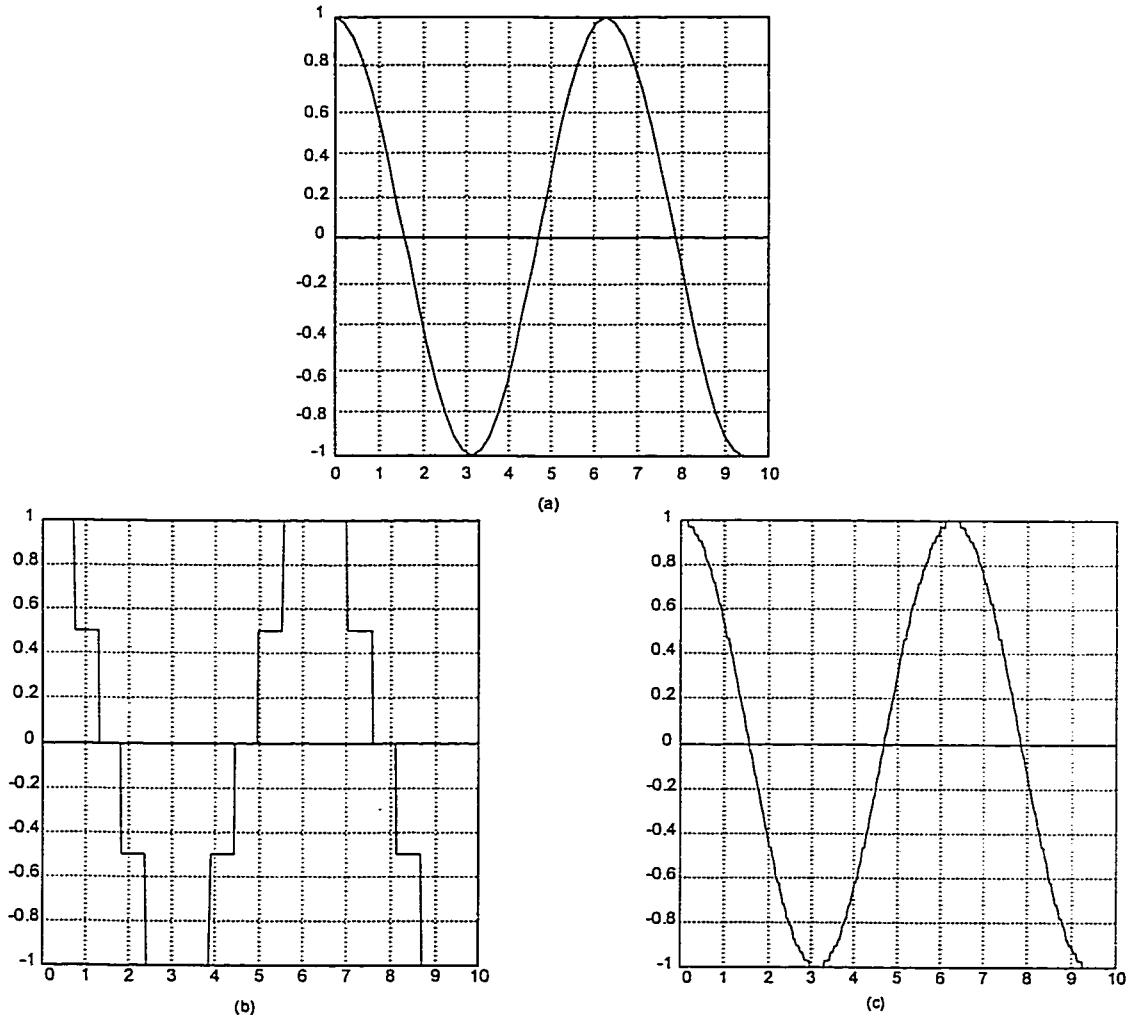


Figure 2.3 An example of quantization

(a) original input signal *y=cos(x)*; (b) 2-bit quantization of the signal;

(c) 6-bit quantization of the signal

The results are shown in Fig. 2.3(b) and (c). Apparently, with higher resolution, the waveform in (c) is much smoother and closer to the original signal in (a). The reason is that the higher the resolution, the more quantization levels the quantizer has and the finer the measurement resolution is.

Deceptively simple in its description and construction, the uniform quantizer has proved to be surprisingly difficult to analyze precisely, because of its inherent nonlinearity. A common analysis technique is to linearize the quantizer by assuming that the quantization error q, the difference between the quantizer input and output, consists of a sequence of uniformly distributed random variables that are uncorrelated with each other and with the input signal (a discrete time stationary random process). In this case, the quantizer is replaced by the simple system of Fig. 2.4, where X_n is the input and q_n is an independent and identically distributed (i.i.d.) sequence of uniformly distributed random variables. $n \in Z$, where Z is the set of all integers. The quantizer output is the reproduction process $\hat{X}_n = Q(X_n)$.

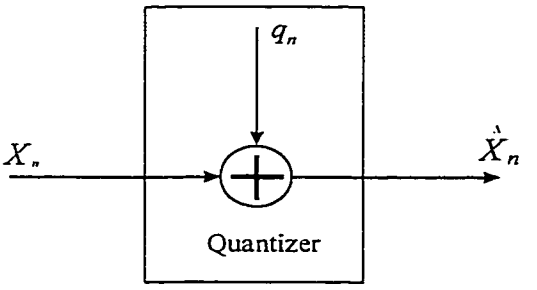


Figure 2.4 Additive noise model of a quantizer

This approximation was shown by Bennett [3] to be reasonable when the number of quantizer levels n is large, the spacing Δ between the levels is small, and the input probability density function is f_x smooth. It has long been known that this

approximation can be quite poor in certain cases, e.g., for sinusoidal inputs. Furthermore, many modern oversampled analog-to-digital converters clearly violate the Bennett conditions in that they have few levels, relatively large spacing Δ, and non-smooth quantizer input density functions. One common means of attempting to force the Bennett approximations to hold is to use a *dither* signal, which is the focus of this thesis and will be discussed in detail in the second part of this chapter. But before we go to it, let us go one step further to the theory of quantization. We will investigate it and find out the reason for *dither* from a statistical point of view.

2.1.2 Statistical Theory of Quantization

Sampling discretizes time, and quantization discretizes amplitude. One would expect that quantization has a similar effect on functions of the amplitude as sampling has on functions of time. This recognition led Widrow [51]-[53] to the development of a statistical theory of quantization.

The sampled signal can be obtained from the continuous-time signal by multiplying it (modulation) with a certain impulse carrier. The spectrum of the original signal is repeated along the frequency axis. When the repeated spectra do not overlap, the original spectrum can be restored, and its inverse Fourier Transform yields the original input signal. In other words, the sampled signal contains the same information as the continuous-time signal, and it can be used for the same purposes as the original signal. This statement in a precise mathematical form is the *sampling theorem* [21], [28] and [40].

Now, let us study the probability density functions (PDF's) of the quantizer input/output.

Figure 2.5 shows a typical PDF of the quantizer input and output. The input PDF $f_x(x)$ is smooth, and the output PDF $f_x^q(x)$ is discrete. The reason for this is that each input value is rounded toward the nearest allowable discrete level. The probability of each discrete output level equals the probability of the input signal occurring within the associated quantum band. For example, the probability that the output signal has the value Δ equals the probability that the input signal falls into $(\frac{\Delta}{2}, \frac{3\Delta}{2}]$.

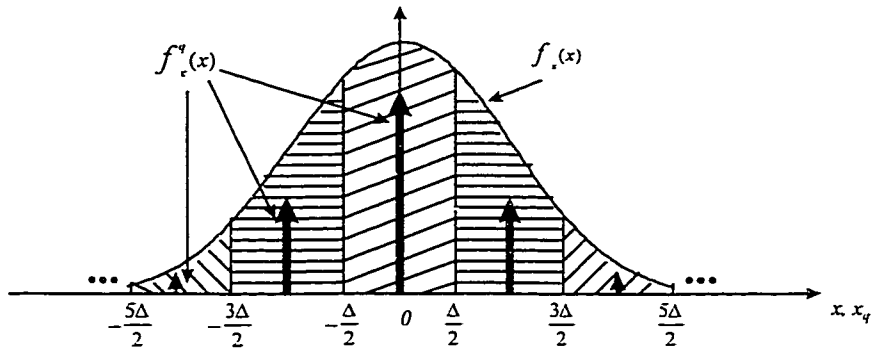


Figure 2.5 Formation of the PDF of the quantizer output x_q (adapted from [53])

Careful study of quantization reveals that the PDF's of the input and output signals are related to each other through a special type of sampling. The output PDF is a string of Dirac delta functions whose areas correspond to the areas under the input PDF within the bounds of each quantum box. Cutting up the input PDF into strips as in Fig. 2.5, the area of each strip is compressed into an impulse in the center of the strip when forming the output PDF. This is like a sampling process, and it is called *area sampling*.

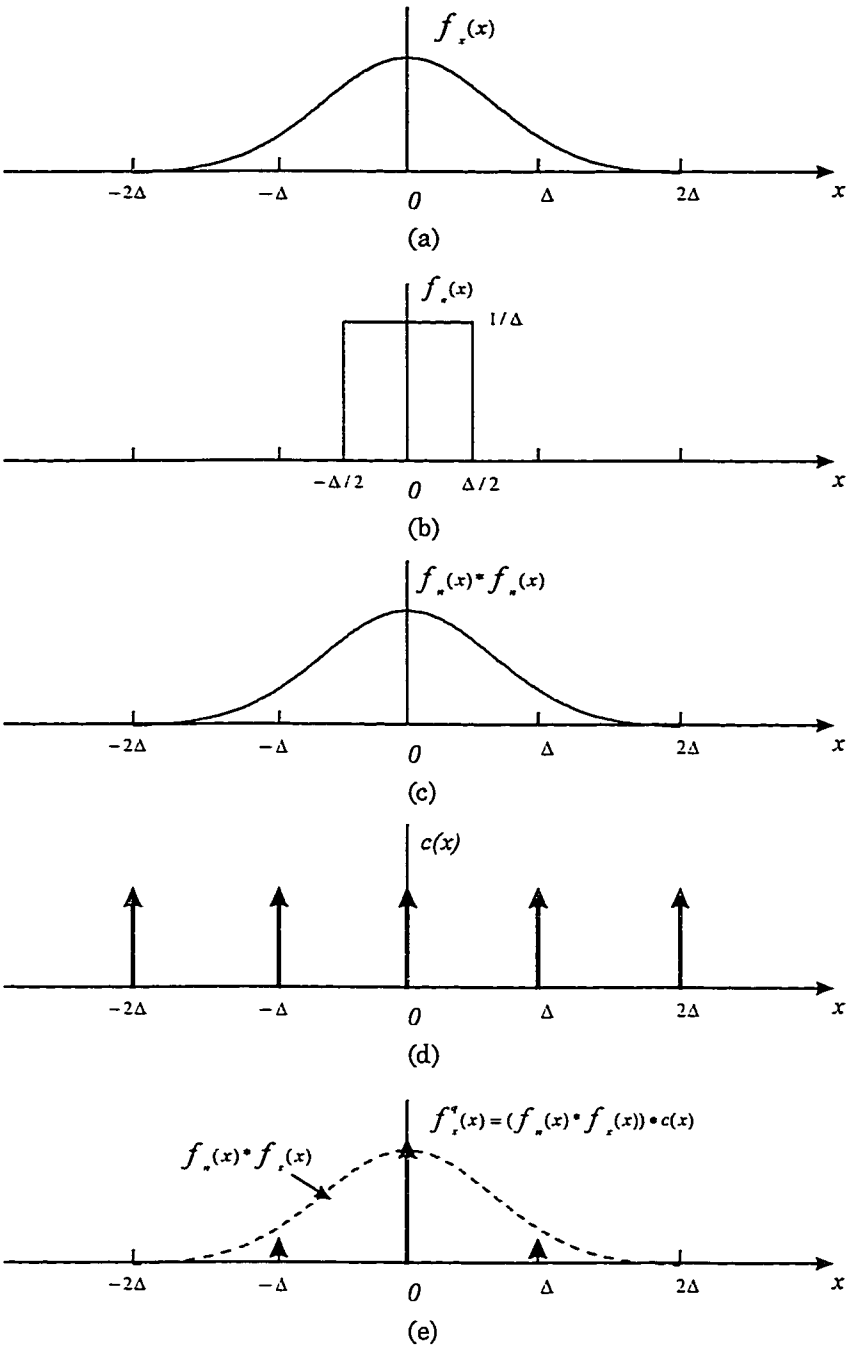


Figure 2.6 Derivation of PDF of x_q from area sampling of the PDF of x

(a) PDF of x; *(b)* rectangular pulse function; *(c)* convolution of *(a)* and *(b)*; (c) the impulse train; and *(e)* PDF of x_q, the product of *(c)* and *(d)*

(adapted from [53])

Area sampling can be accomplished by first convoluting the input PDF $f_x(x)$ with a uniform pulse

$$f_n(x) = \begin{cases} \dfrac{1}{\Delta}, & -\dfrac{\Delta}{2} < x < \dfrac{\Delta}{2} \\ 0 & \textit{elsewhere}, \end{cases} \qquad (2.4)$$

then following this with conventional sampling. See Fig. 2.6.

The input PDF $f_x(x)$ is shown in Fig. 2.6(a). The function $f_n(x)$ is shown in Fig. 2.6(b). The convolution of $f_x(x)$ and $f_n(x)$ is illustrated in Fig. 2.6(c). To do conventional sampling, a uniform impulse train is represented in Fig. 2.6(d). Multiplying this train by the convolution gives the impulse train of Fig. 2.6(e). This final impulse train is the PDF of the quantizer output, $f_x^q(x)$. Every step of the way in going from $f_x(x)$ to $f_x^q(x)$ involves linear operations.

The Fourier transform (FT) of the PDF is known in statistics as the characteristic function (CF). The definition of FT is given in (2.5),

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-j\omega x} dx \qquad (2.5)$$

so the input CF is

$$\Phi_x(u) = \int_{-\infty}^{\infty} f_x(x) e^{-jux} dx = E\{e^{-jux}\} \qquad (2.6)$$

The CF is as useful in quantization theory as the Fourier transform of signals is in sampling theory. Therefore, it makes sense to see the area sampling in the CF domain.

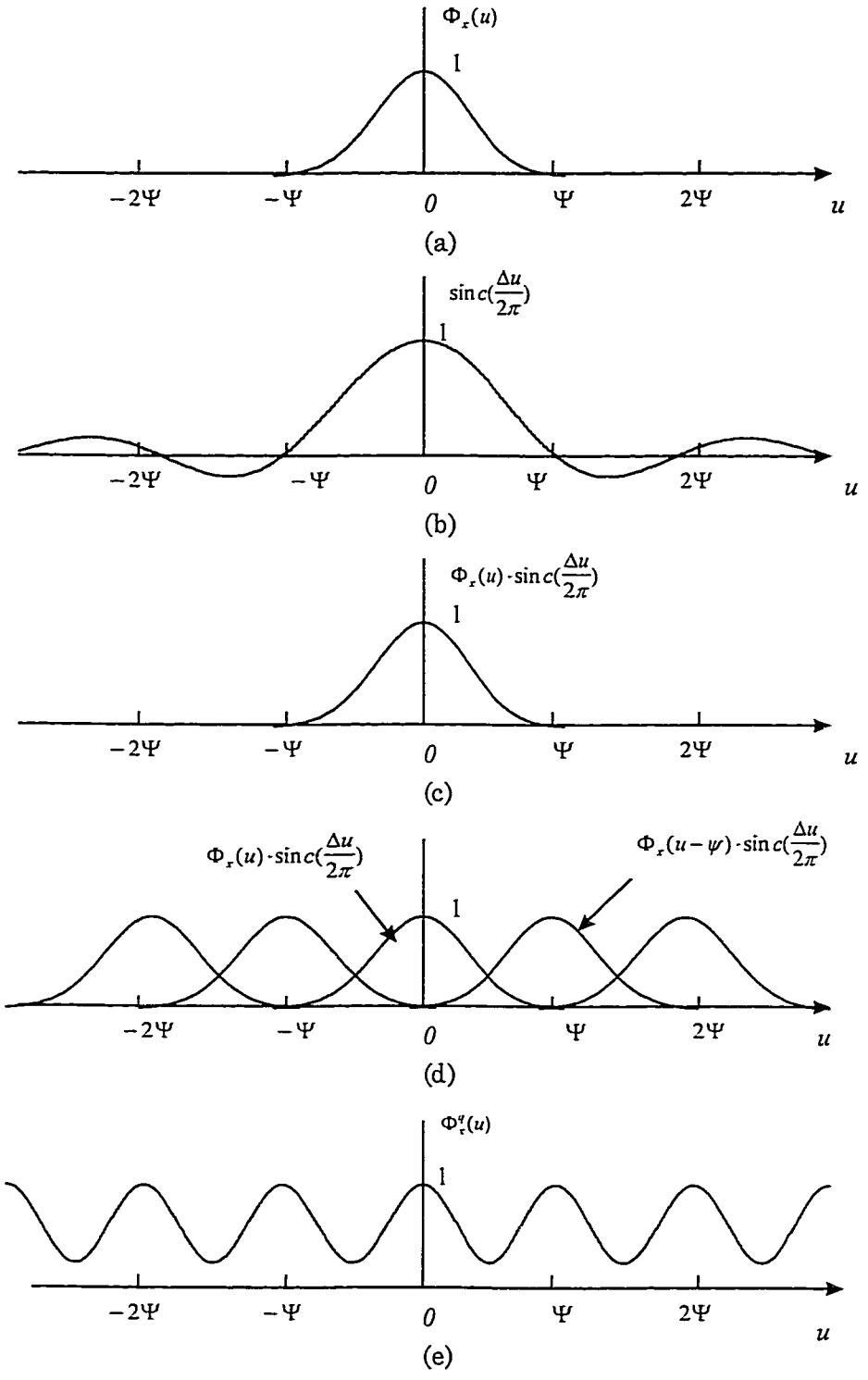


Figure 2.7 Formation of area sampling in the CF domain (adapted from [53])

(a) CF of x; (b) CF of n; (c) CF of $x+n$; (d) the repetition of (c); (e) CF of x_q

The input CF is shown in Fig. 2.7(a), corresponding to the input PDF of Fig. 2.6(a). The FT of the rectangular pulse is shown in Fig. 2.7(b). The product of the FT's is shown in Fig. 2.7(c), corresponding to the convolution of Fig. 2.6(c). The product is repeated in the transform domain with a "frequency" of ψ given by $\psi = 2\pi / \Delta$. The repetition is shown in Fig. 2.7(d), and the sum of the repetitions is shown in Fig. 2.7(e). This is a sketch of the FT of the output PDF of Fig. 2.6(e).

A general expression for the CF of the quantizer output is

$$\Phi_x^q(u) = \sum_{i=-\infty}^{\infty} \Phi_x(u + i\psi) \operatorname{sinc}\left[\frac{\Delta \cdot (u + i\psi)}{2\pi}\right] \qquad (2.7)$$

where $sinc(x) = \dfrac{\sin \pi x}{\pi x}$.

Equation (2.7) clearly shows the repetition at integer multiples of ψ, the quantization frequency. This is analogous to the sampling radian frequency, $\Omega = 2\pi / T$, where T is the sampling period. The sampling period is analogous to the quantization step Δ.

In terms of the bandlimitedness of the CF, a *Quantizing Theorem* can be formulated, as follows.

Quantizing Theorem I (QT I): If the CF of x is band-limited, so that

$$\Phi_x(u) = 0 \quad for \quad |u| > \frac{\pi}{\Delta} = \frac{\psi}{2} \qquad (2.8)$$

then

- the CF of x can be derived from the CF of x_q, and
- the PDF of x can be derived from the PDF of x_q.

The proof is straightforward from Equation (2.8). QT I provides the condition for the output PDF of the quantized signal to contain all the information about the input PDF. In other words, we established a one-to-one connection between the statistical descriptions of the input and output signals of the quantizer.

The above considerations lead to another very important consequence. By taking the central replica of the CF, or equivalently, by interpolating the output PDF, we obtain not the input PDF, but its convolution with a uniform PDF. Therefore, the central replica of the CF is the product of the CF of x and the CF of a uniform distribution.

The analogy between sampling and quantization is even more profound. When a signal is not band-limited, we usually apply an anti-aliasing filter to it before sampling. The anti-aliasing filter multiplies the spectrum by the transfer function of the filter, which is zero outside the desired passband. Similarly, in quantization we can find a way to multiply the CF by a desired function. A product of characteristic functions corresponds to convolution in the PDF domain, since CF's and PDF's are Fourier transform pairs. Convolution of PDF's corresponds to addition of independent random variables. Therefore, we can limit the band of CF by adding an independent random variable with limited CF bandwidth to the input signal. This auxiliary signal is called *dither*.

As we know, when two statistically independent signals are added together, the sum has a PDF which is the convolution of the PDF's of the two signals. Accordingly, the CF of the sum is the product of the CF's of the two signals. These facts are very important in the development of the statistical theory of quantization. It is useful to compare quantization with the addition of uniform independent noise.

Referring to Fig. 2.8(a), quantization of x yields x_q, and addition of independent noise n yields $x+n$. The PDF of the noise to be added is shown in Fig. 2.8(b). The PDF of x_q is discrete, and the PDF of $(x+n)$ is smooth. The latter is equal to the convolution of x, i.e., $f_x(x)$, and the PDF of n, i.e., $f_n(x)$. By inspection of Fig. 2.6(e), one can deduce that the discrete PDF $f_x^q(x)$ of the quantizer output is equal to the samples of the smooth PDF $f_{x+n}(x)$ of the sum of x and n. This is true in general, with or without satisfaction of the quantizing theorem.

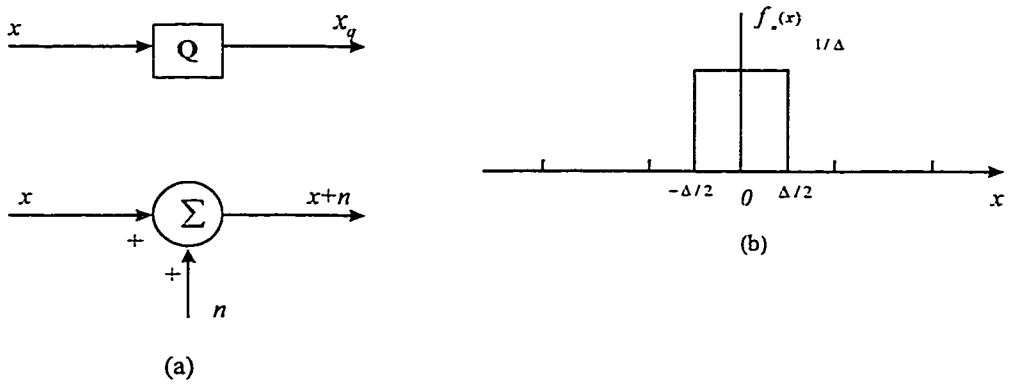


Figure 2.8 Comparison of quantization with addition of independent noise: (a) quantization and noise addition; (b) PDF of the noise.

It is well known that the moments of a random variable x, such as the mean, mean square, mean cube, etc., can be determined by taking derivatives of the CF at the origin. The kth moment is

$$E\{x^k\} = \frac{1}{j^k} \frac{d^k \Phi_x(u)}{du^k}\Big|_{u=0} \qquad (2.9)$$

Referring again to Fig. 2.8(a), it is useful to compare the moments of x_q with those of $x+n$. In general, they do not correspond. But they do correspond when certain quantzing theorems apply.

Refer now to Fig. 2.7. When QT I is satisfied, the replications of Fig. 2.7(d) do not overlap. When summing the replications, the derivatives at the origin in Fig. 2.7(e), related to the moments of x_q, will correspond exactly to the derivatives at the origin in Fig. 2.7(c), related to the moments of $x+n$. Thus, when QT I is satisfied,

$$E\{(x_q)^k\} = E\{(x+n)^k\} \qquad (2.10)$$

This relation will allow the moments of x to be calculated from the moments of x_q. In fact, equation (2.10) will apply even when the replicas of Fig. 2.7(d) overlap, as long as the overlap doses not impact on the derivatives at the origin. This leads to a second quantizing theorem that applies to the moments.

Quantizing Theorem II (QT II): If the CF of x is band-limited so that

$$\Phi_x(u) = 0 \quad for \quad |u| > \frac{2\pi}{\Delta} - \varepsilon = \psi - \varepsilon, \qquad (2.11)$$

with ε positive and arbitrarily small, then the moments of x can be calculated from the moments of x_q.

Next, we will examine what the *quantization error* will be like when QT I or QT II is satisfied.

Quantization error is defined as the difference between the input and output of the quantizer. It may be regarded as the difference between an input variable and the value of the center of the box which it falls into. Therefore, the quantization error is distributed in a certain random form between $[-\frac{\Delta}{2}, \frac{\Delta}{2}]$. A simple method to

obtain the quantization CF was given by Widrow [52]. The distribution of quantization error resulting from inputs within the zero-th box may be constructed by plotting $f_x(x)$ between $[-\frac{\Delta}{2},\frac{\Delta}{2}]$. The distribution of quantization error resulting from inputs within the first box may be obtained by considering $f_x(x)$ for values $[\frac{\Delta}{2},\frac{3\Delta}{2}]$ recentered to the origin. Events taking place in the various boxes are exclusive of each other. The probability of a given error magnitude arising is, therefore, the sum of the probabilities of that noise arising from each quantization box. This summing process can be achieved in the following way. The $f_x(x)$ can be added to itself, shifted a distance Δ to the right, shifted 2Δ to the right, etc., and shifted Δ to the left, shifted 2Δ to the left, etc. This infinite sum could then be multiplied by a "window function" which has the value unity over the range $[-\frac{\Delta}{2},\frac{\Delta}{2}]$ and zero elsewhere.

The sum of the PDF's is represented by equation (2.12):

$$\sum_{k=-\infty}^{\infty} f_x(x+k\Delta) \tag{2.12}$$

The Fourier transform of this sum is

$$\sum_{k=-\infty}^{\infty} \psi\Phi_x(k\psi)\delta(u-k\psi) \tag{2.13}$$

Multiplication in the probability density domain by the window function corresponds in the CF domain to convolution with:

$$\frac{\sin(\pi u/\psi)}{(\pi u/\psi)} \tag{2.14}$$

It follows, therefore, that the CF of the quantization error is:

$$\Phi_q(u) = \sum_{k=-\infty}^{\infty} \psi \Phi_x(k\psi) \frac{\sin \pi(\frac{u}{\psi} - k)}{\pi(\frac{u}{\psi} - k)} \qquad (2.15)$$

An examination of equation (2.15) makes apparent the fact that when QT I is satisfied, the sum has only one nonzero term, which is $\frac{\sin(\pi u / \psi)}{(\pi u / \psi)}$. This means that when QT I holds, the quantization error has a *sinc* CF and the corresponding PDF is uniformly distributed between $\pm\frac{\Delta}{2}$. A closer examination of the equation (2.15) shows that the quantization error will be precisely uniform distributed even when only QT II is satisfied, i.e., when the input CF $\Phi_x(u)$ is zero for $|u| > \psi$, rather than for $|u| > \psi / 2$. It is interesting to note that satisfaction of QT I allows complete recovery of an original probability density given the quantized probability density. Satisfaction of QT II allows only moments to be recovered. At the same time, QT II insures a statistically independent uniform distributed quantization error as well.

So far, we have investigated the fundamentals and the statistical theories of quantization. High-resolution quantization gives relatively precise results, but requires more complex and expensive equipment. Although low-resolution system does not need complex equipment, it may not always reach the requirement on accuracy. The popularity of the modern digital systems demands improvement of quantization. Applying dithering technique to quantization is one of the effective

ways to improve quantization performance, as we have already mentioned. This will be the topic of next section.

2.2 Dithering

Dithering has been used in the past forty years to reduce the effects of quantization noise on speech and visual signal [38]. It also has been effective in improving the resolution of certain systems. Chang and Moore [8] used auxiliary random noise to unbias the output of the multilevel digital correlator. Carley [7] employed dithering techniques in a specific topology for an over-sampling differential pulse code modulation (DPCM)-type A/D converter to improve the linearity of ADC transfer function. White and Pickup [49] used dither to decrease the effects of quantization and differential nonlinearity on systematic errors of digital cross correlators.

Applying dithering technique to quantization, i.e., dithered quantization, is a technique in which a dither signal is added to an input signal prior to quantization. This purposeful distortion of an input signal is common, because it can result in a more subjectively pleasing reproduction and because, under certain conditions, it can cause the quantization error to behave in a statistically nice fashion. In particular, suitably choosing random dither signals can cause the quantization error to be signal independent, uniformly distributed noise. In other words, a suitable dither choice allows the whitening of the quantization error spectrum. This means

that a significant portion of the error power is located outside the signal band and can be easily eliminated by low-pass filtering the quantized data.

Some of the arguments against the use of dither are that more equipment and/or more computing is necessary to generate and apply dither, and that it causes greater output noise in certain systems. The latter objection does not always apply. Usually the reverse is true, particularly in cases where the quantizer is followed by low-pass filter.

In this section, we will investigate the reason for dithering, the theoretical properties of dithering, effects of different types of dither signals on quantization errors. It should be noted that the quantization under our consideration is the uniform quantization and it is in the no-overload region. These will not be repeated in the rest of the thesis.

2.2.1 The Reason for Dithering

As we introduced in the previous section, quantization is a nonlinear operation which takes an input signal $x(t)$ of arbitrary amplitude distribution and produces an output $y(t)$ whose possible amplitude is, at most, a countably infinite number of values. In general, $y(t)$ can be considered as a sum of two terms: the information signal $x(t)$ and an additive noise. Since the quantizer is a nonlinear device, one would expect different distortion effects to be caused by the quantizing operation for different signal inputs $x(t)$; that is to say, the additive noise term is input-dependent. So it is denoted by $n(t,x)$.

But it has been shown by Widrow [52] that the minimum loss of statistical data due to the quantization occurs when *n(t,x)* can be made independent of the signal *x(t)*.

$$n(t,x) = n(t) \qquad (2.16)$$

And the usefulness in satisfying (2.16) is not limited to the recovery of the signal statistics. The results of [38], in the field of TV imply that the satisfaction of (2.16) leads to a minimum loss of the signal waveform accuracy.

As we mentioned in the previous section, Widrow has proved that the quantization error PDF is uniformly distributed between $[-\frac{\Delta}{2},\frac{\Delta}{2}]$, statistically independent of the PDF of the input signal, as long as either QT I or QT II is satisfied. That is

$$\Phi_x(u) = 0 \qquad for \quad all \quad |u| > \frac{2\pi}{\Delta} \qquad (2.17)$$

where $\Phi_x(u)$ is the characteristic function of *x(t)*, Δ is the quantization interval. This is the condition for the one-dimensional case. An analogous "band-limited" result holds for the n-dimensional case.

Just as precisely band-limited signal waveforms occur infrequently in physical situations, precisely band-limited CF's are rare also. Signals that satisfy (2.17) have infinite amplitude ranges so that there are relatively few signals of practical interest which have this property. From a practical standpoint, however, many signals are essentially band-limited, and at the same time, many CF's are almost band-limited, for example, Gaussian input signals with zero means will

approximately satisfy (2.17). Widrow [53] has described, in detail, the quantizer output statistics when the input is a zero-mean Gaussian signal.

Of course, not all random signals can be described as zero-mean Gaussian signals. Fortunately, the condition described by (2.17) can always be met if a proper *dither* signal is used. Dither signal is a signal that is added to the input signal and then removed after the quantization. Obviously, not all signals can be employed as dither signals. Next, we will see what kind of property the dither signals should possess.

2.2.2 Theoretical Property of Dithering

The principal theoretical property of dithering was developed by Schuchman [38] in early 1960's. As we mentioned before, the reason of adding dither signal is to satisfy the quantization theorems so that the quantization error and the input signal can be made independent. Therefore, the goal is to derive the conditions the probability density function of the dither signal, $p_d(d)$, just meet so that the quantization error and signal are independent. The sufficient and necessary condition to satisfy this independence is that the conditional probability density function of the quantization error given the signal ($p(q|x)$) is equal to the unconditional probability density distribution of the quantization error $p(q)$.

When the input signal is known, the quantization error becomes a function only of the PDF of the dither signal. That is,

$$p(q|x=X)=f[p_d(d)] \tag{2.18}$$

where $p_d(d)$ is the PDF of the dither signal. The quantization error e_r is given by (2.19)

$$q=k\Delta-(x+d) \tag{2.19}$$

where

$$k\Delta-\frac{\Delta}{2}\le x+d\le k\Delta+\frac{\Delta}{2}$$

k is an integer and Δ is the quantization interval.

On the other hand, each value of d is uniquely specified by

$$d=k\Delta-x-q \tag{2.20}$$

Hence, we obtain

$$p(q|x=X)=\sum_k p_d(k\Delta-q-X) \tag{2.21}$$

Since we only consider a finite-level, no-overload and uniform quantizer, the following must be true

$$p(q|x=X)=0 \quad for \quad |q|>\frac{\Delta}{2} \tag{2.22}$$

We now find the FT of (2.21) and derive conditions on the characteristic function of the dither signal so that

$$p(q|x=X)=p(q) \tag{2.23}$$

The FT can be written in the following manner

$$\Phi_{q/x}(u)=\int_{-\Delta/2}^{\Delta/2}\sum_k p_d(k\Delta-q-X)e^{-juq}\,dq$$

$$=\int_{-\infty}^{\infty}[s(q+\frac{\Delta}{2})-s(q-\frac{\Delta}{2})]\cdot\sum_k p_d(k\Delta-q-X)e^{-juq}\cdot dq \tag{2.24}$$

where the unit step function is defined as

$$s(x) = \begin{cases} 1 & for \quad x \geq 0 \\ 0 & for \quad x < 0 \end{cases} \tag{2.25}$$

Using the convolution theorem, we obtain

$$\Phi_{q/x}(u) = \int_{-\infty}^{\infty} [s(q + \frac{\Delta}{2}) - s(q - \frac{\Delta}{2})] \cdot e^{-juq} dq * \int_{-\infty}^{\infty} \sum_k p_d(k\Delta - q - X) e^{-juq} dq \tag{2.26}$$

By evaluating the first integral and interchanging the order of summation and integration, the expansion becomes

$$\Phi_{q/x}(u) = \Delta \cdot \sin c \frac{\Delta u}{2\pi} * \sum_k \int_{-\infty}^{\infty} p_d(k\Delta - q - X) e^{-juq} dq$$

$$= -\Delta \cdot \sin c \frac{\Delta u}{2\pi} * \sum_k \int_{-\infty}^{\infty} p_d(v) e^{ju(v + X - k\Delta)} dv \tag{2.27}$$

Note that we changed the variable in the integration so that $v = k\Delta - q - X$. So the integration above becomes

$$\Phi_{q/x}(u) = -\Delta \cdot \sin c \frac{\Delta u}{2\pi} * \sum_k e^{-ju(k\Delta - X)} \int_{-\infty}^{\infty} p_d(v) e^{-juv} dv$$

$$= -\Delta \cdot \sin c \frac{\Delta u}{2\pi} * \frac{1}{\Delta} \sum_k \delta(u - \frac{k}{\Delta}) e^{j2\pi uX} \Phi_d(u)$$

$$= \sum_k \Phi_d(\frac{k}{\Delta}) e^{-j\pi Xk/\Delta} \sin c \frac{\Delta}{2\pi}(u - \frac{k}{\Delta}) \tag{2.28}$$

Therefore, if the transform is zero at the points

$$\Phi_d(\frac{k}{\Delta}) = 0 \qquad for \quad k = \pm 1, \pm 2, \ldots \tag{2.29}$$

then, the characteristic function $\Phi_{q/x}(u)$ is independent of x and $p(q|x) = p(q)$. The result shows the sufficiency of (2.29) but not its necessity.

If $\Phi_d(\frac{k}{\Delta}) \neq 0$ for some value of k, then, from (2.28), $\Phi_{q/x}(u)$ must be a periodic function of x. If $\Phi_{q/x}(u)$ is a periodic function of x, then its transform is a function of x. Therefore, $p(q|x)$ is not independent of x and $p(q|x) \neq p(q)$. Thus, (2.29) is the necessary and sufficient condition for $p(q|x) = p(q)$.

We now note the following:

(1) All PDF's that satisfy (2.17) can also satisfy (2.29),

(2) An infinite class of PDF's that are not "band-limited" and that satisfy (2.29) are

$$p_d(v) = \frac{1}{k\Delta}[s(v + \frac{k\Delta}{2}) - s(v - \frac{k\Delta}{2})] * f_\alpha(v) \qquad (2.30)$$

where $f_\alpha(v)$ can be any arbitrary density distribution, k is any positive integer (1,2,3...).

One of the most widely used dither signals, which satisfies (2.30) is the uniform dither. Consider the simplest case that $f_\alpha(v) = \delta(v)$ and k=1, then we obtain

$$p_d(v) = \frac{1}{\Delta}[s(v + \frac{\Delta}{2}) - s(v - \frac{\Delta}{2})] \qquad (2.31)$$

The PDF of this uniform dither is depicted in Fig. 2.9.

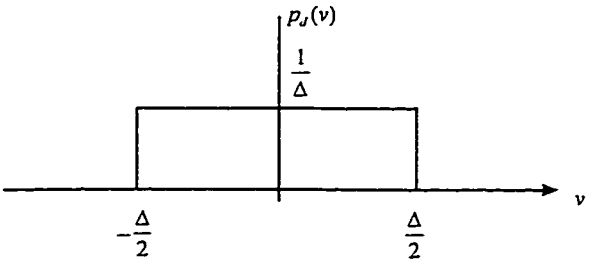


Figure 2.9 Probability density function of uniform dither

There are different types of dither, which we will introduce and analyze in the next section.

2.2.3 Different types of Dither

Dithering decorrelates signal-dependent noise, thus resulting in no harmonic distortion at the expense of increasing signal-independent noise [47]. The practical application of dither to Analog/Digital Converter (ADC) has been investigated before [5], [11], and [45]. It tends to linearize the average observed ADC transfer characteristics, thus effectively increasing the ADC resolution.

There are different types of dither signal. Due to the lack of quantitative criteria for comparing the effects of dither types, how to choose a proper dither signal remains a problem to be studied further. Carbone and Petri [6] analyzed the effect of additive dither on the quantization error average of a slowly varying signal. They obtained a lower bound on the resolution performances that can be achieved by adopting dithering techniques. And they concluded that uniform dither and Gaussian dither outperform discrete dither and sinusoidal dither. Wadgy *et al.*, [46]-[48] also tried to evaluate the effect of various dither forms on quantization errors for ideal ADC using the concept of nonlinearity spectra that he developed in [46].

We will also adopt the concept of nonlinearity spectra and analyze three most common dither types: uniform dither, triangle dither and Gaussian dither.

Let us introduce the concept of nonlinearity spectra first. Although previous simulation [44] showed that dithering reduces harmonic distortion, there was no mathematical basis. To fill this gap, Wadgy [47] studied the spectra of the average quantization errors.

The quantization error function for an ideal ADC is shown in Fig. 2.2(b), with x being the quantizer input. The quantization error function is readily expanded as Fourier series in x [49]:

$$q(x) = \sum_{n=1}^{\infty} \frac{\Delta}{n\pi} \sin\left(\frac{2\pi n x}{\Delta}\right) \cdot (-1)^n \qquad (2.32)$$

When a dither signal d is added to the signal x at the quantizer input, then the average observed error at the ADC output is

$$\bar{q}(x) = \int_{-\infty}^{\infty} q(x+v) \cdot p_d(v) dv \qquad (2.33)$$

where $p_d(v)$ is the PDF of dither. Equation (2.33) can be recognized as a convolution integral and transformed to the frequency domain as

$$\bar{Q}(\omega) = Q(\omega) \cdot P_d(\omega) \qquad (2.34)$$

where $\bar{Q}(\omega)$, $Q(\omega)$ and $P_x(\omega)$ are the FT's of $\bar{q}(x)$, $q(x)$ and $p_d(v)$, respectively.

Considering (2.32), we find that the positive frequency side of the dither-free amplitude spectrum is given by

$$|Q(\omega)| = \sum_{n=1}^{\infty} \frac{\Delta}{n} \delta(\omega - n\omega_0) \qquad (2.35)$$

where $\omega_0 = 2\pi / \Delta$. Therefore, $|\bar{Q}(\omega)|$, the amplitude spectrum of the average observed quantization error, is given by the equation (2.36).

$$|\overline{Q}(\omega)| = \sum_{n=1}^{\infty} \frac{\Delta}{n} P_d(n\omega_0) \cdot \delta(\omega - \omega_0) \qquad (2.36)$$

We will now introduce three most common used dither forms.

(1) Uniform Dither

The expression of uniform dither PDF has been given in equation (2.31) and Fig. 2.9. A more general expression of it is given below:

$$p_d(v) = \begin{cases} \dfrac{1}{2A_u} & |v| \le A_u \\ 0 & otherwise \end{cases} \qquad (2.37)$$

where A_u is the amplitude of the dither signal. The FT of (2.37) is given in (2.38).

$$\Phi_d(\omega) = \frac{\sin(\omega A_u)}{\omega A_u} \qquad (2.38)$$

Theoretically, uniform dithering can provide an infinite resolution, but its performance may decrease quickly in the presence of dither amplitude uncertainties. But if we have enough *a prior* experience of the quantized signal and make a good control over the quantization so that the quantization is in the no-overload region, uniform dither is often a good choice. This will be indicated in the applications coming later.

(2) Triangle Dither

The PDF of triangle dither is sketched in Fig. 2.10. The general expression is given in equation (2.39).

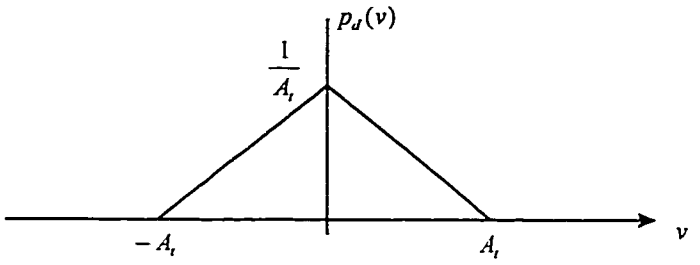


Figure 2.10 Probability density function of triangle dither

$$p_d(v) = \begin{cases} \frac{1}{A_t}(1 - \frac{|v|}{A_t}), & |v| \le A_t \\ 0, & otherwise \end{cases} \tag{2.39}$$

where A_t is the amplitude of the dither signal. And the FT of (2.39) is given in (2.40).

$$\Phi_d(\omega) = \left(\frac{\sin(\omega A_u / 2)}{\omega A_u / 2} \right)^2 \tag{2.40}$$

Stockham [14], when working on one of the first digital recordings of rock and roll in 1980, found out that by using triangle dither signal, the perceived noise level did stay constant, regardless of the signal. But using uniform dithering could not achieve this effect. This was an example indicating that uniform dither is not always the best dither signal to be employed.

It is worth noticing that a triangle dither can be obtained as follows. Add two independent uniformly distributed random variables, then you will have a triangle density random variable.

(3) Gaussian Dither

The PDF of Gaussian dither is sketched in Fig. 2.11. The general expression is given in equation (2.41).

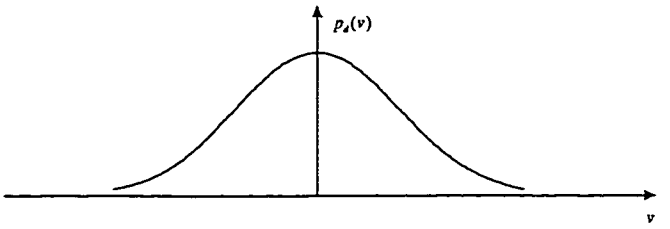


Figure 2.11 Probability density function of Gaussian dither

$$p_d(v) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{v^2}{2\sigma^2}} \qquad (2.41)$$

where σ is the standard deviation of dither signal. The FT of (2.41) is given by:

$$\Phi_d(\omega) = e^{\frac{\sigma^2\omega^2}{2}} \qquad (2.42)$$

Usually, Gaussian dither has a random nature. It's is worth noticing that any unintentional analog noise, e.g., due to the electronic circuits of the data acquisition system or embedded in the input system, can often be seen as a Gaussian dither. For this reason, this type of dither is sometimes called a "self-dither" signal.

To compare the effects of these, all the dither powers are chosen to be equal, i.e.,

$$\frac{A_t^2}{6} = \frac{A_u^2}{3} \quad i.e. \quad A_t = \sqrt{2}\, A_u \qquad (2.43)$$

and

$$\sigma^2 = \frac{A_u^2}{3} \quad i.e. \quad \sigma = \frac{A_u}{\sqrt{3}} \tag{2.44}$$

The amplitude spectra of average observed quantization error for different forms are given below:

Uniform dither:

$$|\overline{Q}(\omega)| = \sum_{n=1}^{\infty} \frac{\Delta}{n} \sin c(2n\frac{A_u}{\Delta}) \cdot \delta(\omega - \omega_0) \tag{2.45}$$

Triangle dither:

$$|\overline{Q}(\omega)| = \sum_{n=1}^{\infty} \frac{\Delta}{n} \sin c^2(n\frac{A_t}{\Delta}) \cdot \delta(\omega - \omega_0) \tag{2.46}$$

Gaussian dither:

$$|\overline{Q}(\omega)| = \sum_{n=1}^{\infty} \frac{\Delta}{n} \exp(\frac{-2\pi^2\sigma^2 n^2}{\Delta^2}) \cdot \delta(\omega - \omega_0) \tag{2.47}$$

In order to compare the effects of the above three dither forms on the smeared quantization error function $\overline{q}(x)$, a figure of merit based on $|\overline{Q}(\omega)|$ is devised as follows:

$$|\overline{Q}(\omega)| = \sum_{n=1}^{\infty} A_n \cdot \delta(\omega - n\omega_0) \tag{2.48}$$

where A_n is the magnitude of the harmonic component at $n\omega_0$. Now the deviation of the smeared ADC characteristics from the 45° line is represented by a deviation factor D given by

$$D = \sum_{n=1}^{\infty} A_n^2 \tag{2.49}$$

The D factor corresponding to (2.45), (2.46) and (2.47) is plotted in Fig. 2.12. Let the normalized value A_u / Δ be simply denoted as C. Note that (2.43) and (2.44) are taken into consideration.

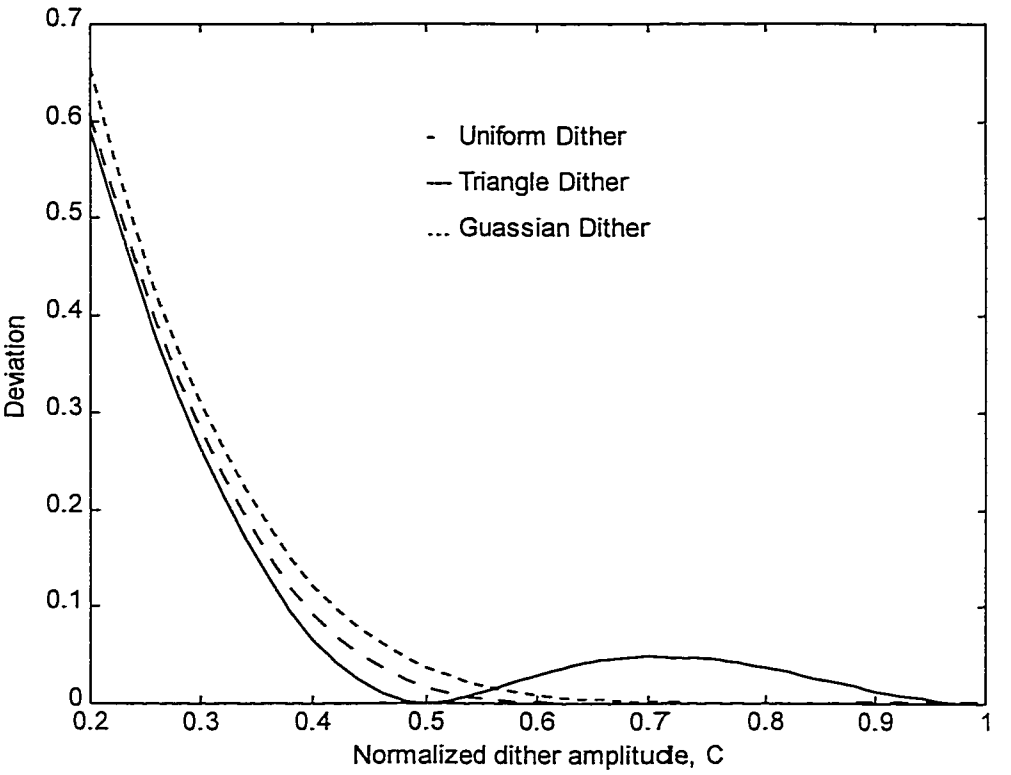


Figure 2.12 Deviation factor D of ADC smeared characteristics versus normalized dither amplitude C for various analog PDF's (with equal noise power)

It is obvious, from Fig. 2.12, that for C=0.5, uniform dither is the only type that gives zero D, i.e., perfectly linear ADC averaged characteristics, whereas triangle dither is slightly better than Guassian dither. Uniform dither is sensitive to alterations in the peak value C. However, for about $\pm 7.5\%$ deviation from C=0.5, the uniform dither still gives the least D. Thus uniform dither is the practical dither for C=0.5, i.e., $A_u = \Delta/2$. When $C > 0.5$, triangle dither and Guassian dither outperform the uniform dither not only because of the smaller deviation, but also because they are less sensitive to alterations in C than uniform dither.

There will be comparison of effect of different forms of dither on system performance when we come up with the applications.

2.2.4 Nonsubtractive and Subtractive Dithered Quantizer

In a dithered quantizer, instead of quantizing an input signal X_n directly, one quantizes a signal

$$U_n = X_n + W_n \tag{2.50}$$

where W_n is a random process, independent of the signal X_n, called a *dither* process. The dither process is usually assumed to be i.i.d., and will be so assumed here.

The idea of dithering is that such randomization can cause the quantization error

$$q_n = Q(X_n + W_n) - (X_n + W_n) \tag{2.51}$$

to have the properties assumed in Bennett approximation, i.e., the quantization error is statistically independent of the input signal. *Quantization error* means the error between the input and output of the quantizer. We will refer to the overall difference between original input and final output

$$\varepsilon_n = Q(X_n + W_n) - X_n = q_n + W_n \qquad (2.52)$$

as the *quantization noise.*

The difference between quantization error and quantization noise can be highlighted by distinguishing between *subtractive dither* and *nonsubtractive dither*, which are shown in Fig. 2.13 and Fig. 2.14, respectively.

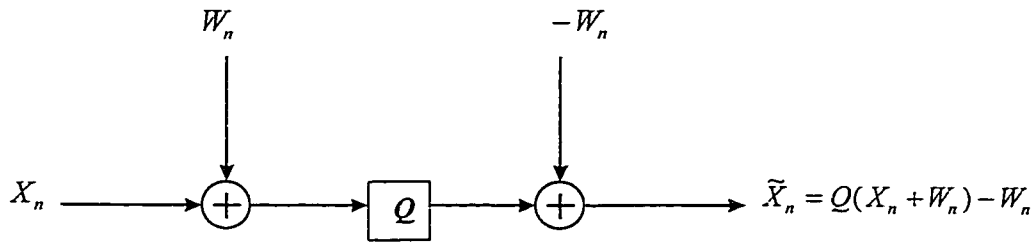


Figure 2.13 Subtractive dithered quantizer

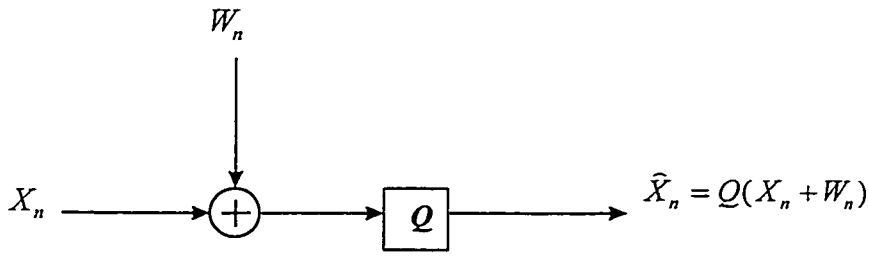


Figure 2.14 Nonsubtractive dithered quantizer

In the *subtractive dithered quantizer*, which was suggested by Roberts[38], the receiver knows the dither signal and subtracts it from the reconstructed quantizer value to produce a final reproduction $\tilde{X}_n$ of the original input X_n. Although there is no guarantee that a dither signal added prior to the nonlinear quantization can be removed by subtracting it from the resulting digital signal, a startling thing happens if the overall quantizer noise in the subtractive dither system is considered:

$$\varepsilon_n = \tilde{X}_n - X_n = Q(X_n + W_n) - W_n - X_n = q_n. \qquad (2.53)$$

That is, in this system the overall quantization noise is equal to the quantizer error in the original nonsubtractive dithered quantizer of Fig.2.14, which is indeed uniform and independent of input if Schuchman's conditions are met.

In a communication system, the subtraction operation must take place in the receiver and implies the use of a known dither waveform and the requirement of synchronous information being transmitted to the receiver.

The subtractive dither result is nice mathematically because it promises a well-behaved quantization noise as well as quantization error. It is impractical in many applications, however, for two reasons. First, the receiver will usually not have a perfect analog link to the transmitter (or else the original signal could be sent in analog form), and hence a pseudo-random deterministic sequence must be used at both transmitter and receiver as proposed in [38]. In this case, however, there will be no mathematical guarantee that the quantization error and noise have the properties that hold for genuinely random i.i.d. dither. Second, subtractive dither of a signal that indeed resembles a sample function of a memory-less random process is complicated to implement, requiring storage of the dither signal, high-precision

arithmetic and perfect synchronization. As a result, it is of interest to study the behavior of the quantization noise in a simple non-subtractive dithered quantizer, that is the simplest dithered quantizer of Fig.2.14. One might correctly suspect that nonsubtractive dither does not possess the nice properties of the quantization error that are promised by subtractive dither, but it remains of interest to study this less difficult type of dithered quantizer.

Nonsubtractive dither, from my point of view, is not real nonsubtractive. Because there does exist an operation of subtraction in the nonsubtractive dither, but in different forms.

For example, Jayant and Noll [20] presented results of applying the dithering technique to image and speech processing. They compared subtractive and nonsubtractive dither in both areas, and pointed out that an important factor in image dithering is the low-pass filtering action of the eye. The low-pass function of the eye gives a smooth appearance to the oscillatory outputs of dithered system reconstruction. While the absence of such low-pass filtering in speech perception makes dither subtraction more critical in speech work. And this is one reason why implementations of dithering in video work have been more extensive than in speech coding.

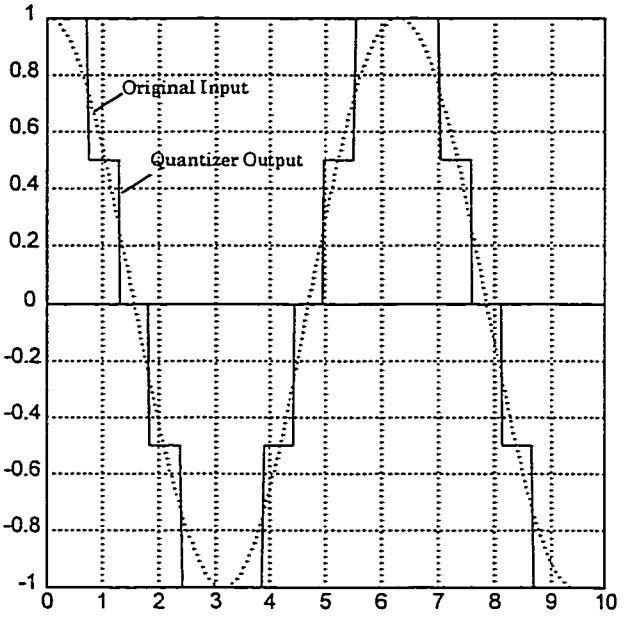
Another example is the research work on which this thesis is based. When apply dithering technique to digital Fuzzy Logic Controller, we add a low-pass filter to filter the output of the digital fuzzy logic controller. Results have shown that a dithered quantizer followed by a low-pass filter improves the sensor resolution significantly. Note that nonsubtractive dithered quantizer, followed by a low-pass filter is adopted in the applications considered in this thesis.

Using ordinary low-pass filter produces an effect on the characteristic functions which is like low-pass filtering in the CF domain [53]. Often this smoothing effect on CF's is adequate so that the input signal to the quantizer has the same statistical characteristics as if the quantization error were independent. Because nonsubtractive dithered quantizer and a low-pass filter can be implemented with little difficulty, this type is more popular.

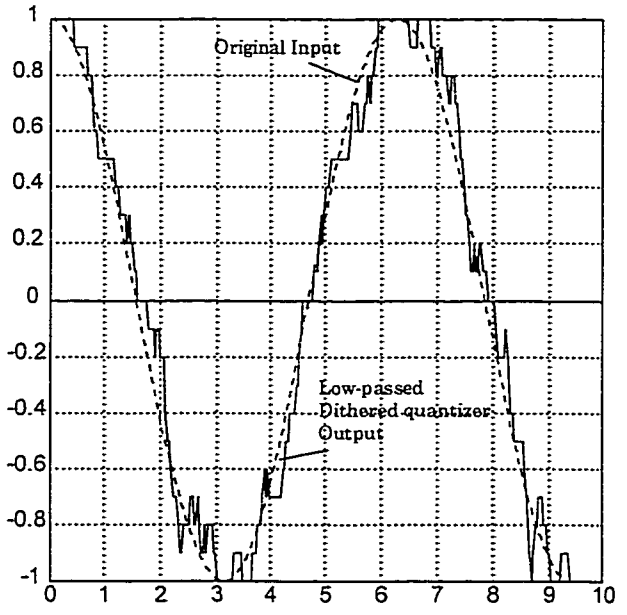
In practical applications, the quantizer is usually followed by a low-pass filter whose output represents an estimation of the mean of the original input signal, e.g., the moving average filter whose output is the normalized sum of N successive quantized data. It is understandable that bigger N results in better estimation of the mean, but longer delay of output. In different systems, the extent that the delay will affect the performance is different. Sometimes, the delay may cause the failure of the system. We need to make a tradeoff between good approximation and short delay.

To have an impression of the effects of dithering and low-pass filtering, let us look at a simple example. We quantized a signal $y=cos(x)$ with different quantization resolutions previously. Now, we add a uniform dither whose PDF is shown in Fig.2.9 before quantization and apply a low-pass filter after the quantization.

Fig. 2.15(a) shows the 2-bit quantization result without dithering. This quantization is too coarse to approximate the original input signal. Fig. 2.15(b) depicts the result of the following operations: adding uniform dither to the original signal, doing 2-bit quantization, and low-passed by a 5-unit moving average filter. Although the curve displayed in Fig. 2.15(b) is not very smooth, but it is a better approximation of the original input.



(a)



(b)

Figure 2.15 Applying dithering to quantization

(a) quantization without dithering;

(b) quantization with dithering and low-pass filtering

Chapter 3 Fuzzy Logic Control

The focus of this research work is to study the application of the dithering technique to Fuzzy Logic Controllers. Therefore, this chapter is to give a brief introduction to the Fuzzy Logic and Fuzzy Logic Controllers.

For the convenience of the reader, we shall briefly summarize some of the basic concepts and terms used in Fuzzy Logic in the first section of this chapter. The second section is dealing with the fundamentals of Fuzzy Logic Controller (FLC). The last section shows a simple example of FLC for car braking.

3.1 Fuzzy Logic

In the middle 1960' s, Lotfi Zadeh recognized that the true-or-false nature of Boolean logic did not account for the vagueness in real world. To account for the infinite gradations between true and false, Zadeh [54] expanded the idea of a classical set to what he termed a *fuzzy set*, which is the corner stone of Fuzzy Logic (FL).

Boolean logic is also referred to as conventional logic, or classical logic. A variable in Boolean logic takes the value of either true or false, but not both.

Unlike Boolean logic, FL is multi-valued. Instead of an element being 100% this or that, or a proposition that is either entirely true or completely false, fuzzy deals in *degrees* of membership and *degrees* of truth --- that is, something can be partially true and partially false at the same time. It has been proven by Bart Kosko that Boolean logic is a special case of fuzzy logic.

3.1.1 Fuzzy Sets and Related Concepts

The idea of Fuzzy Logic is easy to grasp. Let us introduce some basic concepts and terms first.

Fuzzy set, which is characterized by *membership function,* is a set with fuzzy boundaries. A membership function is a mathematical function that gives numerical meaning to a fuzzy set. It maps crisp inputs from a specified domain to membership grades ranging between 0 (zero membership) and 1 (total membership). The membership grade is called *degree of membership*. An element in a fuzzy set manifests itself by degree of membership. Each membership function has a descriptive name, *label*, to identify itself.

Note that *crisp inputs* are distinct and sharply divided inputs, e.g., 25.5°. *Domain* is the width of a membership function. There is another term, *universe of discourse*, should be clarified. It denotes the range of all possible values applicable to a system variable.

Now, let us look at a problem in everyday life. A temperature of 28°*c* may be warm to some people, but hot to others. To describe the temperature, we will see that Boolean Logic and Fuzzy Logic can give us significantly different results. Boolean Logic classifies this problem in a way that is shown in Fig. 3.1.

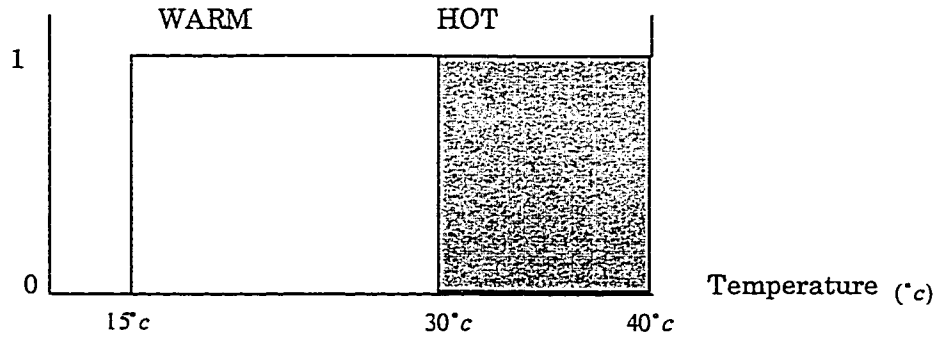


Figure 3.1 Classical logic/Conventional sets

Examine Fig. 3.1, we can find that the transition from set to set is instantaneous. Using these conventional sets, 29.9°*c* would be classified as *warm* and 30.1°*c* would be classified as *hot*. This means that small changes in the system could cause significantly different reaction. It is natural to arrive at a conclusion that Boolean Logic is not appropriate to solve this type of problem

If we apply Fuzzy Logic/Fuzzy Sets to this problem, we will have a different result. In Fig. 3.2, the transition from a set to another could be gradual, i.e., an element can have partial membership in multiple sets. Here, 29.9°*c* and 30.1°*c* belong to the same sets, *warm* and *hot*, at the same time and at approximately the same degrees. It illustrates that small changes in system will result in more graceful

change in system performance, which is closer to the natural life. Of course, the classification in Fig. 3.2 is not the only choice, but one of the simplest.

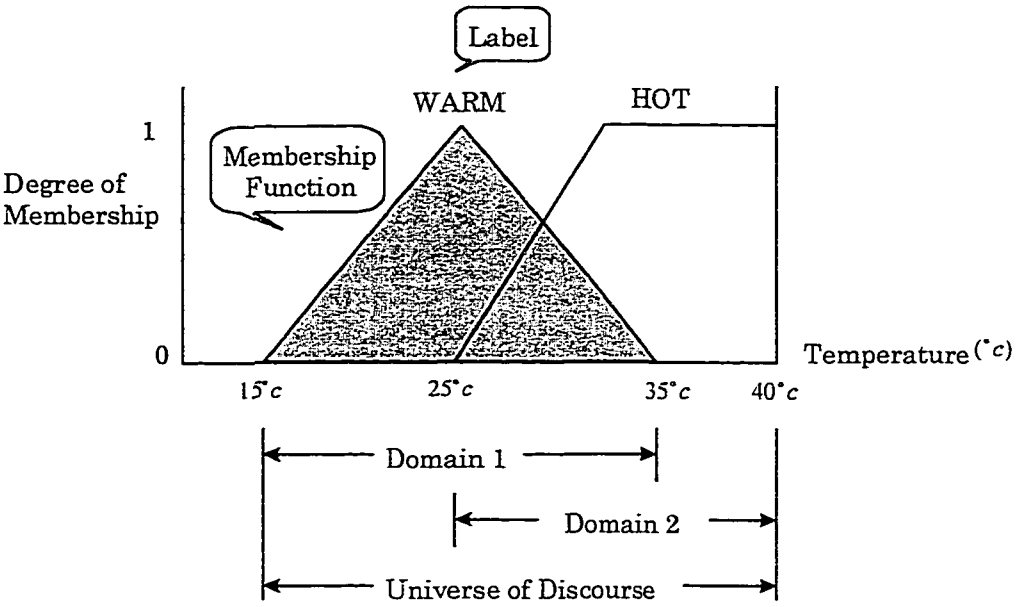


Figure 3.2 Fuzzy Logic/Fuzzy sets

You may find that there are two kinds of membership function in Fig. 3.2, triangular and trapezoidal membership functions. As a matter of fact, there are many membership functions from which you can choose. For example, smooth and symmetric shaped Gaussian membership function, and smooth but asymmetric shaped sigmoid membership functions. For more details, refer to MATLAB/Fuzzy Logic Toolbox. Because the triangular and trapezoidal membership functions are straight-line membership functions, which have an advantage of simplicity, they will be of our interest in designing FLC. No further introduction about other membership functions will be given.

3.1.2 Basic Logic Operations

The most important thing to realize about fuzzy logical reasoning is the fact that it is a superset of standard Boolean Logic. In other words, if we keep the fuzzy values to the extremes of 1 (completely true) and 0 (completely false), standard logical operations will hold.

Consider the three basic logical operations: AND, OR and NOT. If we resolve the statement A AND B, where A and B are limited to range [0,1], by using the function *min*(A,B), we can still preserve the truth table of standard AND, shown in Fig. 3.3.

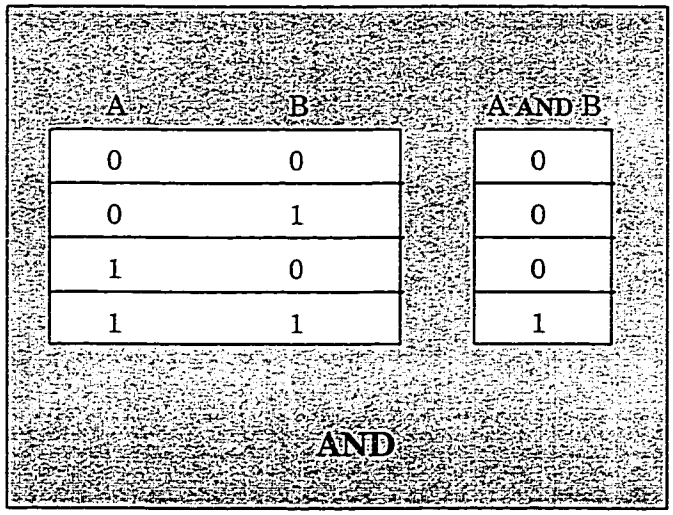


Figure 3.3 Truth table of standard AND

Using the same reasoning, we can replace the OR operation with the *max* function, so that A OR B becomes equivalent to the operation *max*(A,B), in the view of truth table. Finally, the operation NOT A becomes equivalent to the operation 1-A.

Since there is a function behind the truth table rather than just the truth table itself, we can now go on to consider values other than 1 and 0.

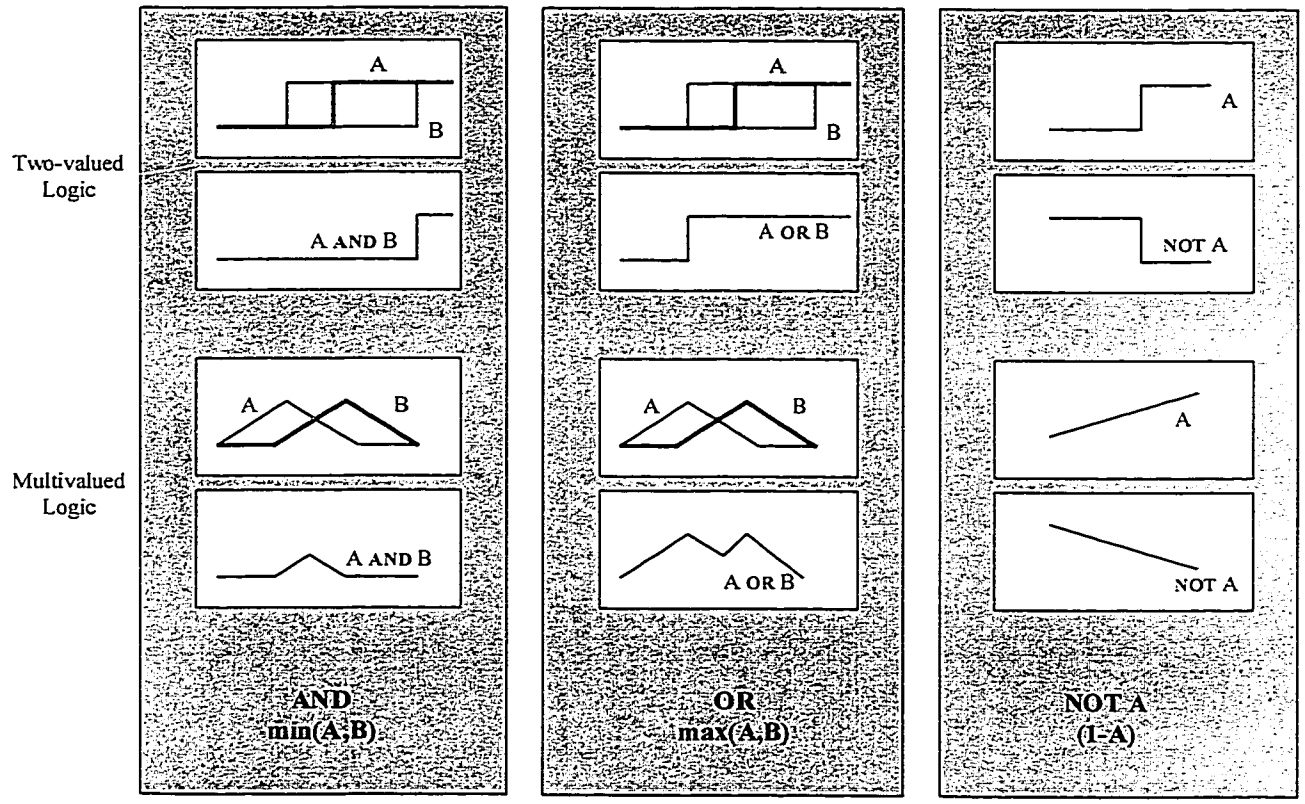


Figure 3.4 Two-valued and multi-valued logic

The upper part of Fig. 3.4 displays three basic logical operations in two-valued logic, while the lower part displays how the operation works over continuously varying range of truth values A and B according to the fuzzy operations defined above.

In more general terms, what we are defining are known as the fuzzy intersection or conjunction (AND), fuzzy union or disjunction (OR), and fuzzy complement (NOT). The correspondence between two-valued and multivalued logical operations for these functions (AND, OR and NOT) is by no means unique. AND=*min*, OR=*max* and NOT=*additive complement*, are called the classical operators for these functions. Typically, most fuzzy logic applications make use of these operations and leave it at that. Therefore, we will stop here and go no further about this topic.

3.1.3 Features of Fuzzy Logic

Some of the general observations about FL are given below[18].

- FL is conceptually easy to understand.

  The mathematical concepts behind fuzzy reasoning are very simple. What makes fuzzy nice is the "naturalness" of its approach and not its far-reaching complexity.

- FL is flexible.

  With any given system, it is easy to massage it or layer more functionality on top of it without starting again from scratch.

- FL is tolerant of imprecise data.

  Everything is imprecise if you look closely enough, but more than that, most things are imprecise even on careful inspection. Fuzzy reasoning builds this understanding into the process rather than tacking it onto the end.

- FL can model nonlinear functions of arbitrary complexity.

  You can create a fuzzy system to match any set of input-output data. This process is made particularly easy by adaptive techniques.

- FL can be built on the experience of experts.

  In direct contrast to neural networks, which take training data and generate opaque, impenetrable models, fuzzy logic lets you stand on the shoulders of people who already understand your system.

- FL can be blended with conventional control techniques.

  Fuzzy systems do not necessarily replace conventional control methods. In many cases fuzzy systems augment them and simplify their implementation.

- FL is based on natural language.

  The basis for fuzzy logic is the basis for human communication. This observation underpins many of the other statements about fuzzy logic.

The last sentence may be the most important one. Natural language, which is used by ordinary people on a daily basis, has been shaped by thousands of years of human history to be convenient and efficient. Since fuzzy logic is built atop of the structures of everyday language, it not only makes it easy for us to use it, but it also takes advantage of the long history of natural language.

It is worth noticing that the integration of fuzzy logic into process engineering and information systems is producing applications such as control systems, consumer appliances, and decision systems that are remarkably more flexible robust, and sophisticated than their conventional counterparts. In this regard, we can say that fuzzy logic leads to the development of a higher machine intelligence quotient or MIQ. These are already appearing in such consumer goods as washing machines, video and single-lens reflex cameras, air conditioners, microwave ovens and many self-diagnosing systems.

3.2 Fuzzy Logic Controller

System modeling based on conventional mathematical tools (e.g., differential equations) is not well suited for dealing with ill-defined and uncertain systems[17]. By contrast, a fuzzy inference system employing fuzzy if-then rules can model the qualitative aspects of human knowledge and reasoning processes without employing precise quantitative analyses. Fuzzy inference systems are also known as fuzzy models, fuzzy associative memories (FAM), or fuzzy logic controllers when used as controllers.

Fuzzy Logic Controller is based on fuzzy logic, which is much closer in spirit to human thinking and natural language than the traditional logical systems. Basically, fuzzy logic provides an effective means of capturing the approximate, inexact nature of the real world. Viewed in this perspective, the essential part of the fuzzy logic controller (FLC) is a set of linguistic control rules related by the dual

concepts of fuzzy implication and the compositional rule of inference. In essence, then, the FLC provides an algorithm, which can convert the linguistic control strategy based on expert knowledge into an automatic control strategy. Experience shows that the FLC yields results superior to those obtained by conventional control algorithms. In particular, the methodology of the FLC appears very useful when the processes are too complex for analysis by conventional quantitative techniques or when the available sources of information are interpreted qualitatively, or uncertainly. Thus fuzzy logic control may be viewed as a step toward a rapprochement between conventional precise mathematical control and human-like decision making.

3.2.1 Fundamentals of FLC

Basically, a fuzzy logic controller is composed of five functional blocks (see Fig. 3.5):

- A database, which defines the membership functions of the fuzzy sets used in the fuzzy rules;
- A rule base containing a number of fuzzy if-then rules;
- A decision logic, which performs the inference operations on the rules;
- A fuzzifier, which transforms the crisp inputs into degrees of match with linguistic values;
- A defuzzifier, which transforms the fuzzy results of the inference into a crisp output.

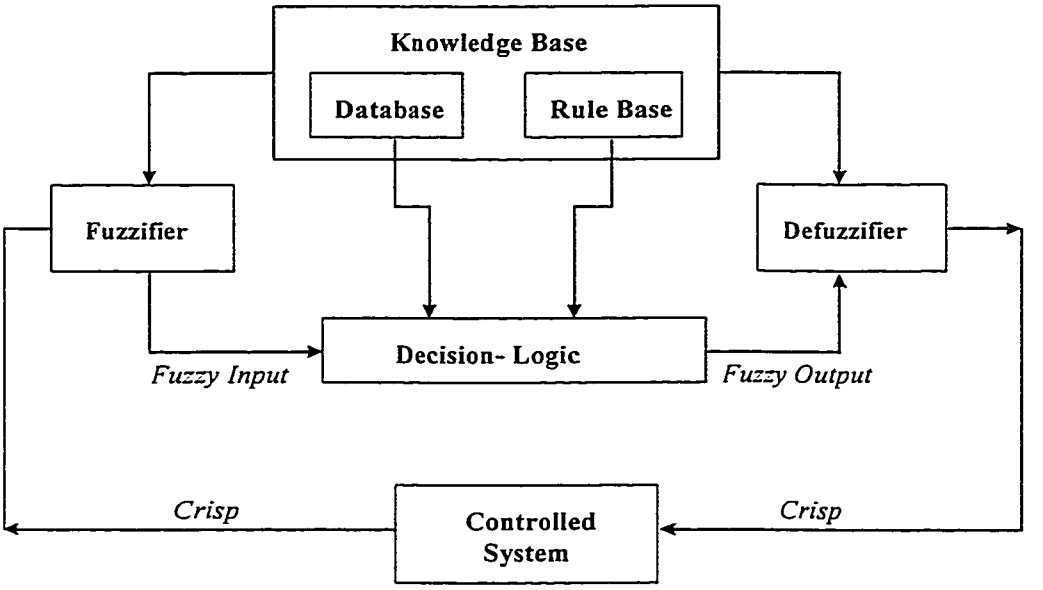


Figure 3.5 Basic Architecture of FLC System

It should be explained here that fuzzy rules are usually if-then statements that describe the action to be taken in response to various fuzzy inputs. Although rules look like free from natural language, they are confined to a predefined set of linguistic terms, and a strict syntax. The syntax is:

If *antecedent 1 AND antecedent 2*··· then *consequent 1 AND consequent 2*···

where the antecedent is in the form of: *input variable = label*, and the consequent is in the form of: *output variable = label.*

Rules follow the common sense behavior of the system and are written in terms of the membership function linguistic labels.

The three key steps of fuzzy logic processing are as follows:

(1) *Fuzzification* - the process of mapping the current system input values onto input membership functions to determine the resultant truth value for each label (the linguistic description or name assigned to a membership function). The result is the fuzzy input.

(2) *Fuzzy inference* - the process of calculating the relative applicability, or "truth value", for each rule. In Min-Max inference, one of the most often employed inference methods, this is equal to the minimum of the antecedents (fuzzy inputs) for that rule. Fuzzy outputs are then calculated by determining the maximum rule strength for each output label.

(3) *Defuzzification* - the process of calculating the crisp output based on all fuzzy outputs for a given output variable. There are many defuzzification methods, but the most commonly used defuzzification technique is called Center of Gravity (COG) method or centroid method.

3.2.2 Fuzzy Inference

Basically, there are two types of fuzzy inference style, Mamdani-style inference and Sugeno-style inference.

Mamdani-style inference was among the first control systems built using fuzzy set theory. It was proposed in 1975 by Ebrahim Mamdani [27], as an attempt to control a steam engine and broiler combination by synthesizing a set of linguistic control rules obtained form experienced human operators. Mamdani inference expects the output membership functions to be fuzzy sets. After the aggregation process, there is a fuzzy set for each output variable that needs defuzzification. It is possible, and in many cases much more efficient, to use a single spike as the output membership function rather than a distributed fuzzy set. This is sometimes known as a *singleton* output membership function, and it can be thought of as a pre-defuzzified fuzzy set. It enhances the efficiency of the defuzzification process because it greatly simplifies the computation required to find the centroid of a two-dimensional shape. Rather than integrating across a continuously varying two-dimensional shape to find the centroid, we can just find the weighted average of a few data points. Sugeno-style inference supports this kind of behavior.

Sugeno-style inference was first introduced in 1985 by M. Sugeno [43]. It is similar to the Mamdani method in many respects. For instance, applying the fuzzy operator is exactly the same. There are zero-order, first-order and high-order Sugeno fuzzy models, we will only introduce zero-order Sugeno fuzzy model here because it is frequently the case that singleton output functions are completely sufficient for a given problem's needs.

A typical fuzzy rule in a zero-order Sugeno fuzzy model has the form

If x is A and y is B then $z=k$

where A and B are fuzzy sets in the antecedent, while k is a crisply defined constant in the consequent.

The advantages of Sugeno inference are its computational efficiency and guaranteed continuity of the output surface. We will employ it in our example late in this chapter and the applications in next chapter.

3.3 An Example - Car Braking System

So far, we have introduced the fundamentals of FLC. We will now adopt a car braking system example to show how an FLC works.

The car braking system under consideration is a two-input, one-output system. It uses *car speed* and *distance* from object to determine the *force* to apply to the brakes. We assume that this braking system is activated only when car speed is not zero, so we choose [1, 160] km/hr as the universe of discourse for car speed v, just for convenience. It is worth noticing that trapezoidal membership functions are used for input membership functions while singleton membership function is used for output membership functions (see Fig. 3.6).

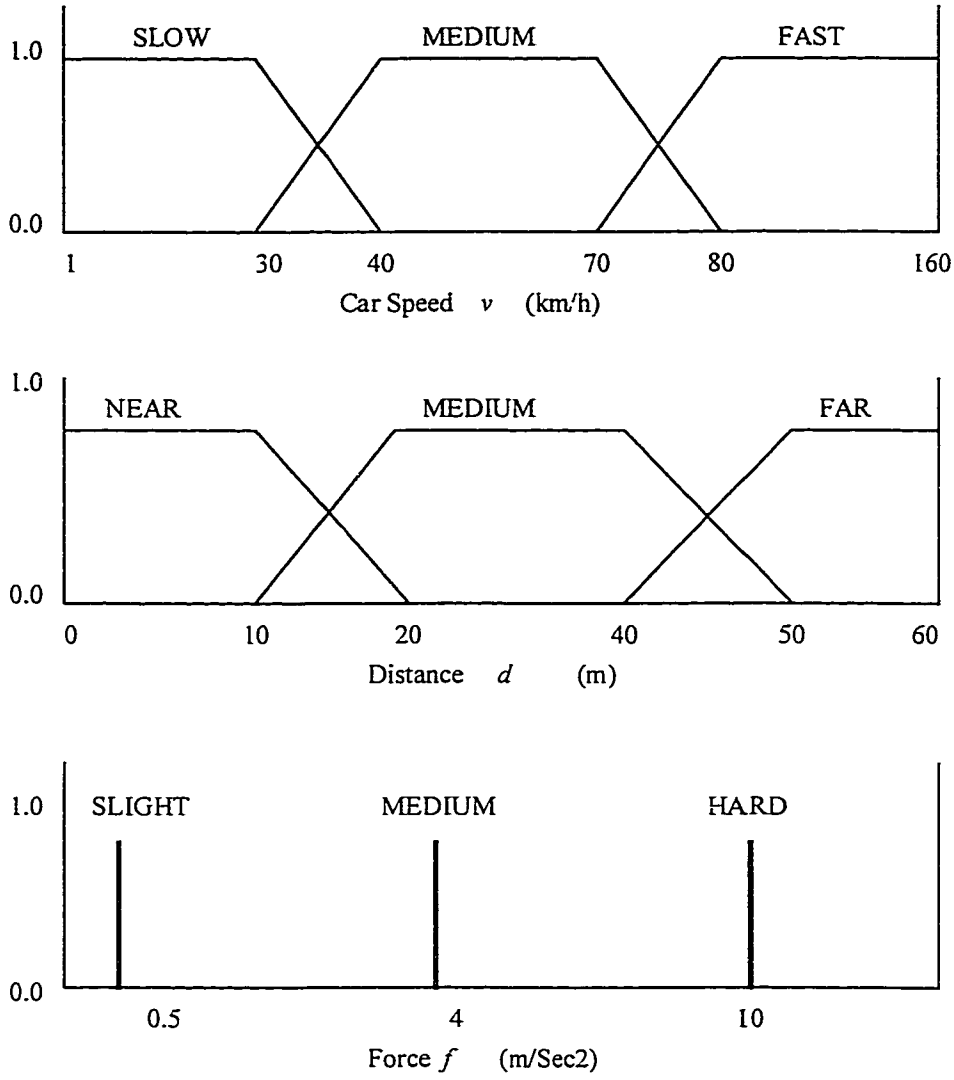


Figure 3.6 Membership functions for the car braking system

For a two-input, one-output system, the rules can be readily represented by the matrix shown in Fig. 3.7.

v \ *d*	NEAR	MEDIUM	FAR
SLOW	1 MEDIUM	2 SLIGHT	3 SLIGHT
MEDIUM	4 HARD	5 MEDIUM	6 SLIGHT
FAST	7 HARD	8 HARD	9 MEDIUM

Figure 3.7 Rule base for the car braking system

In this case, there are 9 rules in the rule base, which correspond to the 9 cells in the shaded area. For example,

Rule #1:

If the *speed is slow* AND *distance is near* then *force is medium.*

Rule #8:

If the *speed is fast* AND *distance is medium* then *force is hard.*

So far, the database and the rule base of FLC have been created. We will demonstrate how the FLC generates crisp outputs given the crisp inputs.

Assume that we have a pair of crisp inputs:

$$v = 72 \quad (km/h)$$

$$d = 11 \quad (m)$$

Let us follow the three key steps to obtain the crisp output.

(1) *Fuzzification*:

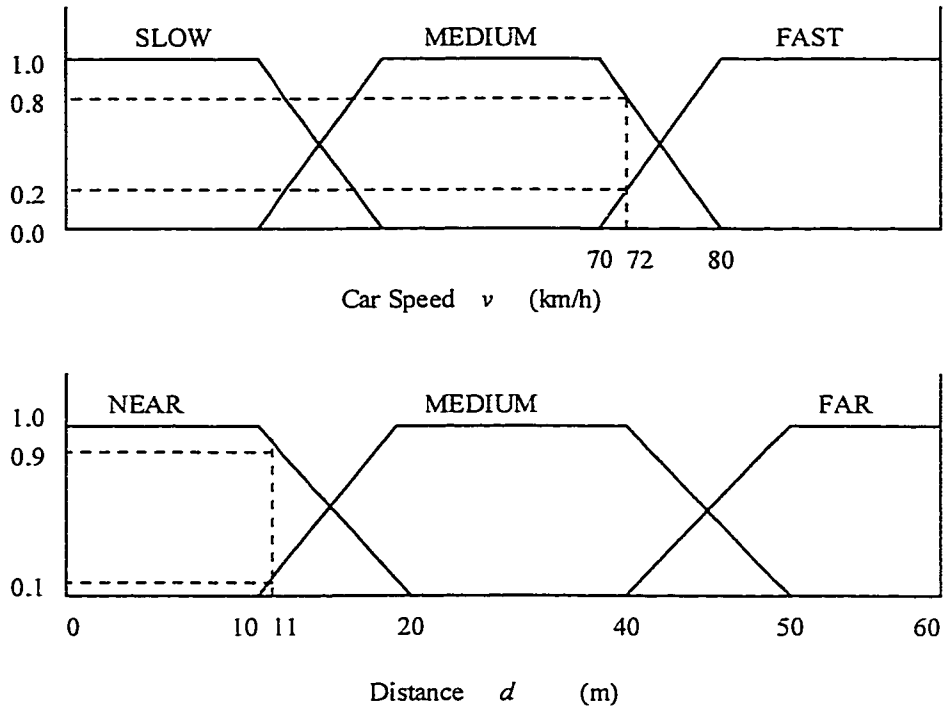


Figure 3.8 Comparison of crisp inputs and the database for car braking system

The results of the comparison are fuzzy inputs:

Crisp input	*Fuzzy input*
$v = 72 \quad (km/h)$	MEDIUM and FAST
$d = 11 \quad (m)$	NEAR and MEDIUM

(2) Fuzzy Inference

In the fuzzification process, we saw how the crisp inputs, car speed and distance, were transformed to fuzzy inputs in the fuzzy braking system. Now we will

see how those inputs are employed in the fuzzy inference process. The fuzzy inference method we will use is the Min-Max method.

The first thing we should do is to evaluate the relevance or degree of membership of each rule's antecedent. To find the relevance of each antecedent, extend a vertical reference line through the crisp input (x-value) and find the y-value where it intersects the membership functions (also see Fig. 3.6). For example, the input of 72km/h-car speed would be found at the intersection of 0.8 of fuzzy set "MEDIUM" and 0.2 of fuzzy set "FAST".

Once the degree of each antecedent has been determined, the next step is to find the degree of truth (rule strength) for each rule. Since Min-Max inference method is used, the rule strength is the smallest strength value of the rule antecedent(s) and the fuzzy output is determined by comparing the rule strengths of all rules that specify the same consequent label.

Referring Fig. 3.7, you can see rule 1, 3 and 4 all dictate the action "HARD" with different rule strengths. When this is the case, the fuzzy output is determined by the maximum rule strength of all the rules involving the same output action. In this example, the maximum of rule strengths for all 3 rules with this output action as a consequent will become the fuzzy output for the output label "HARD". Intuitively, if multiple rules apply for an output action, the one that "most" applies is used. In simpler terms, if two or more rules try to affect the same output, the rule that is most true will dominate.

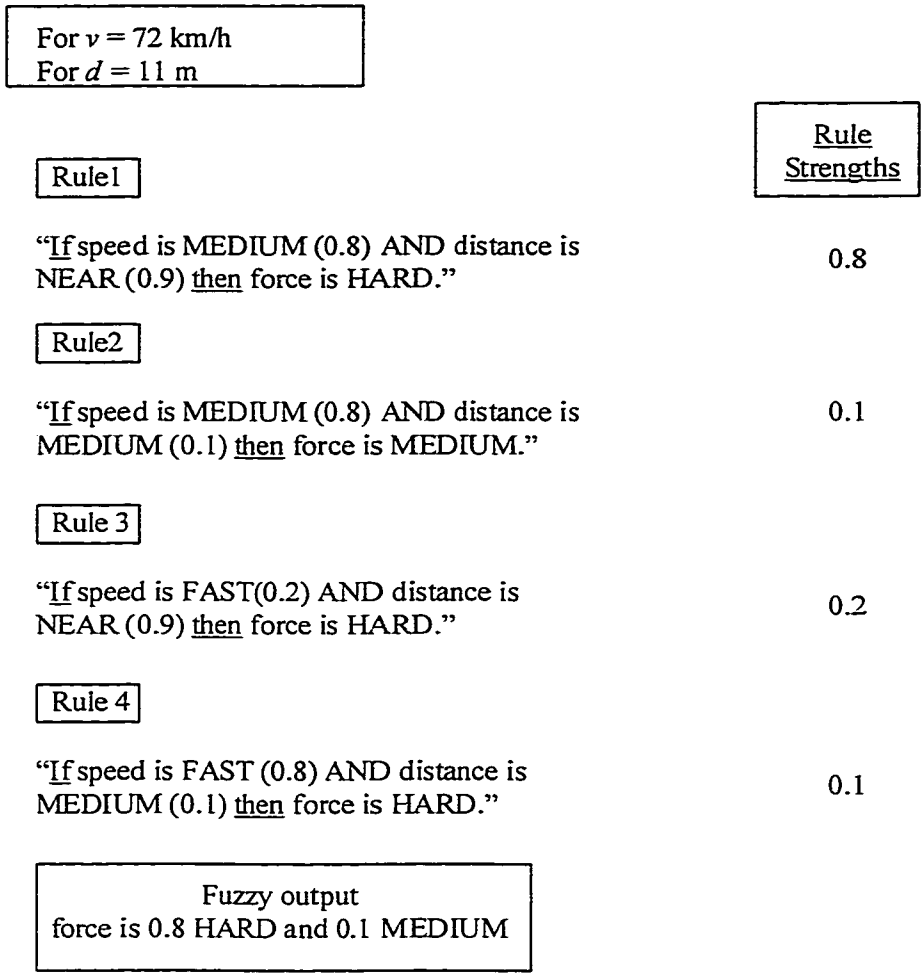


Figure 3.9 Fuzzy Inference for the car braking system

(3) Defuzzification

In defuzzification, all significant fuzzy outputs will be combined into a specific, comprehensive result for that output variable. In this process, all of the fuzzy output values effectively modify their respective output membership functions. As you saw in rule evaluation, by storing the largest rule strength for each consequent, the rules that are most true dominate.

Using the COG defuzzification method, singleton values of outputs are combined using a weighted average. COG formulation for singleton calculation reduces to

$$Y = \frac{\sum (fuzzy\_output_i) \times (\begin{matrix} Singleton & position \\ on & X-axis_i \end{matrix})}{\sum_i (fuzzy\_output_i)} \qquad (3.1)$$

Also note that the singleton defuzzification calculation is significantly less calculation-intensive than the method with non-singleton membership functions.

Let us complete this example by doing COG defuzzification.

$$f = \frac{0.1*4 + 0.8*10}{0.1 + 0.8} = 9.33 \quad m/s^2$$

So $f = 9.33 \quad m/s^2$ should be applied to brakes.

The I/O characteristics of this simple example are shown in Fig. 3.10. A smoother control surface can be achieved by employing more and fine-defined membership functions.

In our example above, the inputs are analog. In the real world, a digital FLC is more popular and practical. This means that sampling and quantization are applied to the FLC. It is apparent that the error caused by quantization may lead to misreaction or even failure of the FLC. To avoid big quantization error, one method is to use high-resolution sensor to collect information. But high-resolution sensors are expensive. A method rather than using expensive equipment is to apply dithering technique to the system. As we discussed in the previous chapter, applying dither can improve the quantization resolution. The performance of dithered digital FLC is of our interest and will be discussed further next.

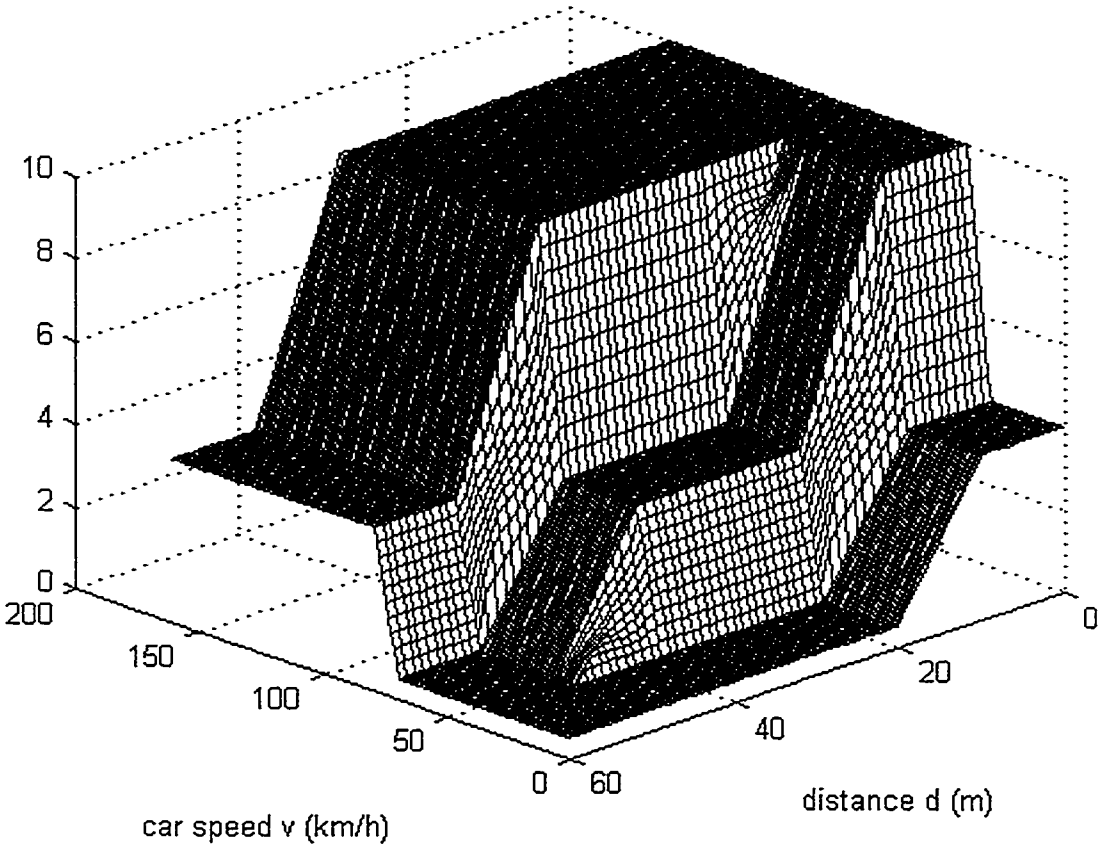


Figure 3.10 I/O characteristics of Analog FLC for car braking system

Chapter 4 Dithering for Fuzzy Logic Control —— Applications

In the previous chapters, we have discussed quantization, dithering technique and FLC, separately.

The objective of this research work is to apply the dithering technique to digital (quantized) FLC and demonstrate the improvement in system performance made by dithering, through system simulation. Simulink /MATLAB provides us a good simulation environment. A brief overview of Simulink is given in the Appendix. Two applications presented in this thesis are (1) using digital FLC to control an inverted pendulum system; (2) using digital FLC to control a truck to back up to a loading dock. In addition to the results in Simulink, the I/O characteristics of analog FLC, digital FLC without dithering and digital FLC with dithering are also displayed and discussed. In the last section of this chapter, a 3-D virtual prototyping environment for fuzzy truck backer-upper has been developed

4.1 Dithered Digital FLC

Digital FLC deals with quantized data. The block diagrams of analog FLC and digital FLC are given in Fig. 4.1.

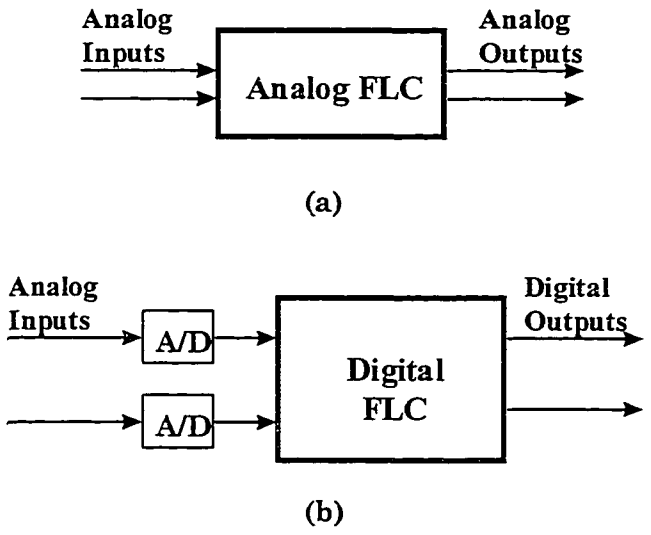


Figure 4.1 Block diagrams of analog and digital FLC's

(a) analog FLC; (b) digital FLC

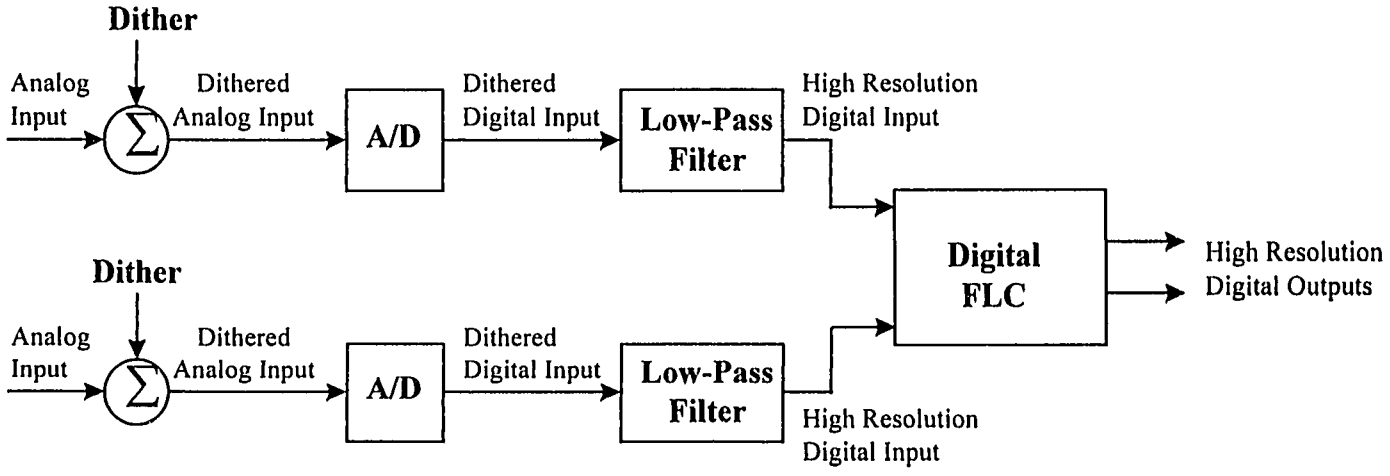


Figure 4.2 Dithered Digital FLC- Solution 1

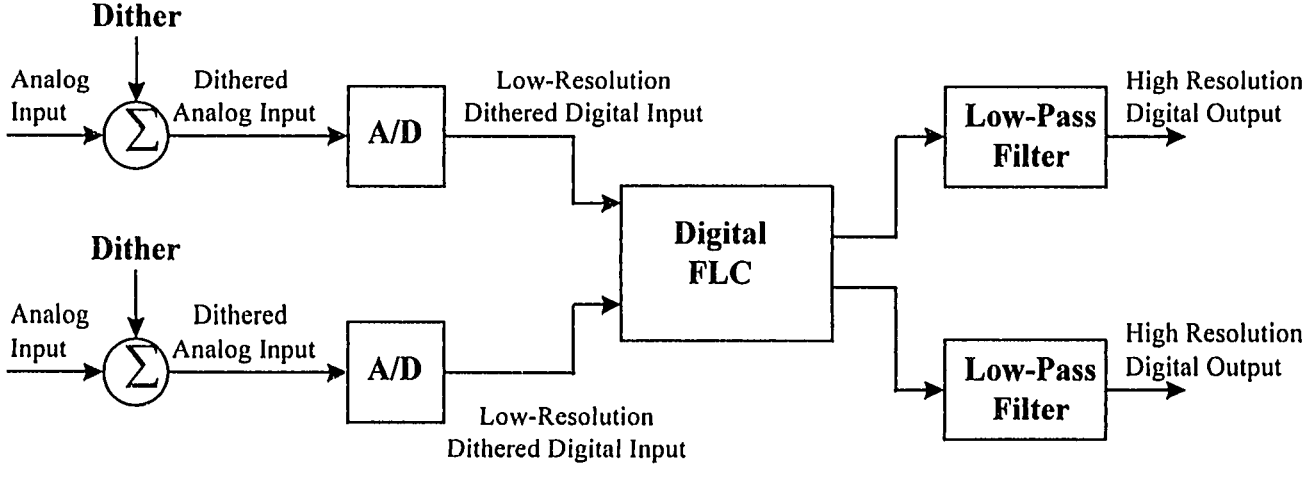


Figure 4.3 Dithered Digital FLC - Solution 2

The A/D conversion of the FLC inputs inevitably causes quantization error. Since we have already discussed the dithering technique to reduce the quantization error, in this chapter, we will apply this technique to digital FLC to validate the effects of dithering on reduction of quantization errors.

There are two solutions of dithered digital FLC, which are described in Fig. 4.2 and Fig. 4.3.

As we mentioned in Chapter 2, in order to achieve a good performance, a dithered quantizer has to be followed by a low-pass filter. Therefore, different position of the low-pass filter can lead to different solution of dithered digital FLC. Solution 1 puts the low-pass filter right behind the dithered quantizer and before the digital FLC, while solution 2 puts the low-pass filter in the end. Compare these two solutions, solution 2 is offering a better performance because the low-pass filter in the end can average and smooth the nonlinearity caused by not only the A/D converter, but the FLC (due to the Min-Max composition rules) as well.

4.2 Inverted Pendulum

4.2.1 Introduction

The inverted pendulum control, also known as the cart-pole problem, is a classical control problem. The system (Fig. 4.4) under control consists of a rigid pole hinged to a cart through a free joint with one degree of freedom. The cart can be moved to its right or left depending on the force exerted on it. The objective is to

balance the pole by moving the cart from side to side. It requires continuous control for all states except for the perfect one that is hard to achieve. The perfect state that does not require control is not maintainable under normal operating conditions, and the slightest perturbation force to the state causes the pendulum to migrate into an unstable region.

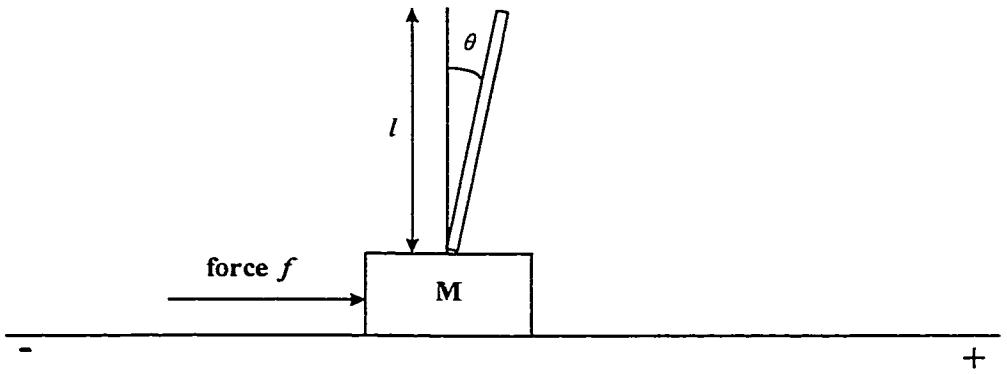


Figure 4.4 Inverted Pendulum

The classical control techniques require a mathematical background that many people may not have or be inclined to deal with. However, as a matter of fact, many of us have had experience with this problem, such as trying to balance a stick in the palm without knowing state space equations. The thing inside is common sense. In the light of this, fuzzy controller was introduced to this problem. We will see later, a few basic intuitive rules are used and tuned to perfect the balancing act.

Typically, the cart is pushed so that the falling rate is reversed and an angle from the vertical that is close to zero is achieved. In terms of parametric values, the desired angle and the desired angular rate are zero. If the initial angle and initial rate are both zero, then the balanced position of the pole is maintained in the absence of any other perturbation. However, if there is a slight perturbation, then

the force of gravity increases the rate and forces the angle to one side or the other of the vertical. If no control action is taken, the pole falls down, and the objective is not achieved. If the starting position is not zero-angle and zero-rate, then control is required immediately.

4.2.2 FLC for Inverted Pendulum

Fuzzy Logic Controller can be employed to control the pendulum system. To do this, first, we should determine how the controller interacts with the system, i.e., what information is passed from system to controller for decision making and what decision is given out by the controller to control the system. This is actually a problem of defining input/output. Since the desired angle and rate both are zero, the state is defined in terms of angle and rate. Thus, the inputs to the controller are two measurements: angle θ and angular rate ω. Obviously, the controller's output should be the force f given to the cart.

Now we need to find out the ranges for input/output of the system. The maximum value of the angle is 90° at which the pole lies flat on the right hand side and the minimum value of the angle is -90° when the pole lies flat on the left hand side. Thus, the universe of discourse for the angle parameter θ is [-90°, 90°]. The angle rate is estimated from the time it takes this pole to fall. Assuming that the pole is 1 meter long, let us say that the time is 1.5 second, which is reasonable. So the average rate is 60°/s. We also know that the speed of the pole slowly increases as it falls down, and its speed begins with zero. Therefore, the maximum rate is

something greater than 60°/s. Let us make a choice of 90°/s, just for convenience. So the universe of discourse for the angle rate ω is [-90°/s, 90°/s].

Second, we should consider how to build a controller that can make appropriate decision upon the inputs. In other words, database and rule base should be created. Last, we should select a defuzzification scheme.

Before defining membership functions, the fuzzy labels are defined below:

NL:	Negative Large
NS:	Negative Small
ZE:	Zero
PS:	Positive Small
PL:	Positive Large

Since our main interest is not how to build a fuzzy logic controller, the procedures of defining membership functions, creating rule base and tuning up the system are skipped, and a fine-tuned system will be given directly. The definitions of membership functions for each input and output are given in Fig. 4.5. Note that we employ only triangular and trapezoidal shaped membership functions for the reason of simplicity. The rule base shown in Fig. 4.6 is based on our thinking relative to what action we will take to balance the pole. All the rules make sense intuitively and will also conform to the control actions and mathematics behind it.

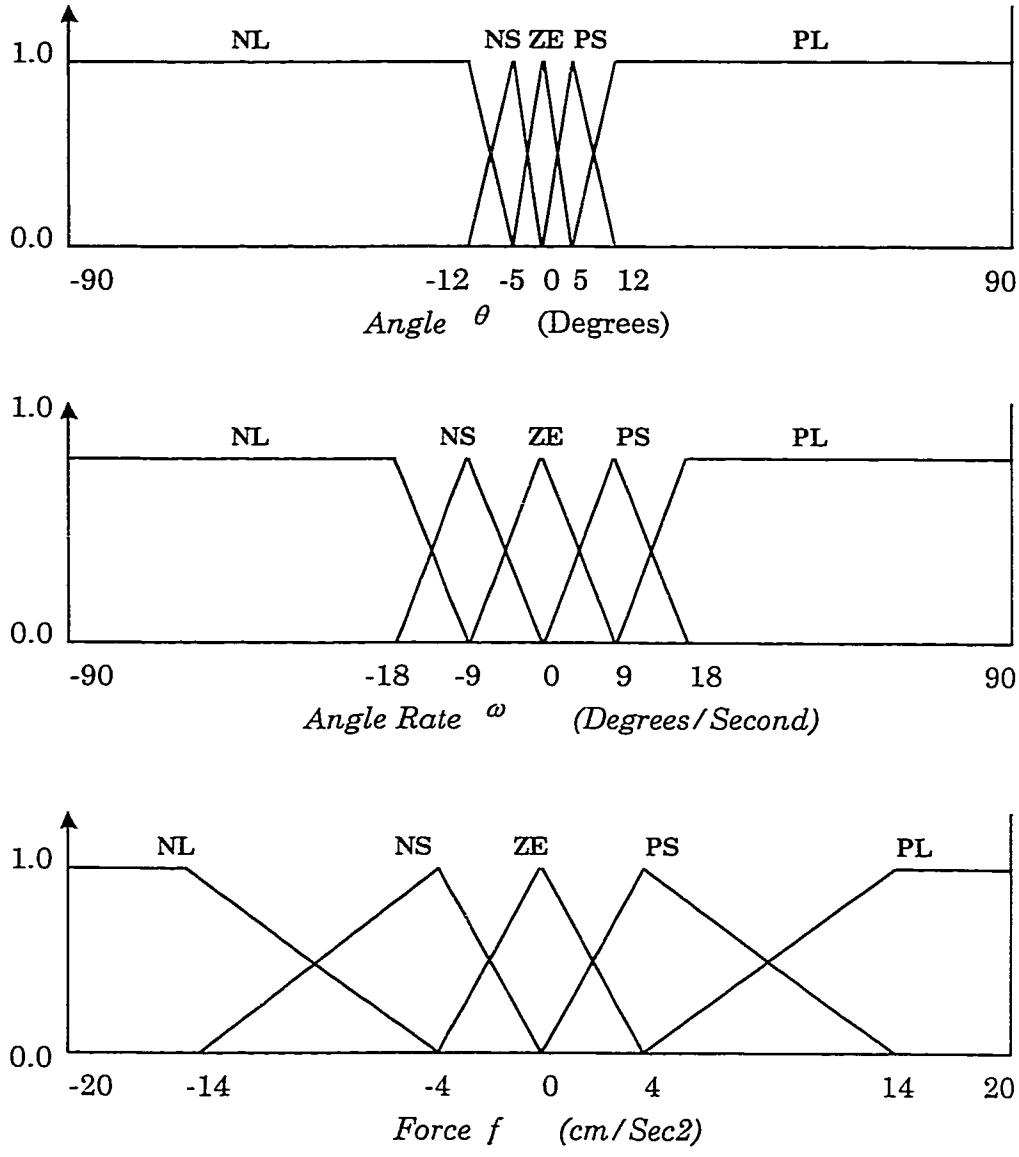


Figure 4.5 Membership functions for inverted pendulum

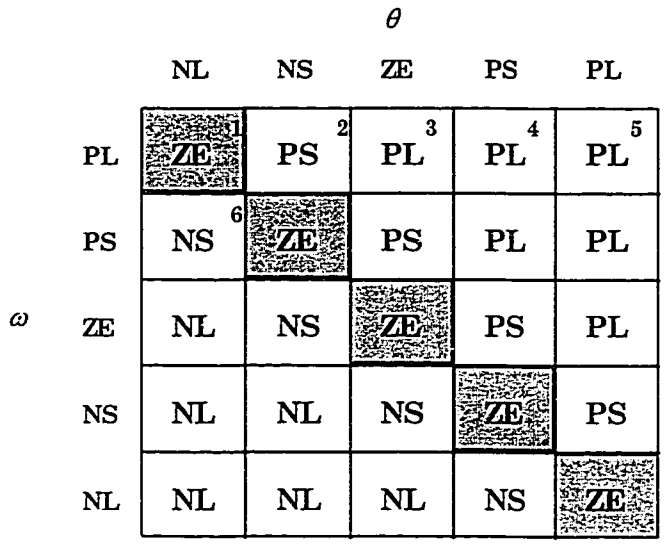


Figure 4.6 Rule base for the inverted pendulum

Now, it is time to select a fuzzy inference scheme and a defuzzification method. In fact, the fuzzy inference scheme must be Mamdani-style, which has already been determined when we defined fuzzy membership functions, instead of *singleton* membership functions, for the output. *Min-Max* is employed for the fuzzy logic operations, and the *centroid* method as shown in equation (3.1) is adopted for defuzzification.

So far, the FLC has been completed.

4.2.3 Simulink Simulation of the FLC for Inverted Pendulum

Simulations for analog system, digital system without dithering and digital system with dithering have been done, respectively, in Simulink/MATLAB. To fully

demonstrate the performances of all the systems, a simulation scenario is designed as follows:

(1) The initial states of the whole system are all-zero states, i.e., $\theta = 0$, $\omega = 0$, $f = 0$. Actually, this is also the perfect state that does not require control.

(2) At $t = 0$, an external force with a very short duration of 0.1 second, is exerted on the cart, which makes the pendulum out of balance.

(3) Right after the external force disappears, the fuzzy controller is triggered to control the pendulum till the next external force is exerted on.

(4) The external force is exerted on the cart every 5 seconds and it lasts 0.1 second each time. The absolute amplitude of the force is $14cm/s^2$.

(5) The overall sampling rate of the digital FLC system is 1000 Hz, i.e., 1 ms per sample.

(6) The simulation stops at $t = 30$ second.

The exertion of the external force is to make the pendulum out of balance, while the FLC should make use of the gap between two consecutive external forces to balance the pendulum. The ultimate goal of a desirable FLC is to achieve the perfect state, on which the rule base is based. Therefore, our standard to examine the system performances becomes the examination of system states during the runtime of the simulation. More specifically speaking, we need to examine the θ vs t curve during the 30 seconds, as well as ω.

The block diagrams of analog system, digital system without dithering and digital system with dithering are given in Fig. 4.7, Fig. 4.8 and Fig. 4.9, respectively.

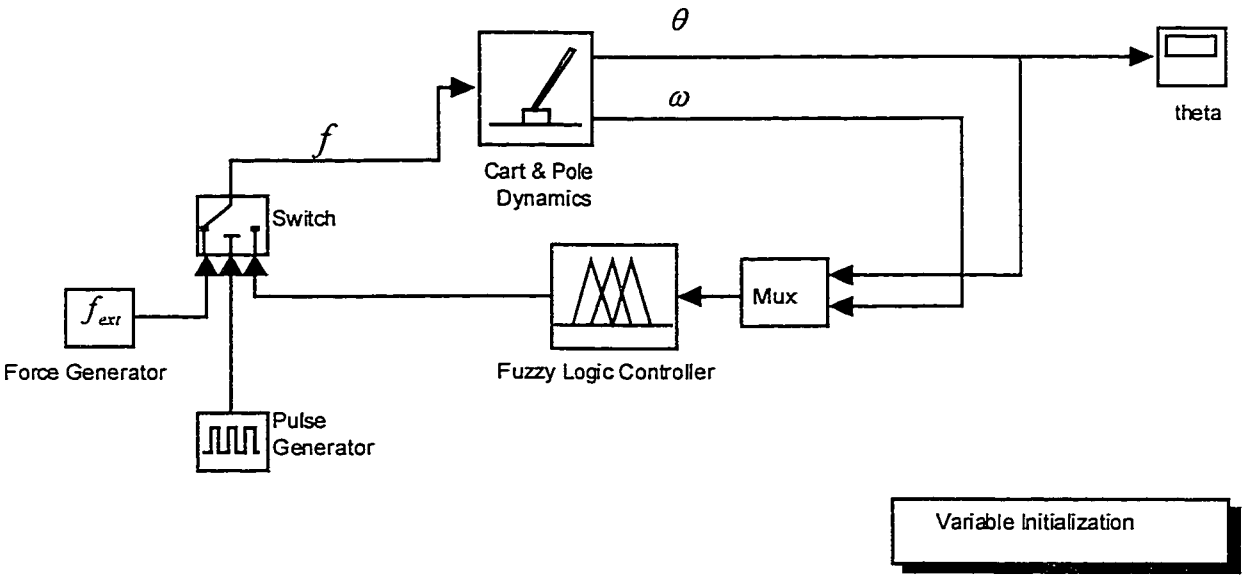


Figure 4.7 Analog FLC for inverted pendulum system

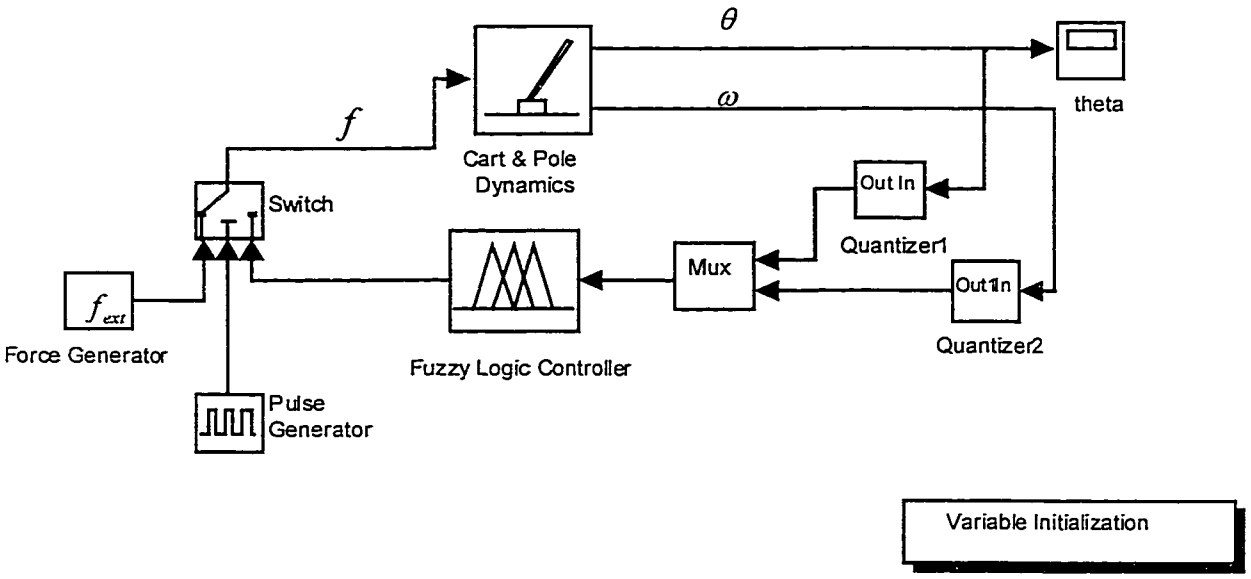


Figure 4.8 Digital FLC for inverted pendulum system without dithering

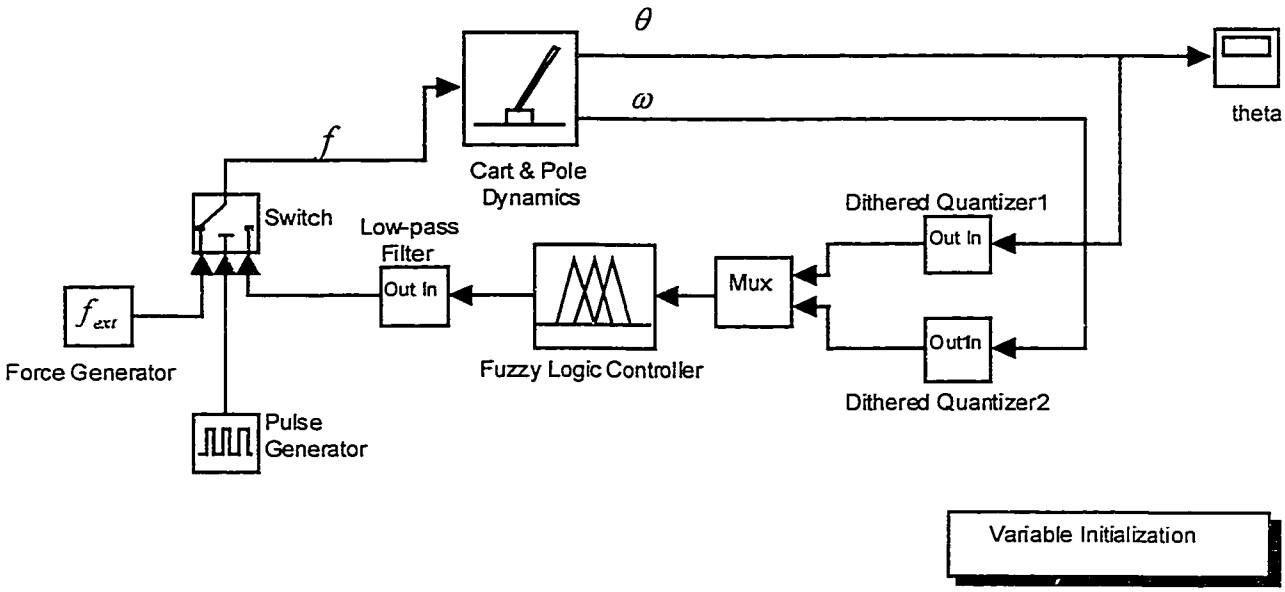


Figure 4.9 Digital FLC for inverted pendulum system with dithering

It should be noted that the *Cart & Pole Dynamics* block is the kinematics model of the pendulum. Given the current states of the pendulum, i.e., θ, ω and the force f exerted on the cart, this model's outputs are a set of new states, θ' and ω'

A state equation is given in [23]. We can obtain the following equation by ignoring the friction.

$$\ddot{\theta} = \frac{g\sin\theta + \cos\theta\left[\dfrac{-f - ml\omega^2}{M+m}\right]}{l\left[\dfrac{4}{3} - \dfrac{m\cos\theta^2}{M+m}\right]} \qquad (4.1)$$

where $\ddot{\theta}$ denotes the second order derivative of θ. g is the gravity acceleration. m is the mass of the pole and M is the mass of the cart. l is the length of the pole. The *Cart & Pole Dynamics* block is actually a model of this equation.

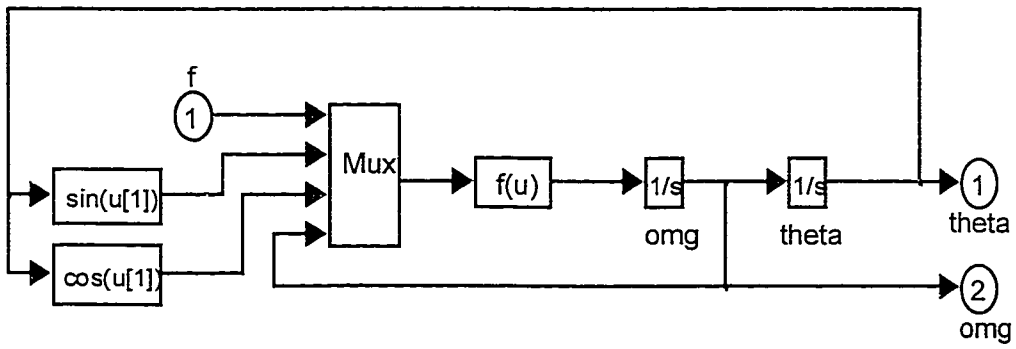


Figure 4.10 Structure of Cart & Pole Dynamics block

You may find that there is no block like *Sampler* even in the digital systems shown in Fig. 4.8 and Fig. 4.9. The reason is that Simulink provides a connection named *In*, which is used as an input port and allows the user to configure the sample time for it. Therefore, what we need to do is just to integrate this connection and configure it as required, in the function model that we define. For instance, in the *Cart & Pole Dynamics* block, we put such a connection and configure its sample time to 0.01s. For more examples, see Fig. 4.11 and Fig. 4.12.

The *Fuzzy Logic Controller* blocks are the same in all three systems. They have the same databases and rule bases. The differences of analog/digital, with/without dithering can be told from the FLC's I/O ports. In Fig. 4.8, there are two *quantizers* at the input side of the FLC; while in Fig. 4.9, there are two *dithered quantizers* at the input side and one *low-pass filter* at the output side of the FLC.

The *Quantizer* blocks in Fig. 4.8 have the structure of Fig. 4.11. The quantization step can be specified in the Quantizer.

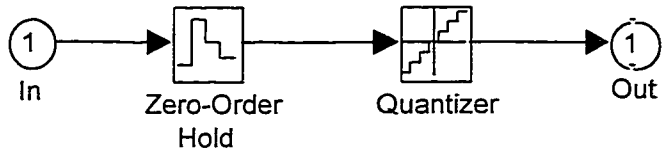


Figure 4.11 Structure of a common quantizer

Note that the *Dithered Quantizer* blocks in Fig. 4.9 should have a dither signal added to the original signal before quantization, a dithered quantizer is modeled in Fig. 4.12. This is one of the most important blocks in our simulations.

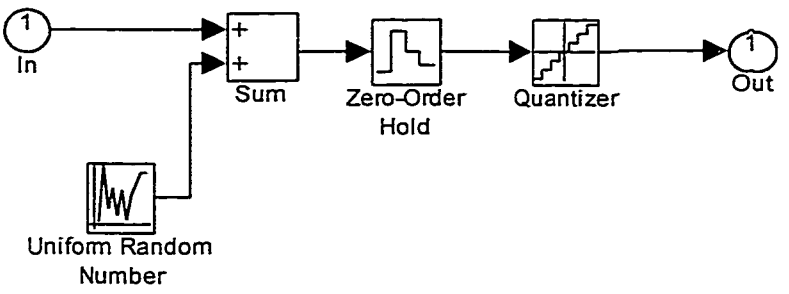


Figure 4.12 Structure of uniform-dithered quantizer

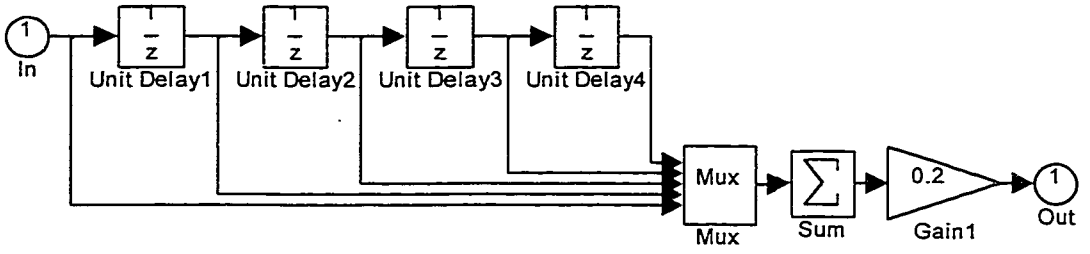


Figure 4.13 Structure of a moving average filter

Another important function block, in the case of adopting dithering technique, is the *Low-pass Filter* block. We choose a moving average filter for a low passing purpose and model it in Simulink. Fig. 4.11 shows a 5-unit moving average filter as an example. If more units are needed, simple modification on this model will be fine.

The *Force Generator* block in the system model is also a user-defined block. It is to generate required external force for the simulation. Different external forces have been experimented and we adopted an external force with exertion period of 5 seconds, within which an analog FLC can just resume the balance of the pendulum. Moreover, the amplitude and the duration of the exertion each time guarantee that the pendulum will not fall down on the cart, in which case the pendulum cannot be in balance again by just moving the cart. The structure of the *Force Generator* is given in Fig. 4.14.

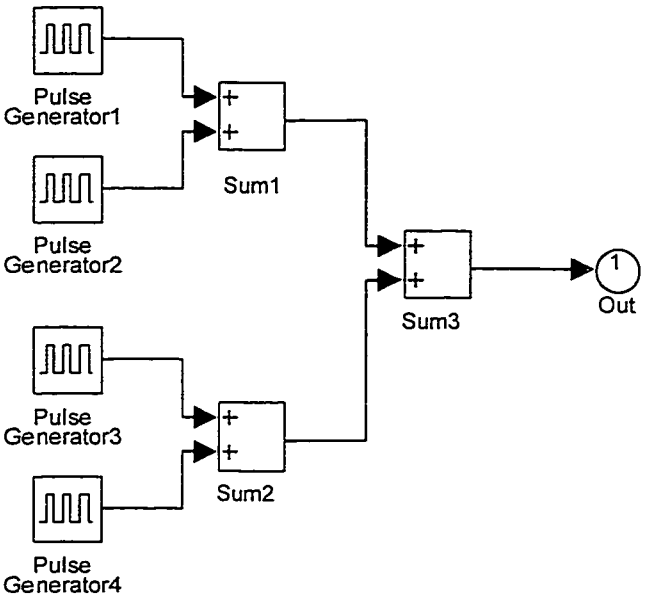


Figure 4.14 Structure of Force Generator

The waveform of the external force is given in Fig. 4.15. As we mentioned before, an analog FLC can just react to balance the pendulum within the gap of two consecutive external disturbances. Fig. 4.16 is telling the same story. Now, we need to examine the influence of quantization. Fig. 4.17 is the simulation result for an

FLC system adopted a 4-bit quantization. We can tell that the pendulum can never be balanced in this situation. Between every two consecutive external force pulses, θ is kept at the level of $\pm 0.025 rad$ most of the time, instead of 0. This is apparently caused by quantization.

The use of applying dithering technique can be found in Fig. 4.18 - Fig. 4.20. Simulations of systems employing uniform dithering, Gaussian dithering and Triangle dithering have been done, and the results are shown in those three figures. Between the two consecutive external force pulses, the curves of θ in Fig. 4.18 and Fig. 4.19 are approaching 0, although they are not as smooth as the curve in Fig. 4.16. Fig. 4.20 shows a poorer balancing process, because θ holds a level of $\pm 0.01 rad$ most of the time, instead of approaching 0. However, the system with triangle dithering outperforms the system without dithering since it has a lower level of θ.

To compare the performances of these three systems to that of the analog system, we introduce the *absolute error*. Choose the θ curve of analog system as the standard curve, the absolute error means the absolute difference between the standard θ curve and that of other systems. Consider the quantized system without dithering, quantized systems with uniform, Gaussian and triangle dithering. The absolute errors of these four systems are given in Fig. 4.21.

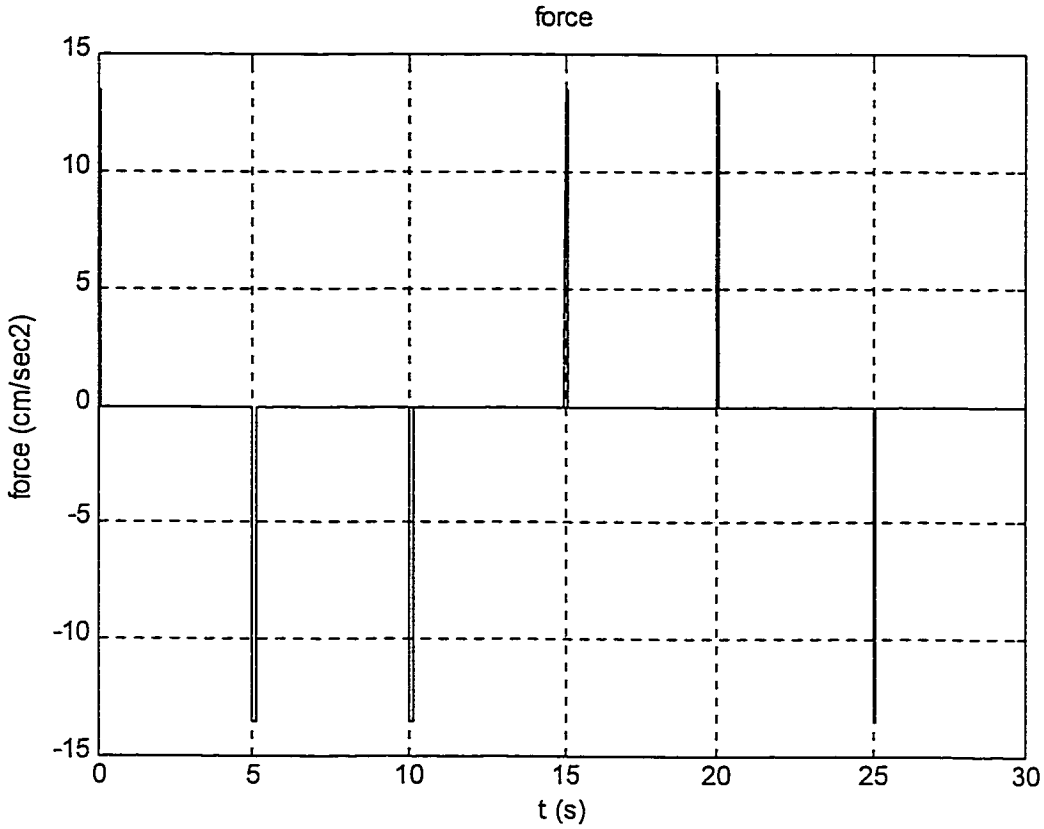


Figure 4.15 External force exerted on the cart

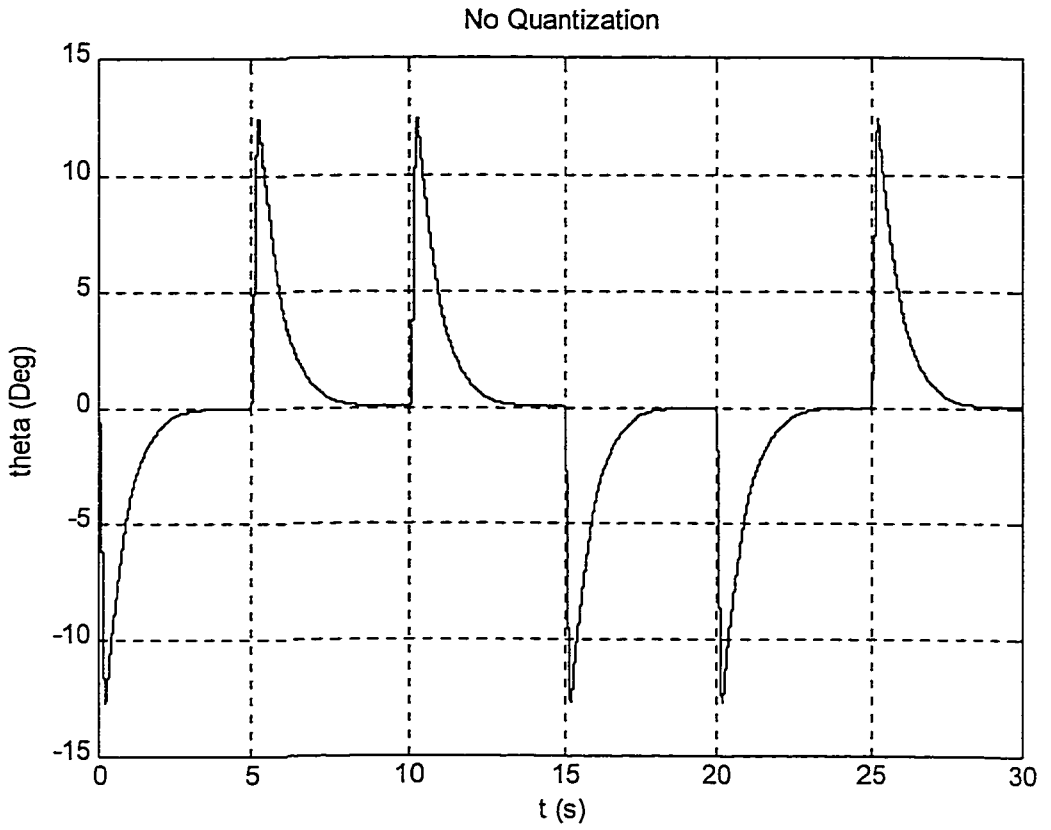


Figure 4.16 θ of the analog FLC pendulum system

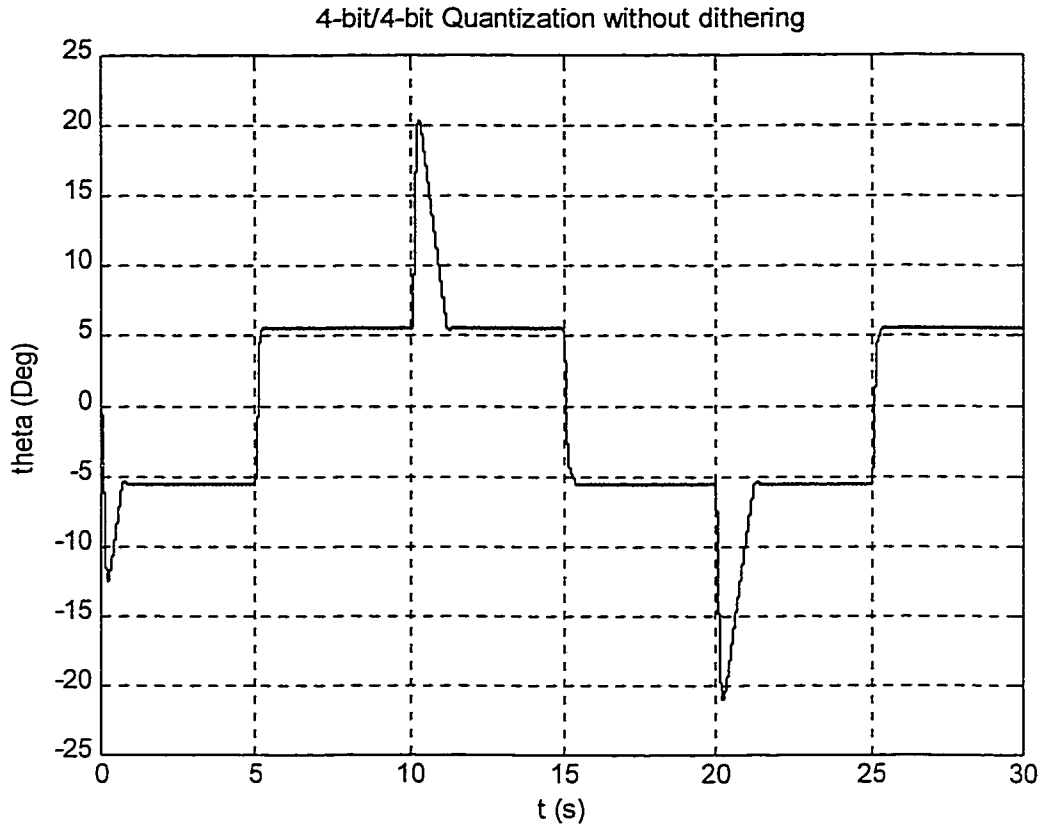


Figure 4.17 θ of the FLC pendulum system with 4-bit/4-bit quantization without dithering

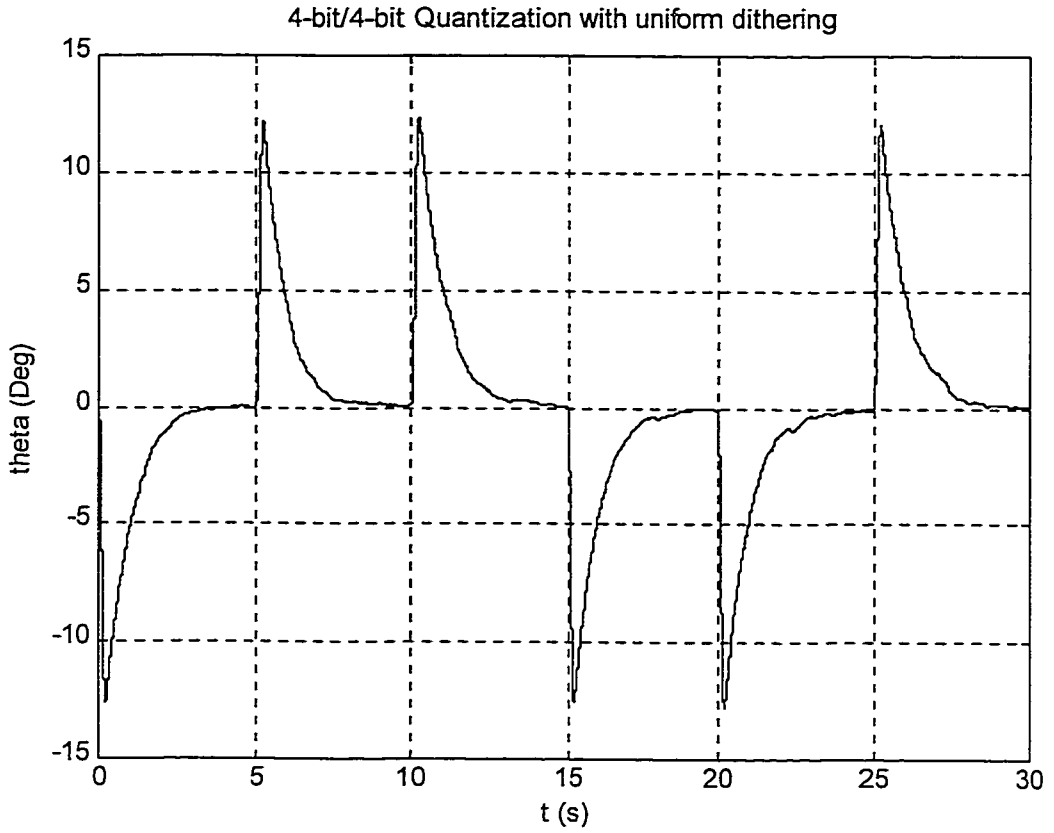


Figure 4.18 θ of the FLC pendulum systemwith 4-bit/4-bit quantization and uniform dithering

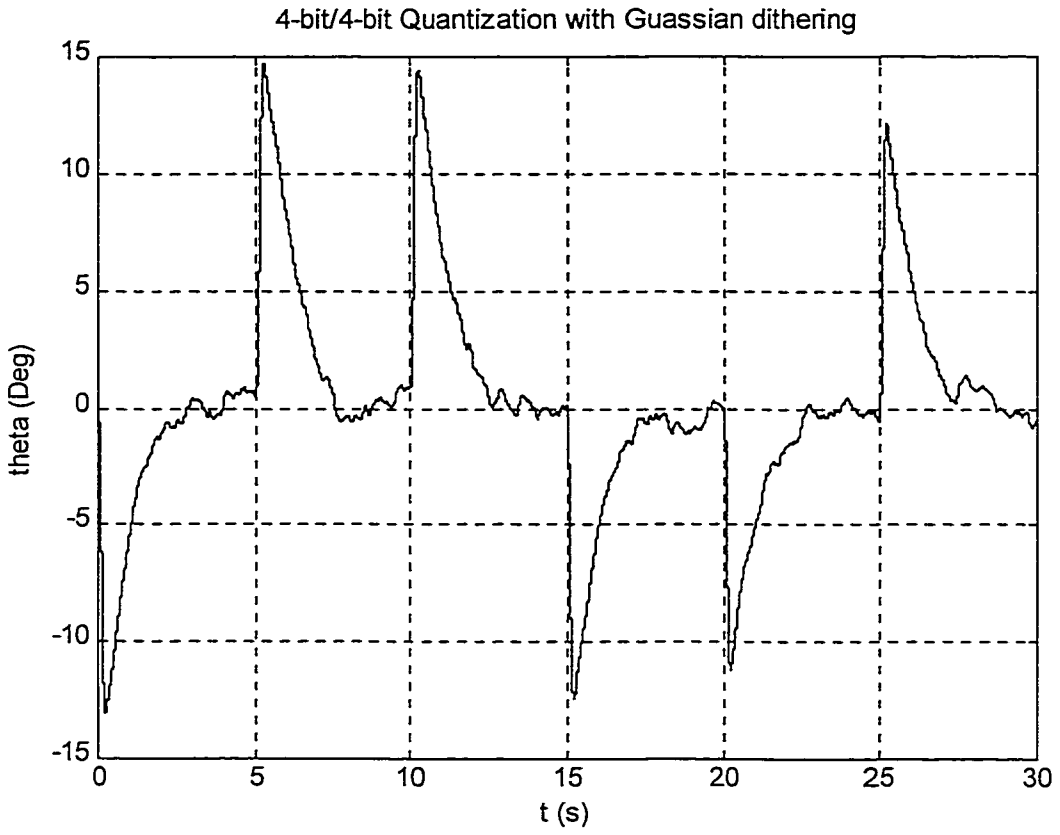


Figure 4.19 θ of the FLC pendulum systemwith 4-bit/4-bit quantization and Gaussian dithering

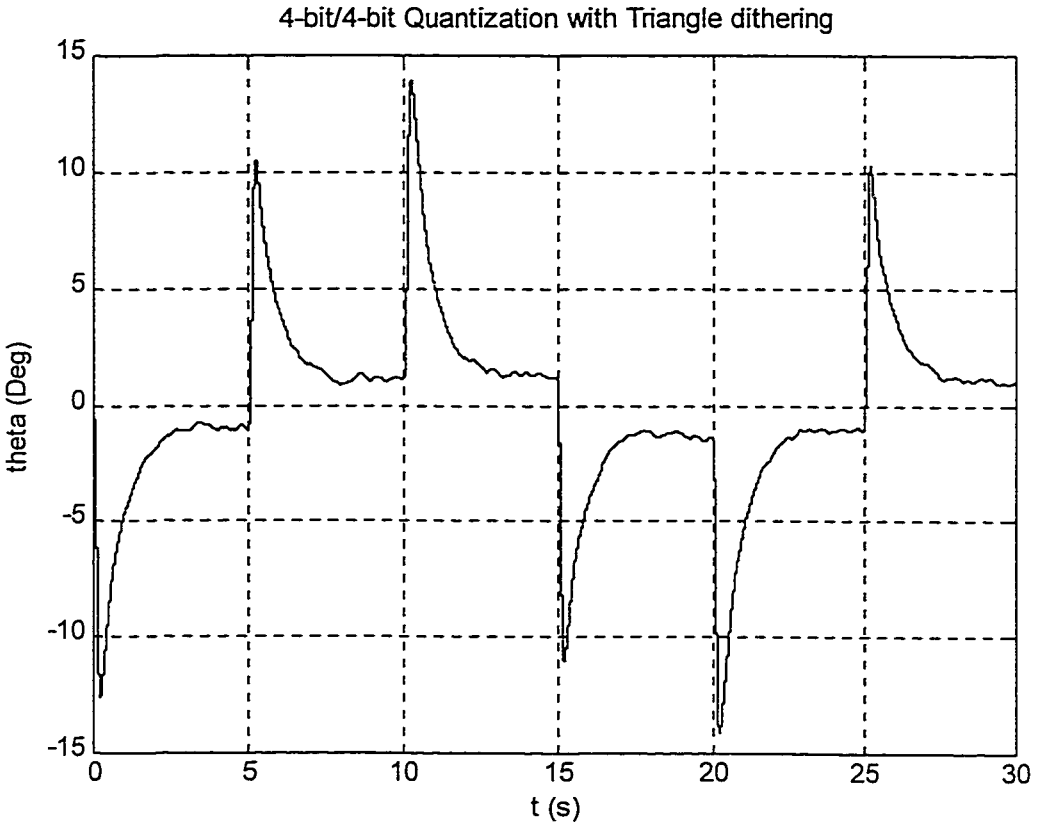


Figure 4.20 θ of the FLC pendulum systemwith 4-bit/4-bit quantization and Triangle dithering

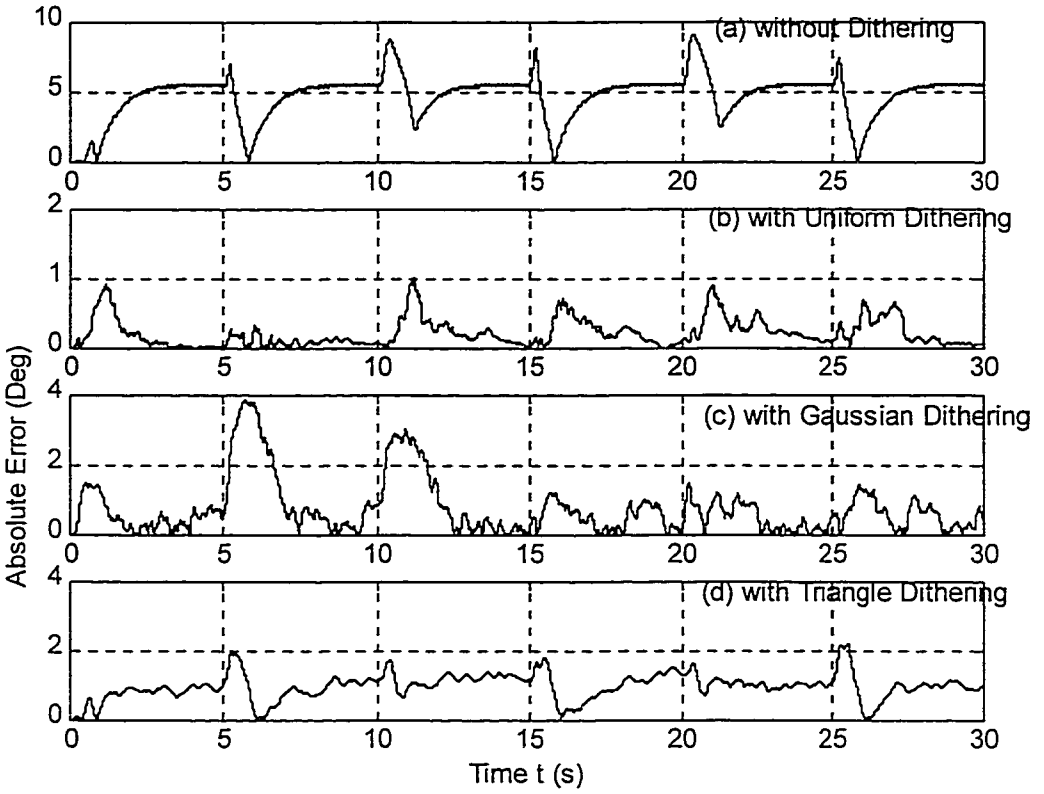


Figure 4.21 Comparison of four systems (a) quantized system without dithering; (b) quantized system with uniform dithering; (c) quantized system with Gaussian dithering; (d) quantized system with triangle dithering.

To demonstrate that a low-resolution digital FLC with dithering can achieve the performance of a high-resolution digital FLC system without dithering, we simulate an 8-bit/8-bit digital FLC for inverted pendulum. The result is shown in Fig. 4.22. Between any two adjacent exertions of external force, the θ curve reaches a stable status at approximately $\theta = 0.024rad$, instead of $\theta \approx 0$.

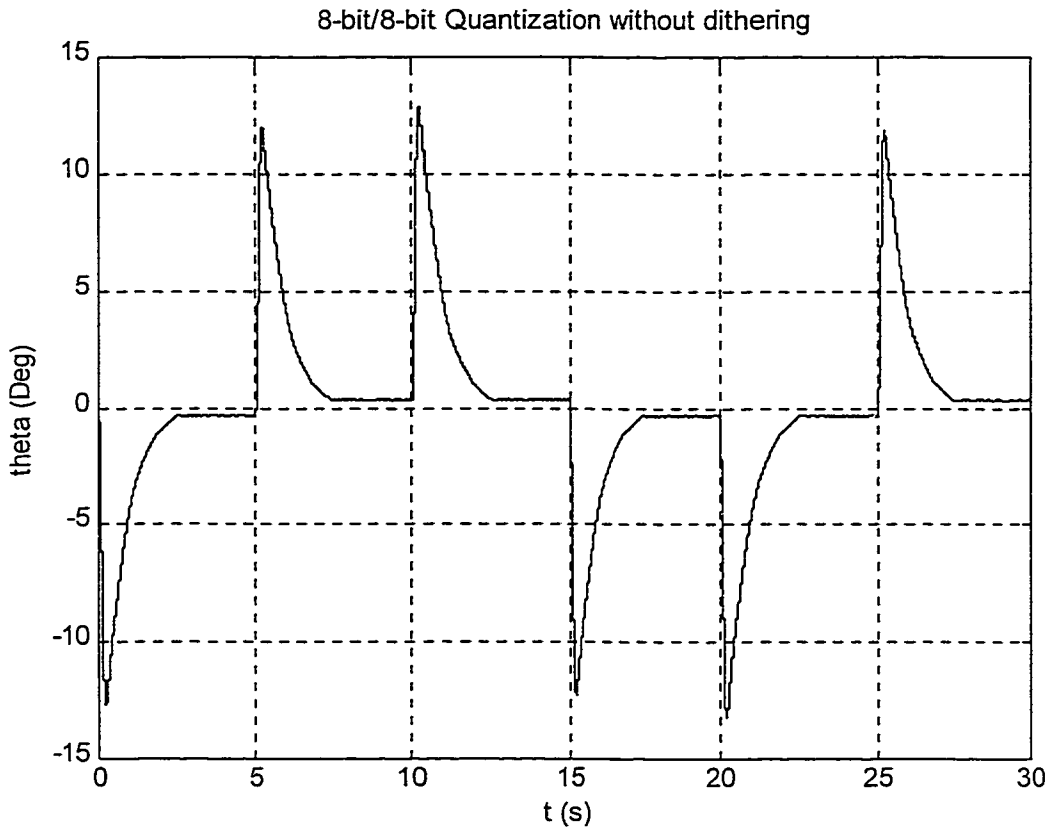


Figure 4.22 θ of the FLC pendulum systemwith 8-bit quantization without dithering

The results shown above can lead us to the following conclusions. Applying dithering technique does reduce the effect of quantization error and improve the performance of a digital FLC. Applying different forms of dither signal would result in different improvement. In our case, the uniform dither is the best one.

4.2.4 Analysis of the I/O Characteristics

To have a big picture of the FLC, we need to obtain the I/O characteristic surface. It is a three dimensional control surface showing the force outputs f that correspond to all combination of the two input state variable θ and ω. A smooth control surface reflects a fine controller. Since we have obtained simulation results in Simulink, let us analyze them from the view of I/O characteristics.

The I/O characteristics of analog FLC for inverted pendulum system are shown in Fig. 4.23. As to the digital FLC, different sensors are considered. As we know, high-resolution sensor decreases the quantization error, but requires longer conversion time and more money; while low-resolution sensor makes big quantization error, but needs less time and money. Therefore, we will show the I/O characteristics of 4-bit/6-bit digital systems, with or without dithering. They are shown in Fig. 4.24/25 and Fig. 4.27/28 respectively. The differences between Fig. 4.23 and Fig. 4.24/25 are shown in Fig. 4.26. Note that we employ uniform dithering, since it is the best in this specific application.

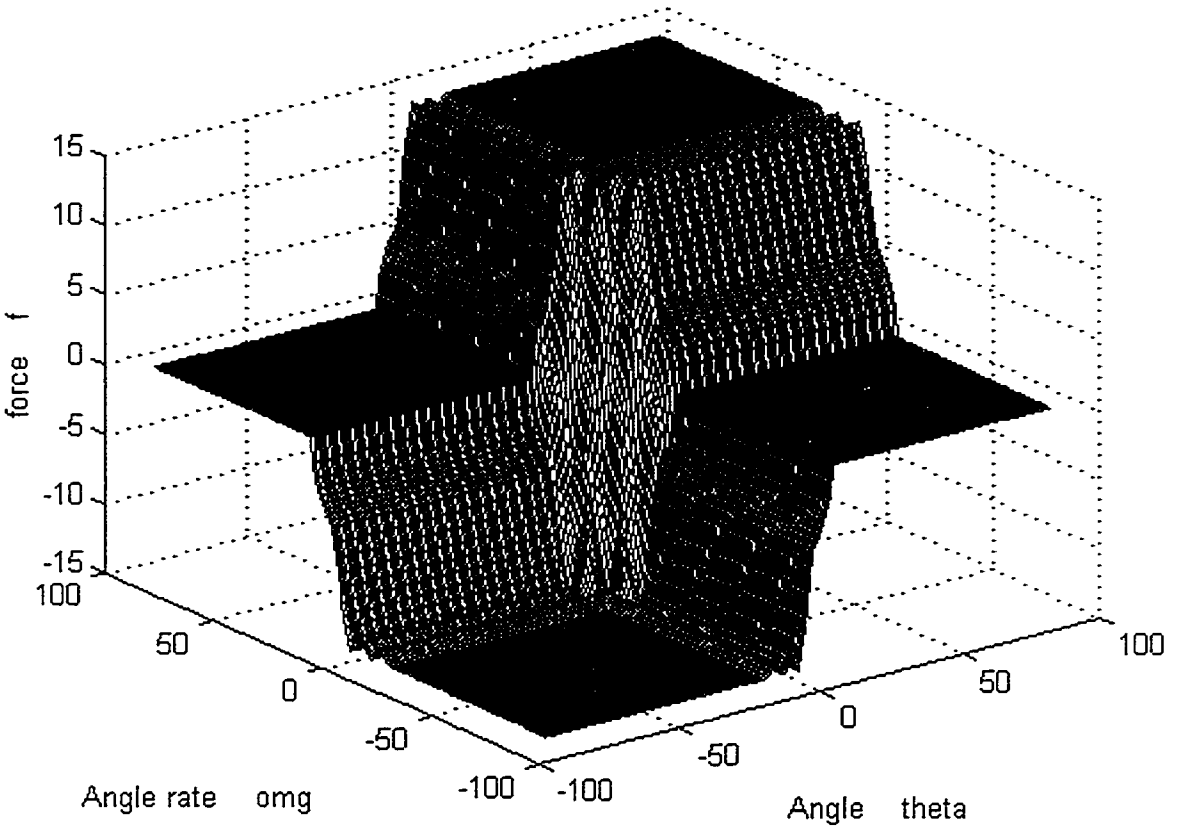


Figure 4.23 I/O characteristics of the analog FLC for inverted pendulum

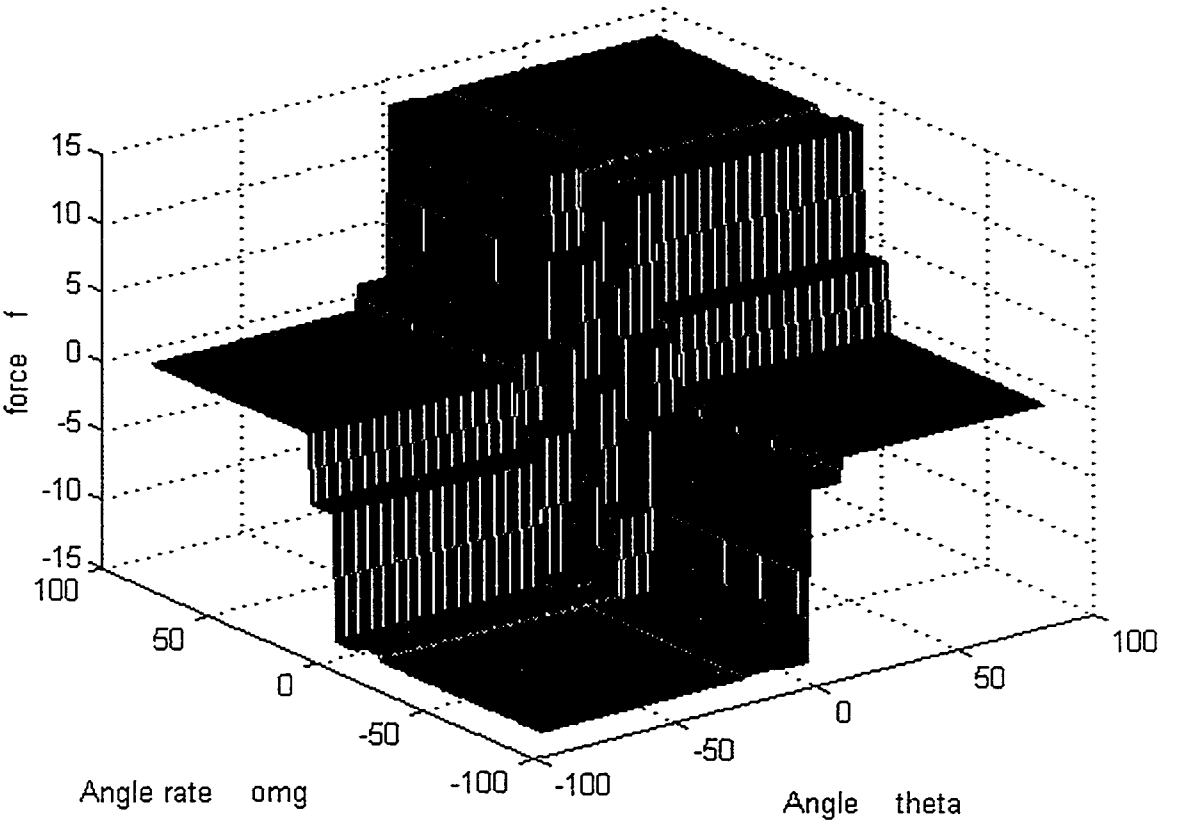


Figure 4.24 I/O characteristics of the FLC for inverted pendulum

4-bit Quantization **is applied to both inputs; without dithering**

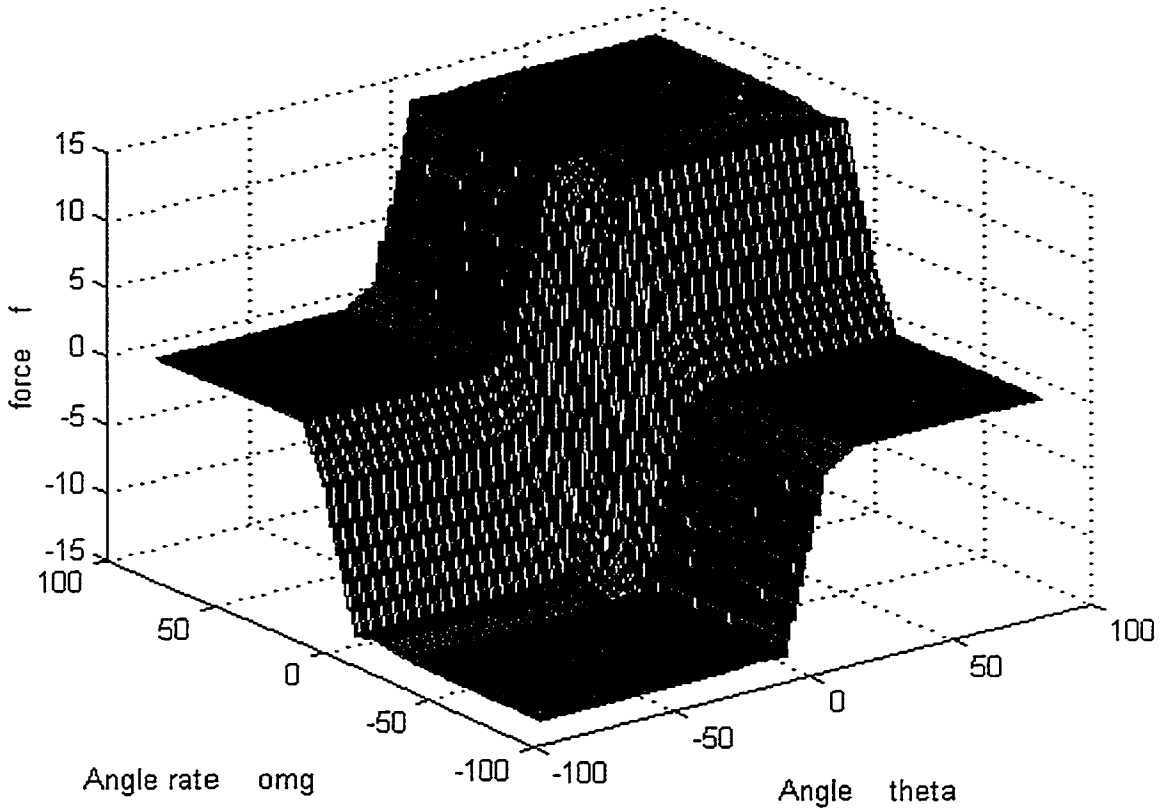


Figure 4.25 I/O characteristics of the FLC for inverted pendulum

4-bit Quantization **is applied to both inputs; with uniform dithering**

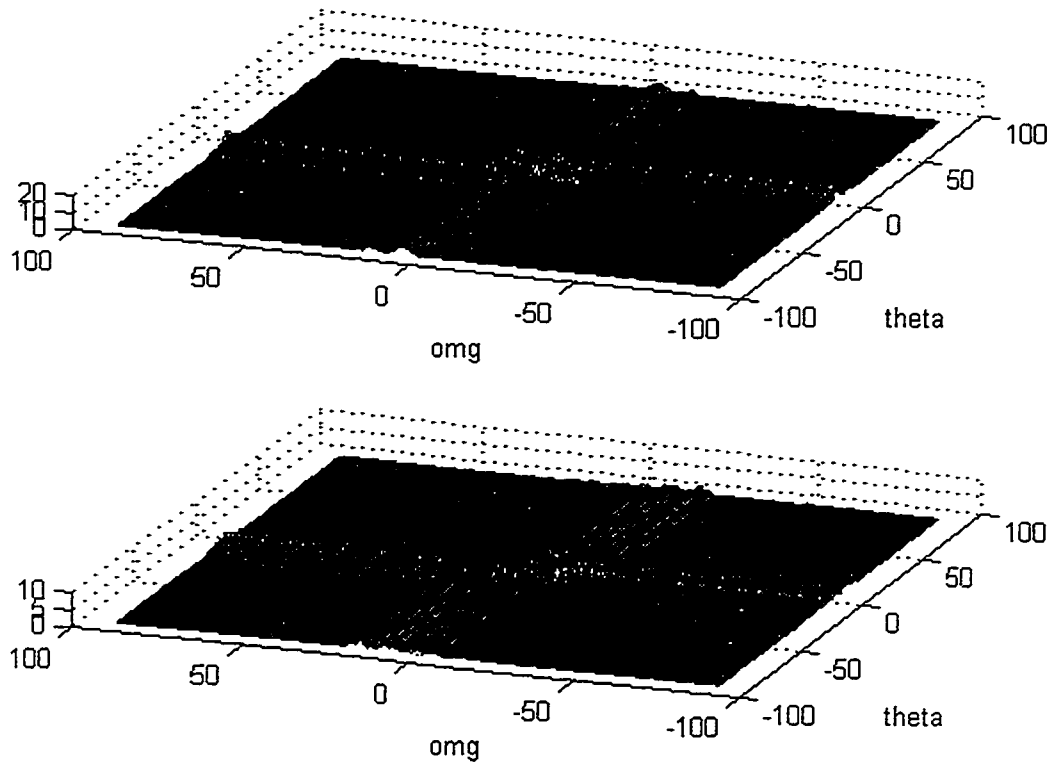


Figure 4.26 Differences in the FLC's output *f*

Upper: difference in the case of 4-bit/4-bit inputs relative to analog inputs

Lower: differences in the case of dithered 4-bit/4-bit relative to analog inputs

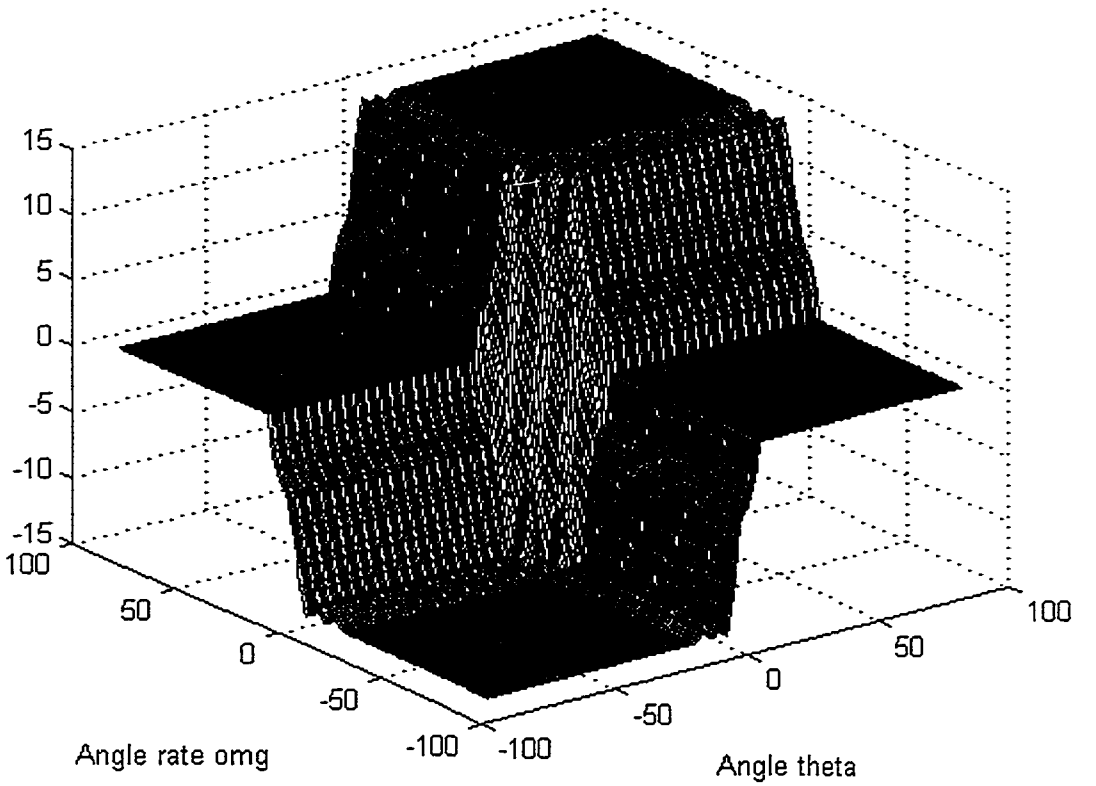


Figure 4.27 I/O characteristics of the FLC for inverted pendulum

8-bit Quantization **is applied to both inputs, without dithering**

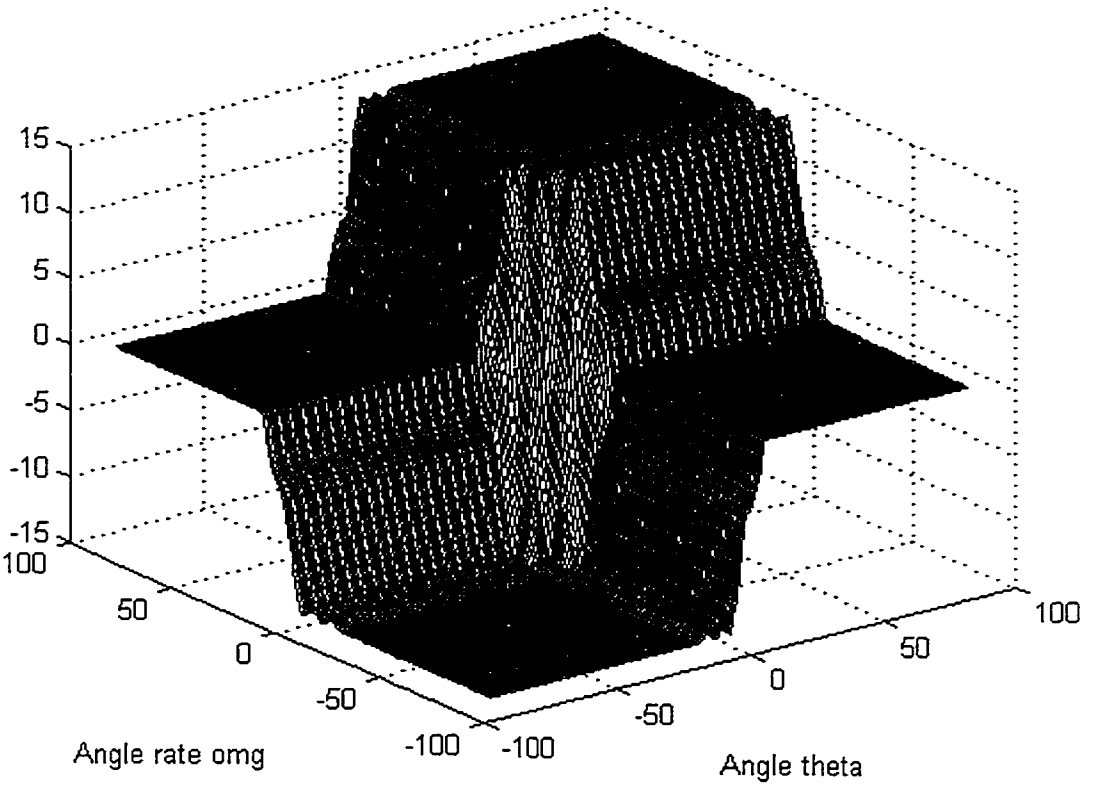


Figure 4.28 I/O characteristics of the FLC for inverted pendulum

8-bit Quantization **is applied to both inputs; with uniform dithering**

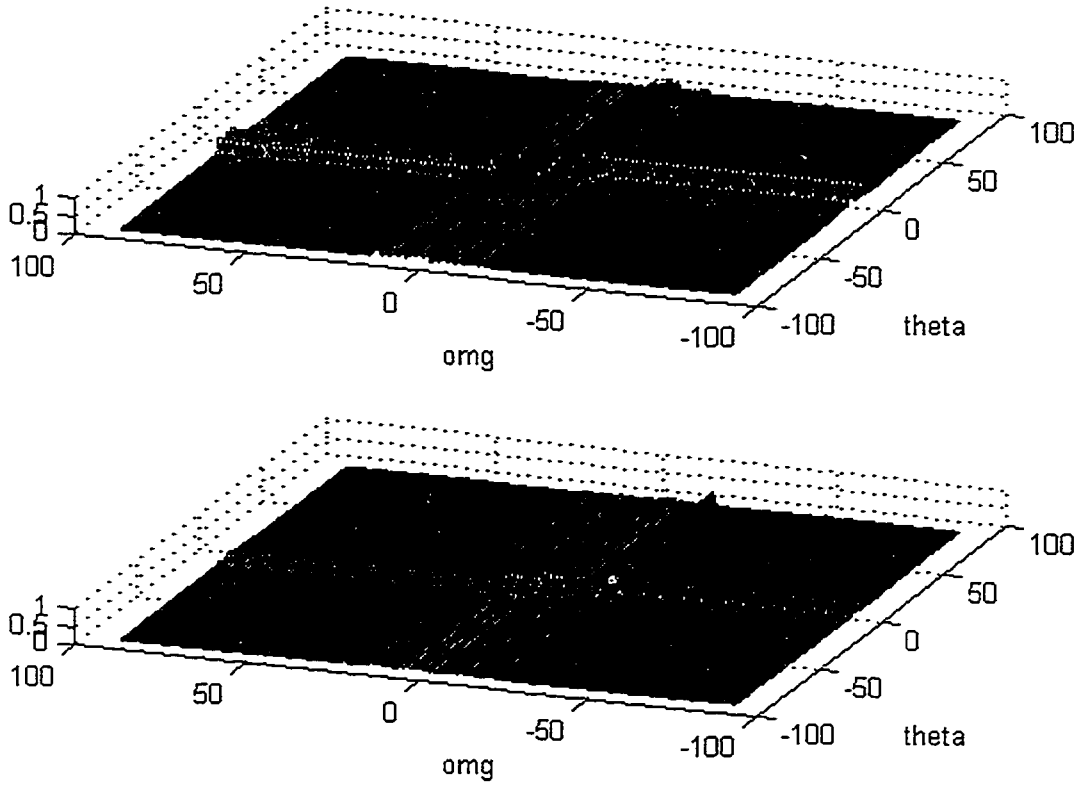


Figure 4.29 Differences in the FLC's output *f*

Upper: difference in the case of 8-bit/8-bit inputs relative to analog inputs

Lower: differences in the case of dithered 8-bit/8-bit relative to analog inputs

Apparently, the I/O characteristics of 4-bit/without dithering system have sharp changes in many areas, which means that tiny difference in the inputs may result in a jump of the output. This is not what we expect for an FLC. When dithering is applied to quantization, the I/O characteristics we got in Fig. 4.25 are much smoother than Fig. 4.24. It is even close to the I/O characteristics in Fig. 4.23. This can be found through Fig. 4.26, the error surfaces. The maximum absolute error is 11.9966 cm/s^2 in the case of no dithering, 5.0315 cm/s^2 in the case of with dithering. This shows the improvement made by dithering in a way. The results for 8-bit inputs shown in Fig. 4.27 and Fig. 4.28 illustrate the same point. However, noticing the little difference between Fig. 4.27 and Fig. 4.28, we compare the error surface of both in Fig. 4.29. The maximum absolute error is 0.8768 cm/s^2 in the case of no dithering, and 0.7135 cm/s^2 in the case of with dithering. The errors shown in Fig. 4.29 are much smaller than those shown in Fig. 4.26. However, the improvement in this case is not so much as that in the case of low-resolution inputs (e.g. 4-bit).

These results conform to the results obtained from simulations, and both prove that dithering can improve performance of the digital FLC, especially for the low-resolution inputs (e.g., 4-bit). For high-resolution inputs (e.g., 8-bit) the dithering efficiency is not so obvious.

4.3 Truck Backer-Upper

4.3.1 Introduction

It is well known that to back up a truck to the loading dock is a difficult exercise for all but skilled truck drivers. This truck backer-upper problem has become a standard problem in the fuzzy logic field. The problem is to design a controller (driver) that can back up a truck into a loading dock from any initial position that has enough clearance from the back wall. Only backing up is allowed, i.e., there is no going forward. Fig. 4.30 shows the simulated truck and loading dock.

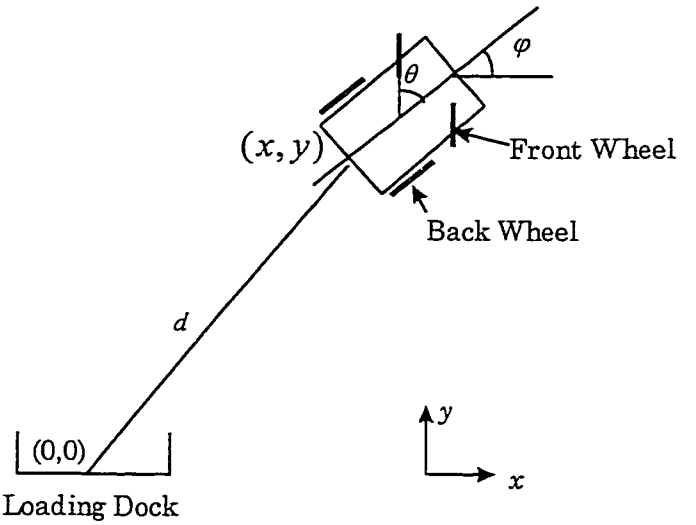


Figure 4.30 Diagram of simulated truck and loading dock

The three state variables φ, x and y exactly determine the truck position. φ specifies the angle of the truck with the horizontal. The coordinate pair (x, y) specifies the position of the rear center of the truck in the plane. θ is the steering angle, which is the very variable controlled by human driver or a controller to achieve the goal. Consider a general case that the truck is back-wheel driving. Therefore, front wheels are for steering while the back wheels are only for driving.

The goal is to make the truck arrive at the loading dock at a right angle ($\varphi = \frac{\pi}{2}$) and to align the position (x, y) of the truck with the desired loading dock (0,0).

An analog FLC has already been developed for this problem in MATLAB/Fuzzy Logic Toolbox. And the FLC does perform very well in backing up a truck. However, after careful investigation, we found that it is not an FLC based on only common sense and experience. Because it employs two state-control equations that are not known to everyone. The FLC is a Sugeno-style FLC with only one fuzzy variable, distance between the rear center of the truck and the loading dock. There are only two fuzzy rules in the rule base:

(1) If the distance is far then the output is the result of equation (1).

(2) If the distance is near then the output is the result of equation (2).

where equations are the state-control equations mentioned above. This makes the FLC in MATLAB very much dependent on mathematical equations.

To avoid too much dependence on mathematics, we decide to develop a new FLC based only on common sense, just like the FLC designed for the pendulum application previously.

4.3.2 FLC for Truck Backer-Upper

We will design an FLC for truck backer-upper in this section.

First, we should determine the input/output of the FLC. The input variables are the truck angle φ and the x-position coordinate x. The output variable is the steering angle θ. We assume that enough clearance between the truck and the loading dock so we could ignore the y-position coordinate. The variable ranges are as follows:

$$-50 \le x \le 50$$
$$-90^\circ \le \varphi < 270^\circ$$
$$-45^\circ \le \theta \le 45^\circ$$

Positive values of θ represent counter-clockwise rotations of the steering wheel, i.e., the front wheel.

Then we should determine the labels of fuzzy set for each variable. The definition is given below:

Angle φ		*x-position* x		*Steering angle* θ	
RB:	Right Below	LE:	Left	NL:	Negative Large
RU:	Right Upper	LC:	Left Center	NM:	Negative Medium
RV:	Right Vertical	CE:	Center	NS:	Negative Small
VE:	Vertical	RC:	Right Center	ZE:	Zero
LV:	Left Vertical	RI:	Right	PS:	Positive Small
LU:	Left Upper			PM:	Positive Medium
LB:	Left Below			PL:	Positive Large

As we mentioned previously, fuzzy membership functions can have different shapes depending on the designer's preference or experience. In practice, fuzzy engineers have found triangular and trapezoidal shapes help capture the modeler's sense of fuzzy numbers and simplify computation. Fig. 4.31 shows the membership functions that were determined after system tune-up. We choose both triangular and trapezoidal shaped functions for the input membership functions, while we adopt *singleton* functions for the output θ.

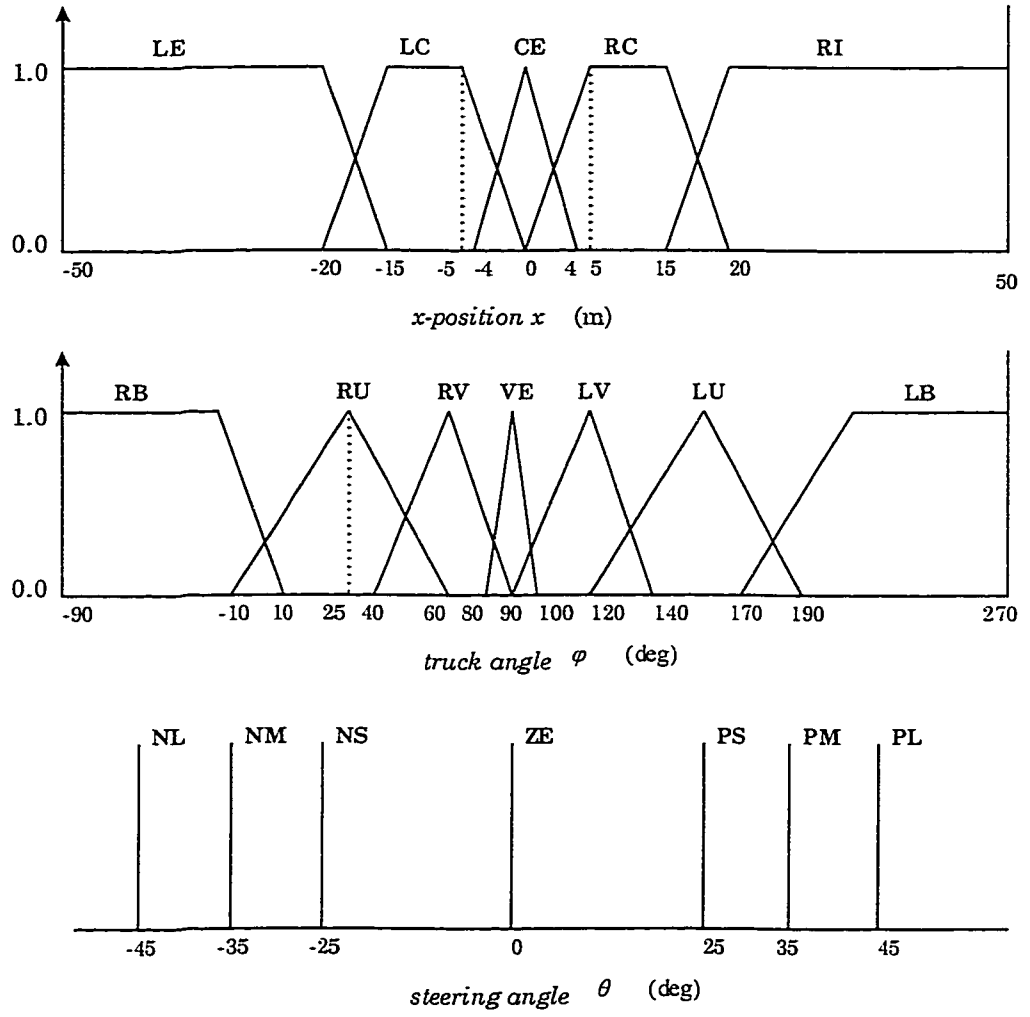


Figure 4.31 Membership functions for truck backer-upper system

In Fig. 4.31, the fuzzy sets CE, VE and ZE are narrower than the other fuzzy sets. These narrow fuzzy sets permit fine control near the loading dock. In contrast, we use wider fuzzy sets to describe the endpoints of the range of the fuzzy variables, because rough control is enough when far from the loading dock. These schemes are trying to make the design closer to the natural life.

Next we should specify the fuzzy rule base. In this case, we define 35 rules in the rule base, all of which are easy to understand.

φ \ x	LE	LC	CE	RC	RI
RL	NL [1]	NL [2]	NM [3]	NM [4]	NS [5]
RU	NL [6]	NL [7]	NM	NS	PS
RV	NL	NM	NS	PS	PM
VE	NM	NM	ZE [18]	PM	PM
LV	NM	NS	PS	PM	PL
LU	NS	PS	PM	PL	PL
LL	PS [31]	PM [32]	PM [33]	PL [34]	PL [35]

Figure 4.32 Rule base for the truck backer-upper system

Examine the rule base shown in Fig. 4.32, those rules reflect the symmetry of the controlled system. Rule #18 in the center indicates that if the truck is in near the equilibrium position, then the controller should not produce a positive or negative steering angle, but a zero angle.

Unlike the FLC that we developed for the inverted pendulum, Sugeno-style fuzzy inference is adopted in this application. In fact this is determined when we decide to use *singleton* functions to define the membership functions of the output, steering angle θ. *Min-Max* is still employed for fuzzy logical operations. *Centroid* defuzzification scheme is also adopted in this application. As we mentioned in Chapter 3 that employing *singleton* membership functions would simplify the defuzzification, the defuzzification in this application will perform the simple operation of equation (3.1).

$$Y = \frac{\sum (fuzzy\_output_i) \times (\begin{matrix} Singleton & position \\ on & X-axis_i \end{matrix})}{\sum_i (fuzzy\_output_i)} \quad (3.1)$$

4.3.3 Simulink Simulation of the FLC for Truck Backer-Upper

After the analysis of the I/O characteristics, we need to demonstrate the improvement made by adopting dithering through simulation. Therefore, an experiment scenario is designed as follows:

(1) The simulation area is a 100 x 50 rectangle. And the loading dock is at the center bottom.

(2) The initial state of the truck can be chosen randomly, but under certain condition: it has enough clearance from the back wall.

(3) When the simulation starts, the FLC will generate steering angle θ to control the backing up. The truck will stop when it hits the loading dock.

(4) The sampling time of the system is 0.1s.

With this simulation, we can examine if applying dithering technique to digital FLC could help the truck backing up to be accurate and smooth. The aspects of our interest are whether the truck can back up to the loading dock, whether the controller can output smooth steering angle and the time that the whole process takes. A good controller that we expect should have the following features. It controls the truck to back up to the desired dock; it performs a smooth control, without outputting vibrating signal; and it achieves the goal in a short time. Therefore, we will focus on these aspects in our simulation.

The block diagrams of analog system, digital system without dithering and digital system with dithering are given in Fig. 4.33, Fig. 4.34 and Fig. 4.35, respectively.

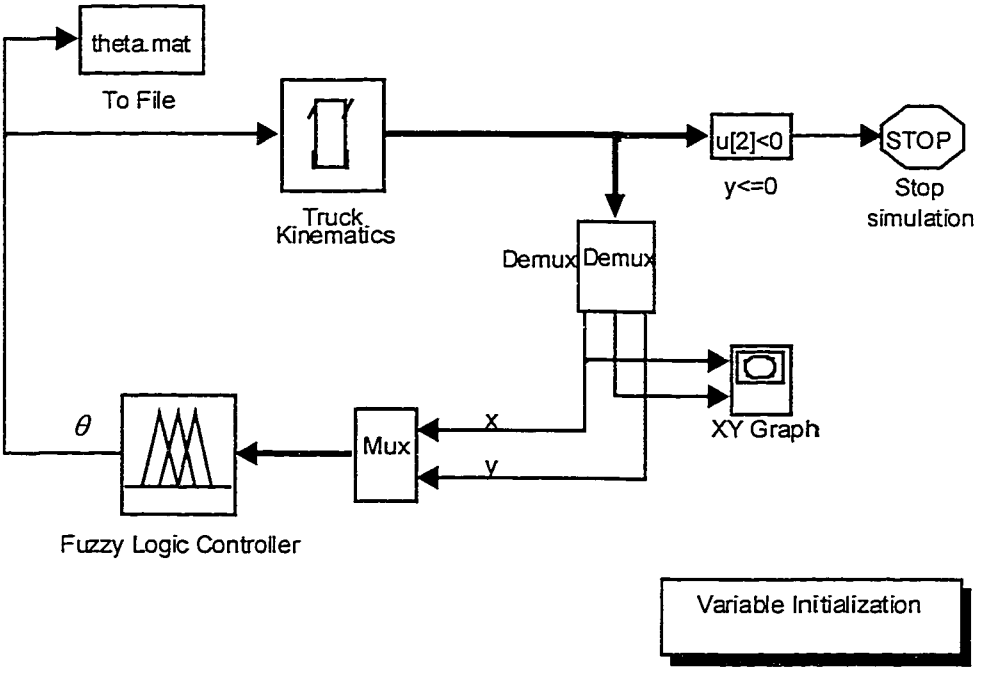


Figure 4.33 Analog FLC for truck backer-upper

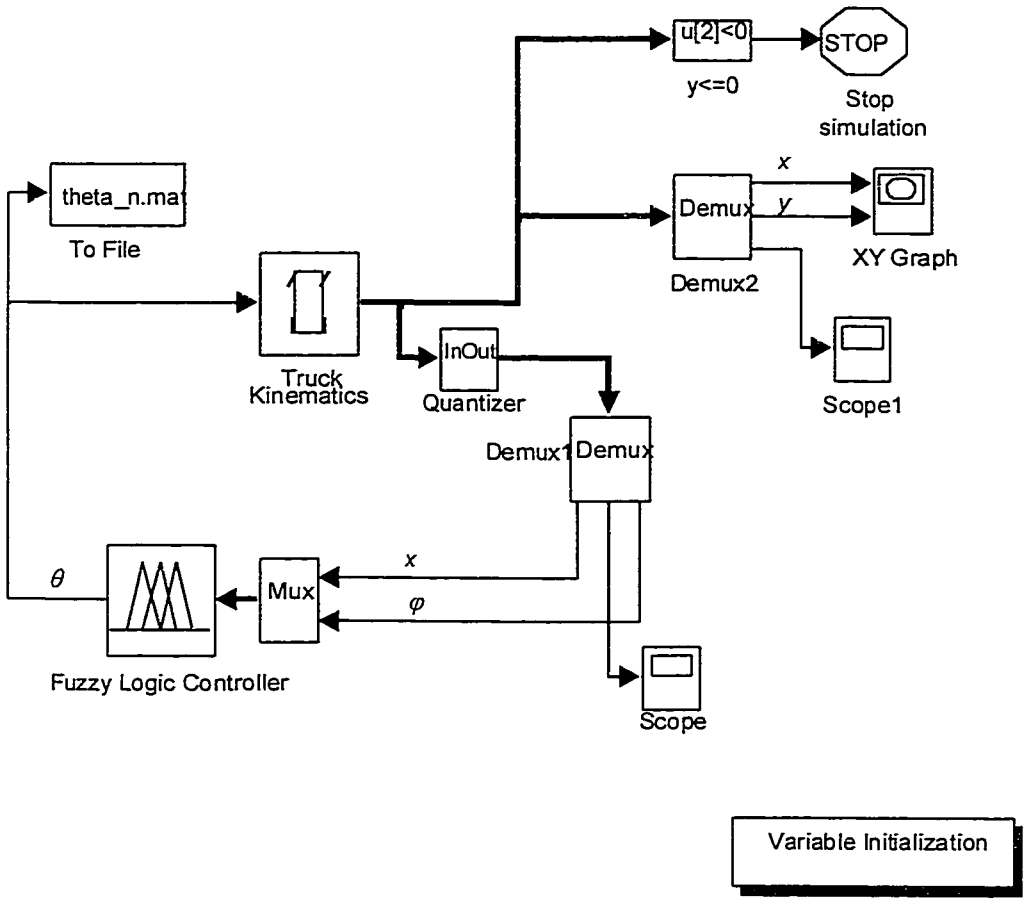


Figure 4.34 Digital FLC system for truck backer-upper; without dithering

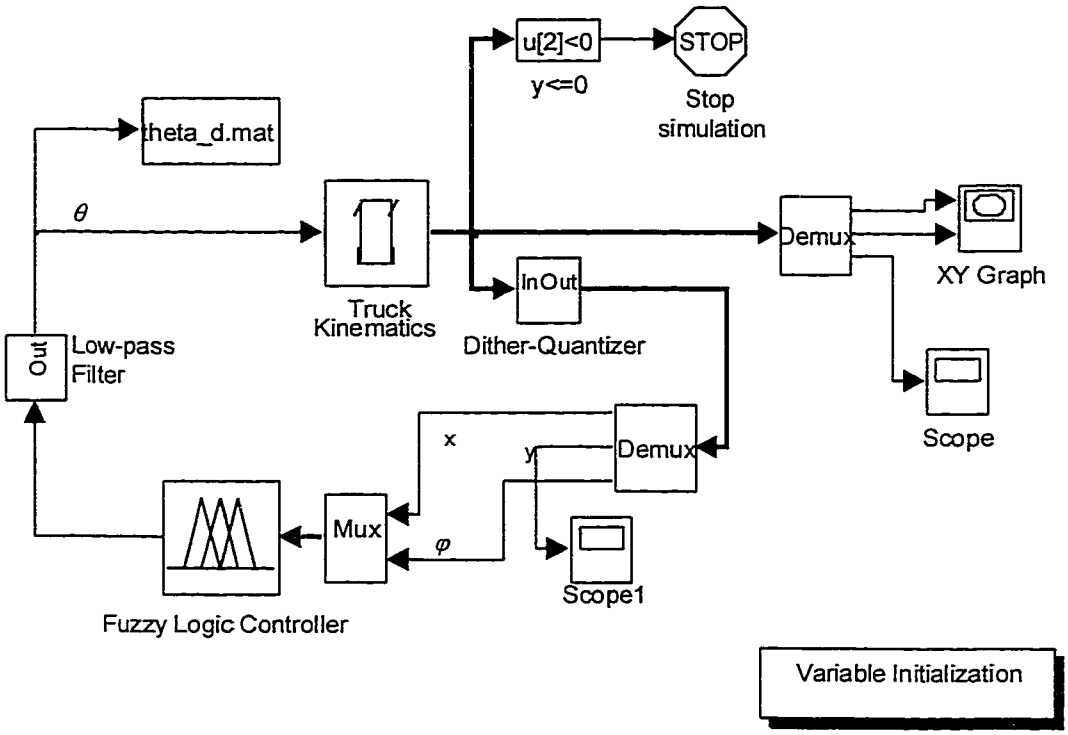


Figure 4.35 Digital FLC system for truck backer-upper; with dithering

The *Truck Kinematics* blocks in these three figures are built according to a set of state equation (4.2).

$$\begin{cases} \dot{x} = -v\cos(\varphi) \\ \dot{y} = -v\sin(\varphi) \\ \dot{\varphi} = -\dfrac{v}{l}\sin(\theta) \end{cases} \tag{4.2}$$

where $\dot{x}$, $\dot{y}$ and $\dot{\varphi}$ are the first order derivative of x, y and φ, respectively. v is the backing up speed of the truck and l is the length of the truck. Note that (4.2) is different from the equations in [39]. This is because the v here is the overall backing up speed, instead of the speed of the wheel. The structure of it is shown in Fig. 4.36. The function of the blocks in dashed-line rectangle is to constrain φ between $[-90°,270°]$.

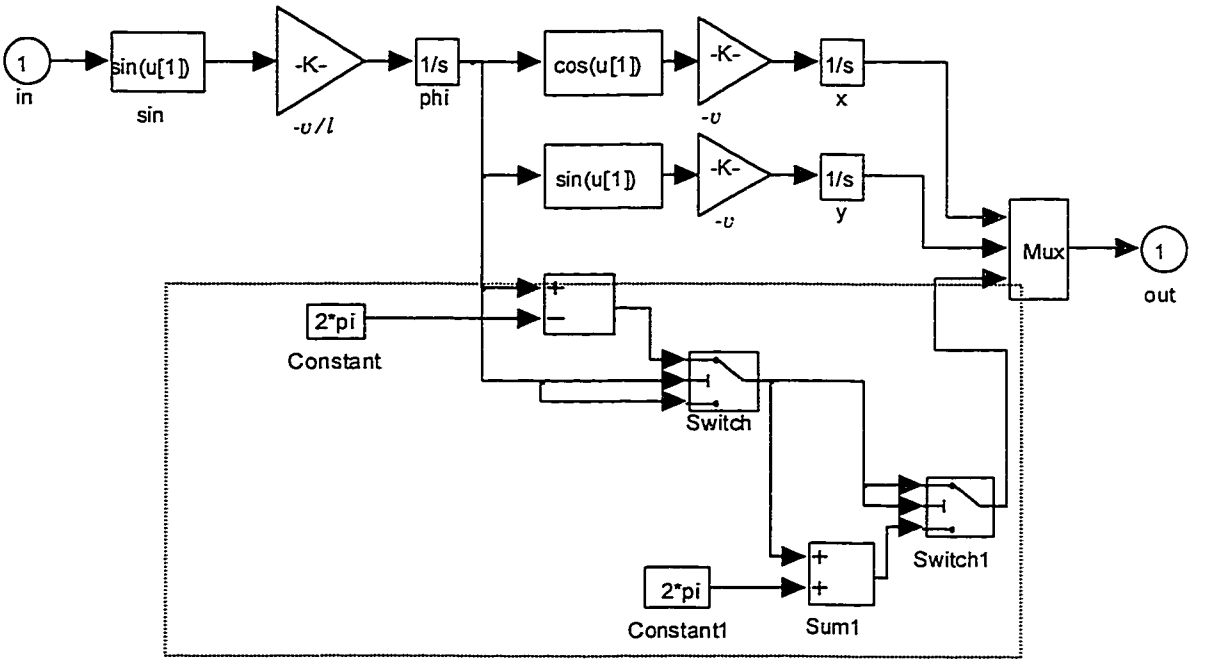


Figure 4.36 Structure of the truck kinematics

The *Quantizer* and *Dithered Quantizer* blocks are similar to those in the first application. However, the *Dithered Quantizer* in Fig. 4.35 actually has two separate quantizers so that it can quantize both x and φ in one block. The *Low-pass Filter* is similar to the one introduced previously as well. The sampling rate in two digital systems is chosen as 0.1s, i.e., FLC updates the steering angle once every 0.1s. The reason is that we have to choose a suitable updating frequency to make the control effective and efficient.

To initialize the simulation, one can configure the following parameters: the length and speed of the truck, initial position and truck angle. Without losing generality, we assume that

$$l = 4m$$
$$v = 2m/s$$

We quantize x to 4-bit and φ to 4-bit, so that the resolution of x is 6.25m and that of φ is 22.5°. Therefore, 4-bit/4-bit quantization is very low in resolution for this truck backer-upper system.

We have experimented on different initial states. A general case will be presented. Assume the initial state is

$$x = -30$$
$$y = 25$$
$$\varphi = -78.5^\circ$$

Three systems start their simulations from this same initial state. The trails of the truck are shown in Fig. 4.37. Trail (a) and (c) both end at the loading dock (0,0), while the trail (b) does not reach the dock in the end. Among our repetitive experiments, this tendency is overwhelmed. This fact illustrates that applying

dithering technique to digital FLC results in a more accurate backing up into the desired loading dock.

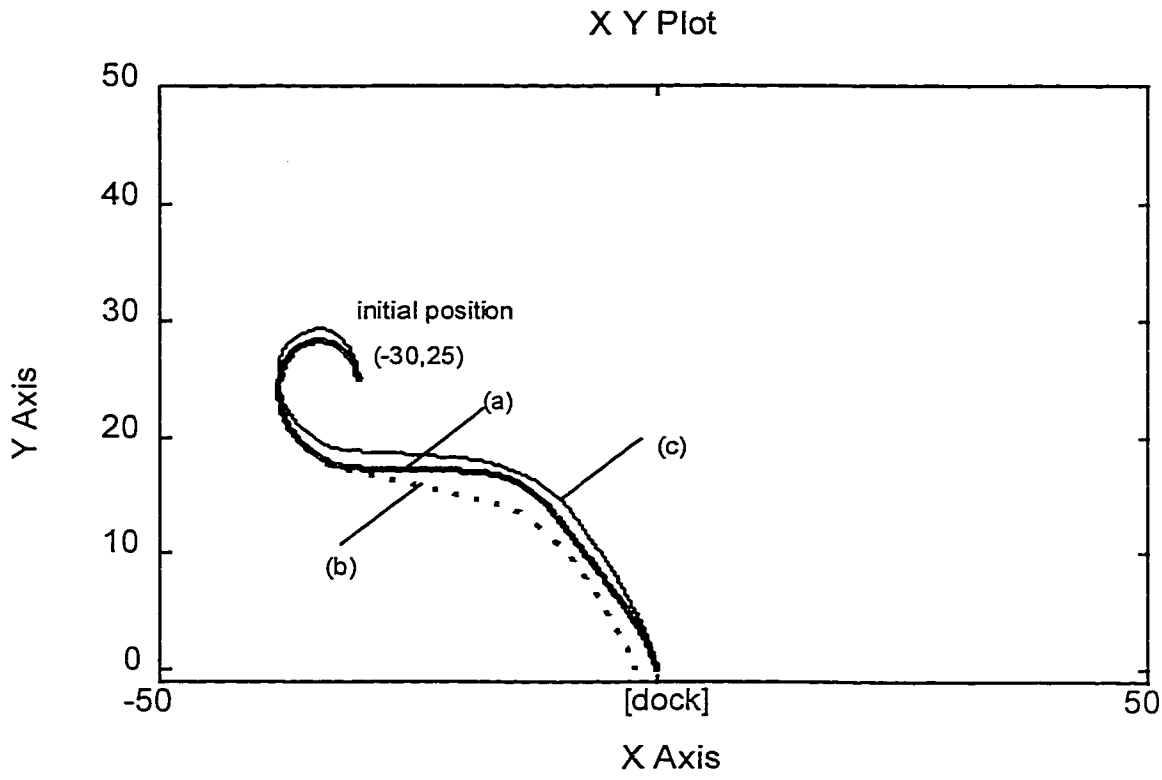


Figure 4.37 Simulated truck trails of different systems

(a) analog FLC; (b) digital FLC without dithering;

(c) digital FLC with uniform dithering and 20-unit moving average filter

After checking the performance on accuracy, we now come to examine the stability of the controller output. Fig. 4.38 - Fig. 4.40 record the changes of the FLC output, steering angle θ, for three systems under consideration.

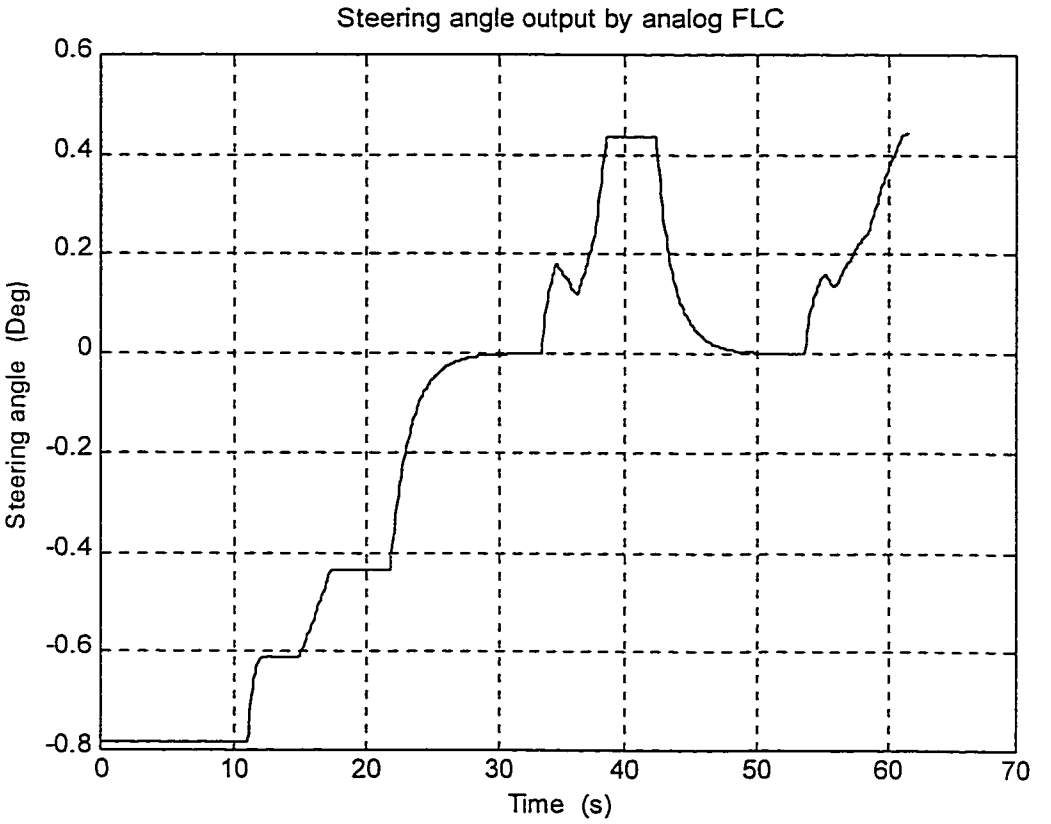


Figure 4.38 Analog FLC output - time diagram

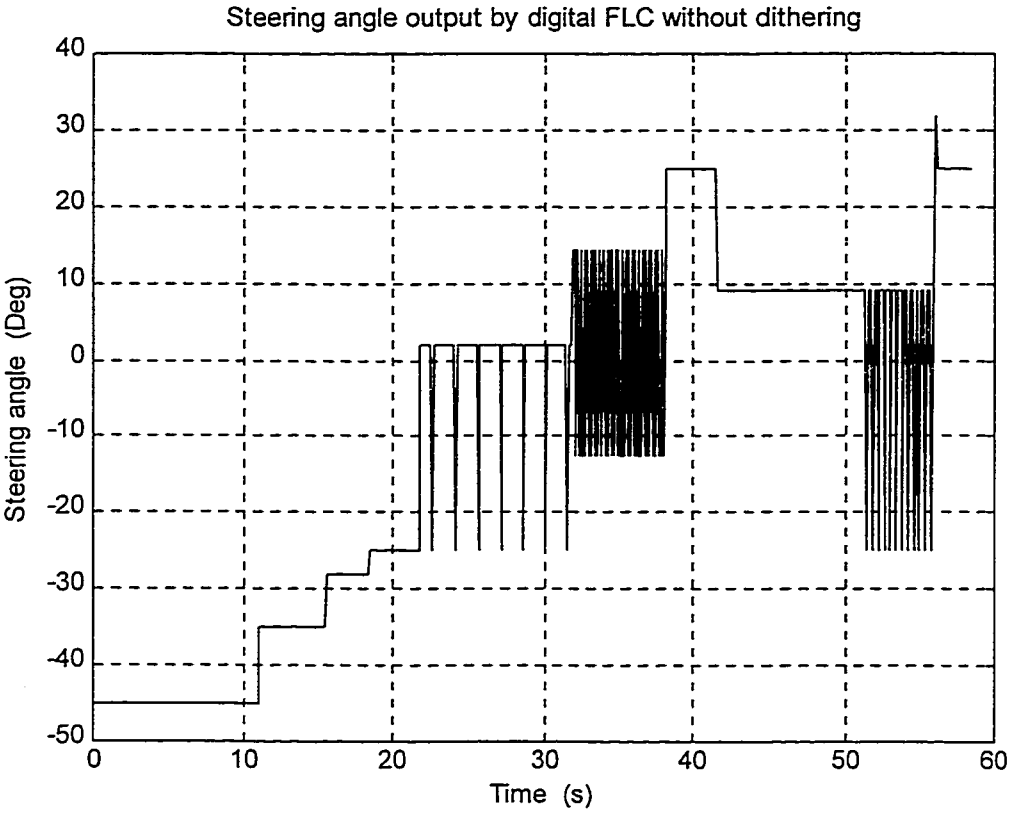


Figure 4.39 Digital FLC (without dithering) output - time diagram

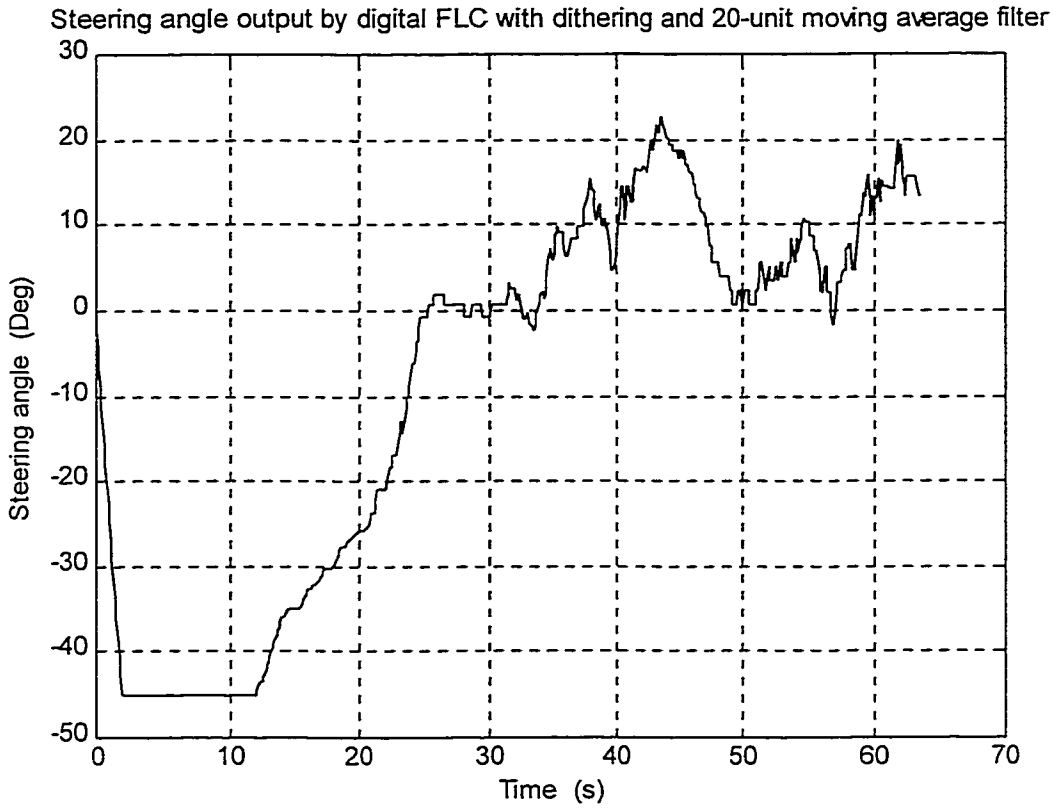


Figure 4.40 Digital FLC (with dithering) output - time diagram

Compare Fig. 4.38, Fig. 4.39 and Fig. 4.40, we can easily find that Fig. 4.39 displays an output with high frequency. From *t=22s* to the end of the simulation, the FLC is generating steering angle quickly jumping between negative and positive values. This is impractical and definitely unacceptable for a controller. We have seen that with dithering technique, the vibration in output is reduced significantly and the tendency of the output curve is close to that of the analog system. This also supports our point that applying dithering technique to digital FLC will significantly improve its performance.

Note that we took 20-unit moving average filter as an example in Fig. 4.37. The reason is given below. As we know, more units in the moving average filter result in smoother approximation of the mean, but longer delay. This specific application is not as time-sensitive as the inverted pendulum application. Therefore, we can choose more units in moving average filter to smoothen the output, as long as the delay does not fail the backing up.

Fig. 4.41 shows a performance comparison of dithered FLC with different moving average filters. If we only consider the accuracy of the backing up, (c), the system with 20-unit moving average filter is the best one. The others can not back up the truck to the dock very accurately.

However, if we take time into consideration, we will find that the more units are in the filter, the longer it takes to back up the truck to the dock. The records of steering angle θ in simulations for different truck backer-upper system are shown in Fig.4.42.

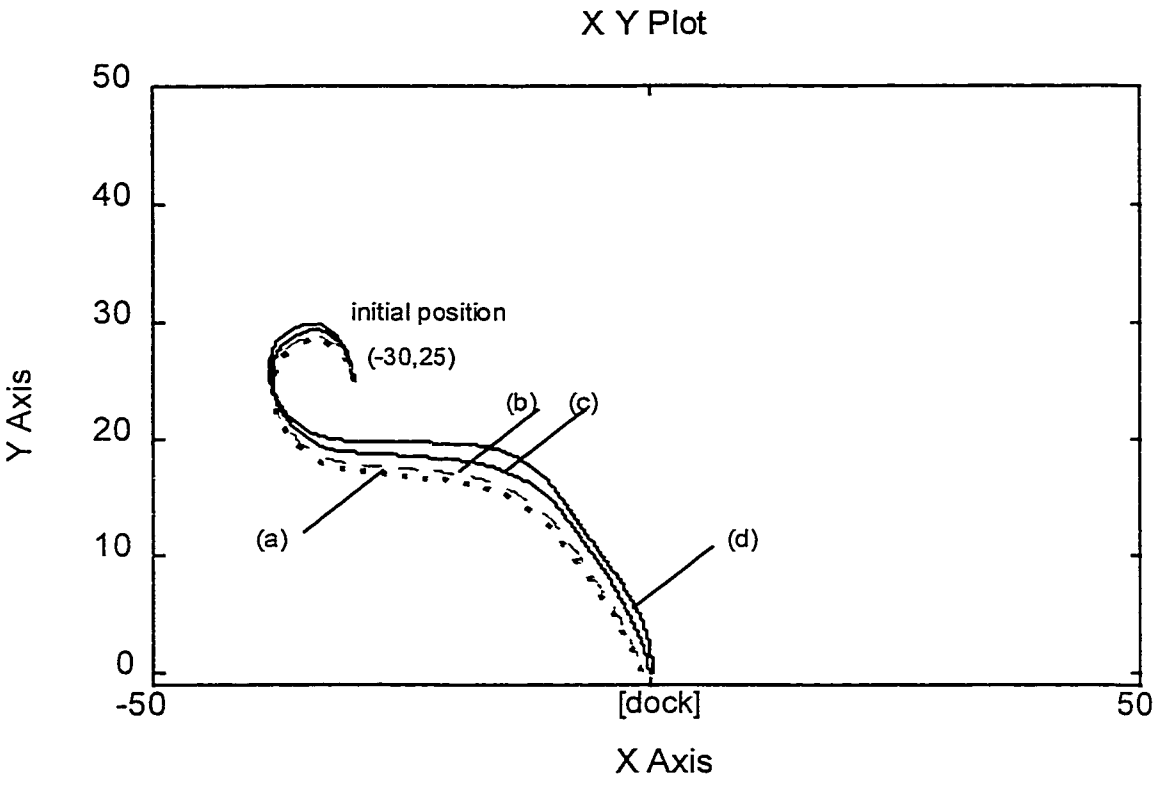


Figure 4.41 Performance comparison of different dithered FLC

(a) with 5-unit moving average filter; (b) with 10-unit moving average filter; (c) with 20-unit moving average filter; (d) with 30-unit moving average filter.

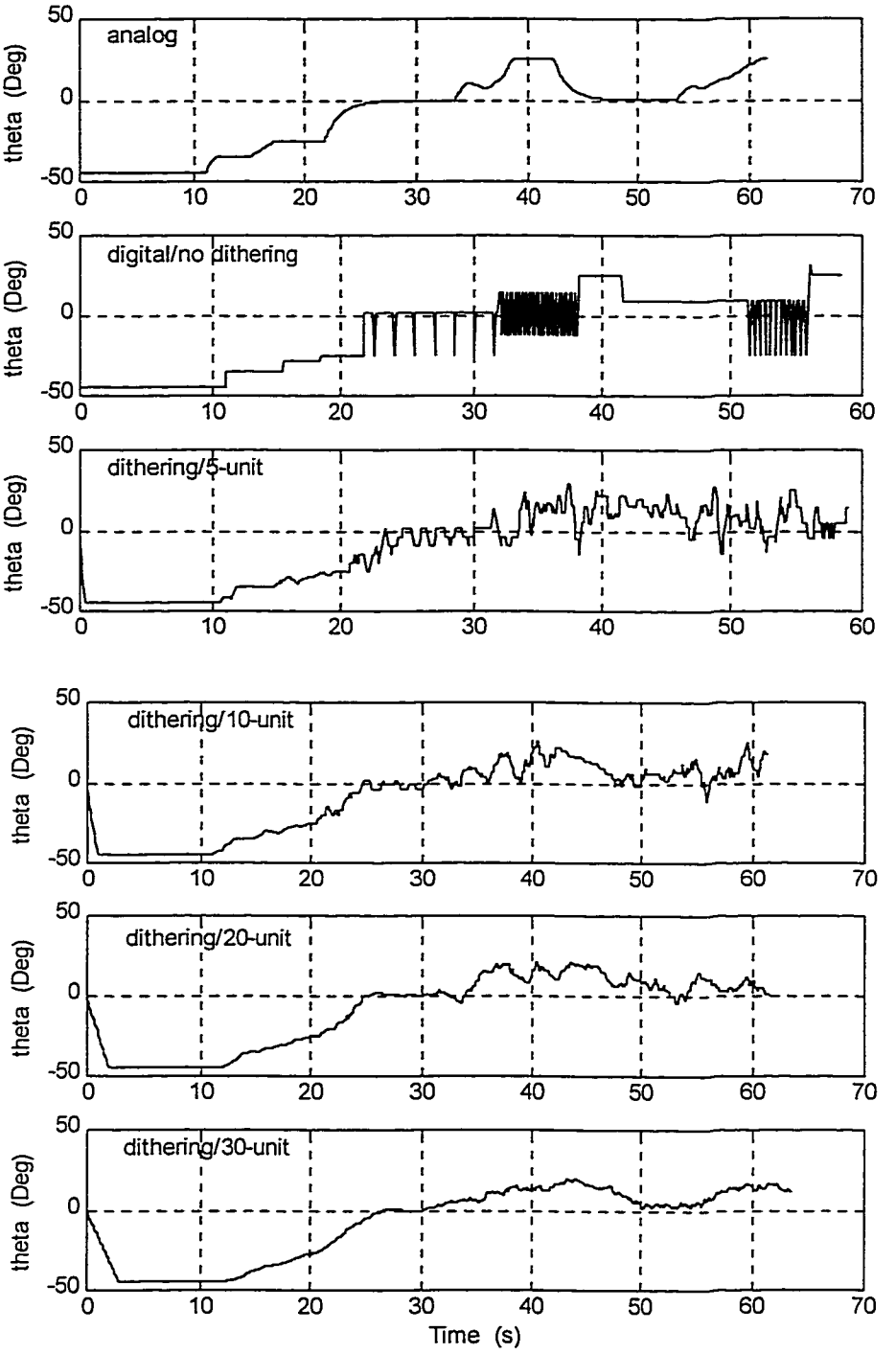


Figure 4.42 Record of θ in the simulation process for different dithered FLC's

Apparently, the digital FLC without dithering finishes the simulation in the shortest time, but the waveform of θ consists of many high frequencies. In general, digital FLC systems with dithering spend longer time backing up the truck to dock; Moreover, the more units in the moving average filter, the longer it costs to back up. The reason is that processing low passing data costs time, although they make improvement in other aspects. However, we must acknowledge that the positive effect of adopting dithering technique is more significant and important.

4.3.4 Analysis of I/O Characteristics

Following the first application, we examine the I/O characteristics in this section. The control surface of the analog FLC is displayed in Fig. 4.43. The digital FLC system simulated in previous section is also studied here. The I/O characteristics of such a system without dithering is shown in Fig. 4.44. Then we apply uniform dithering to this quantized system and its control surface is shown in Fig. 4.45. Apparently the I/O characteristics of the system with dithering is much smoother than that without dithering, which again conforms to the simulation results.

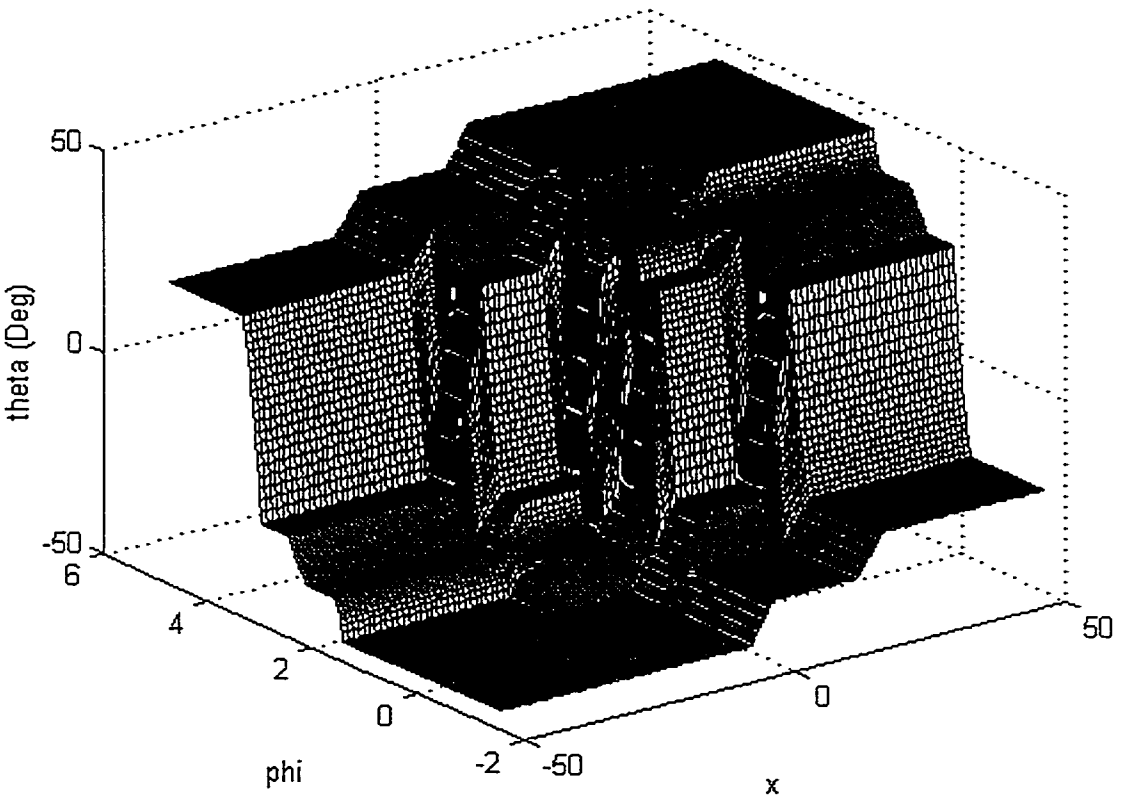


Figure 4.43 I/O characteristics of the analog FLC for truck backer-upper

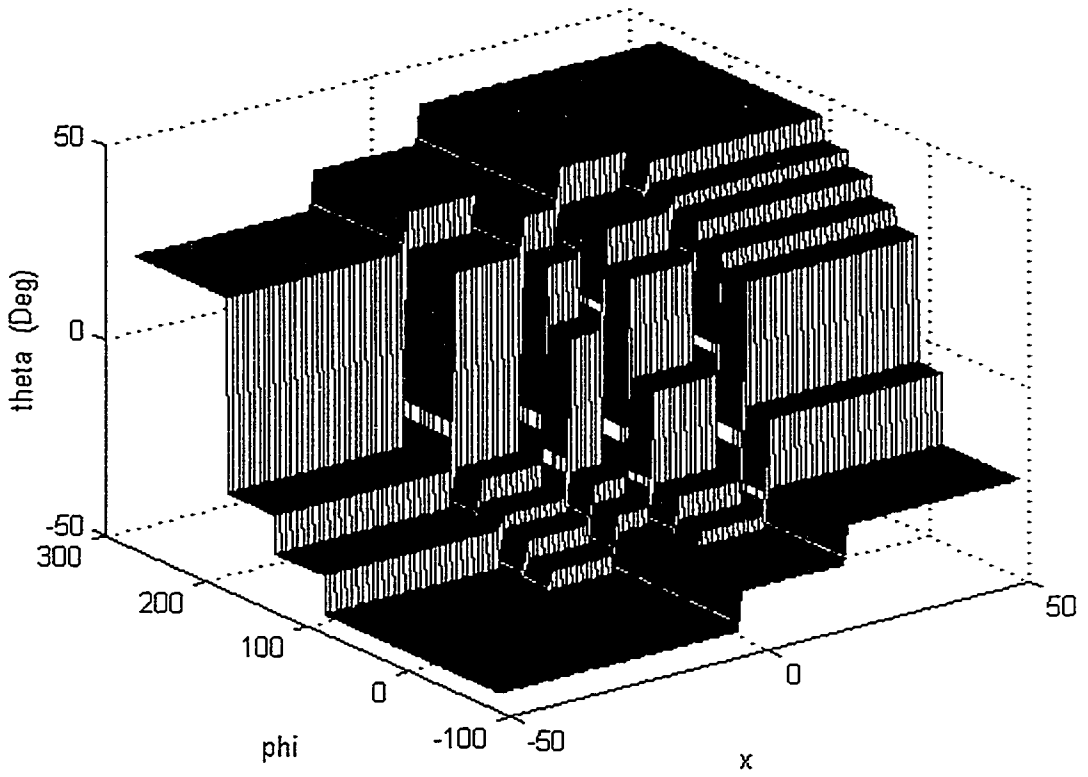


Figure 4.44 I/O characteristics of the digital FLC for truck backer-upper

4-bit/4-bit quantization **is applied to the inputs; without dithering**

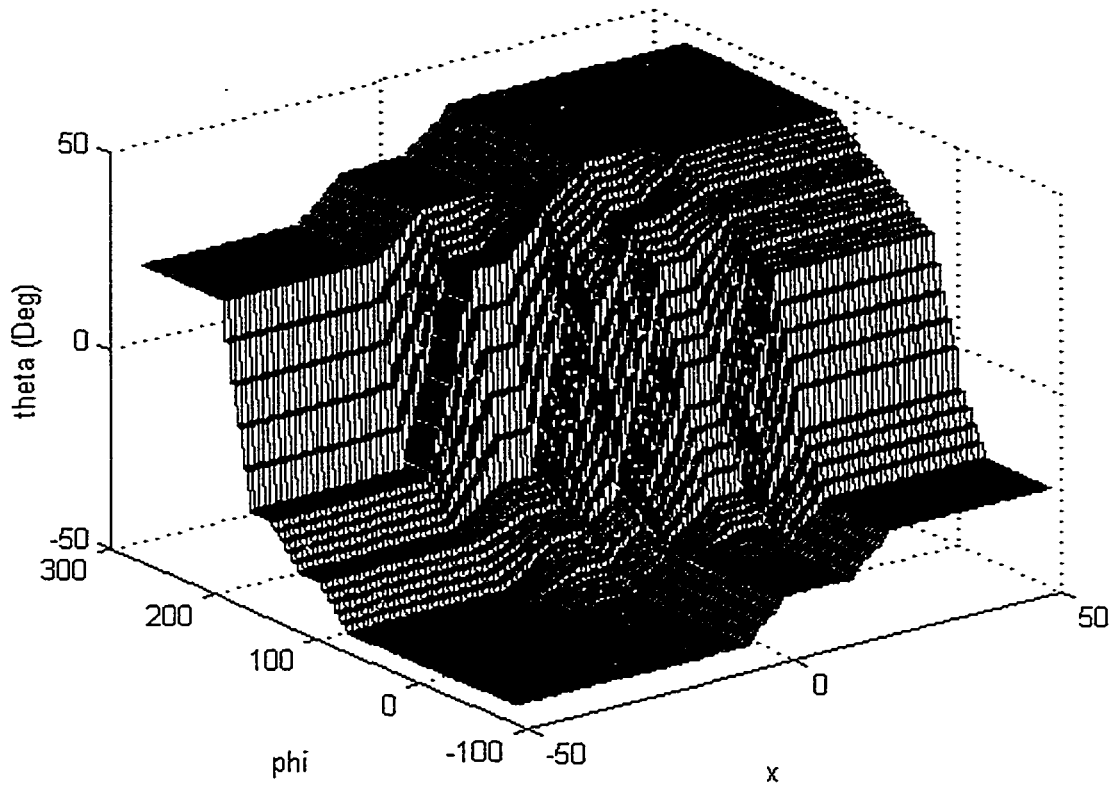


Figure 4.45 I/O characteristics of the digital FLC for truck backer-upper

4-bit/4-bit quantization **are applied to the inputs; with uniform dithering**

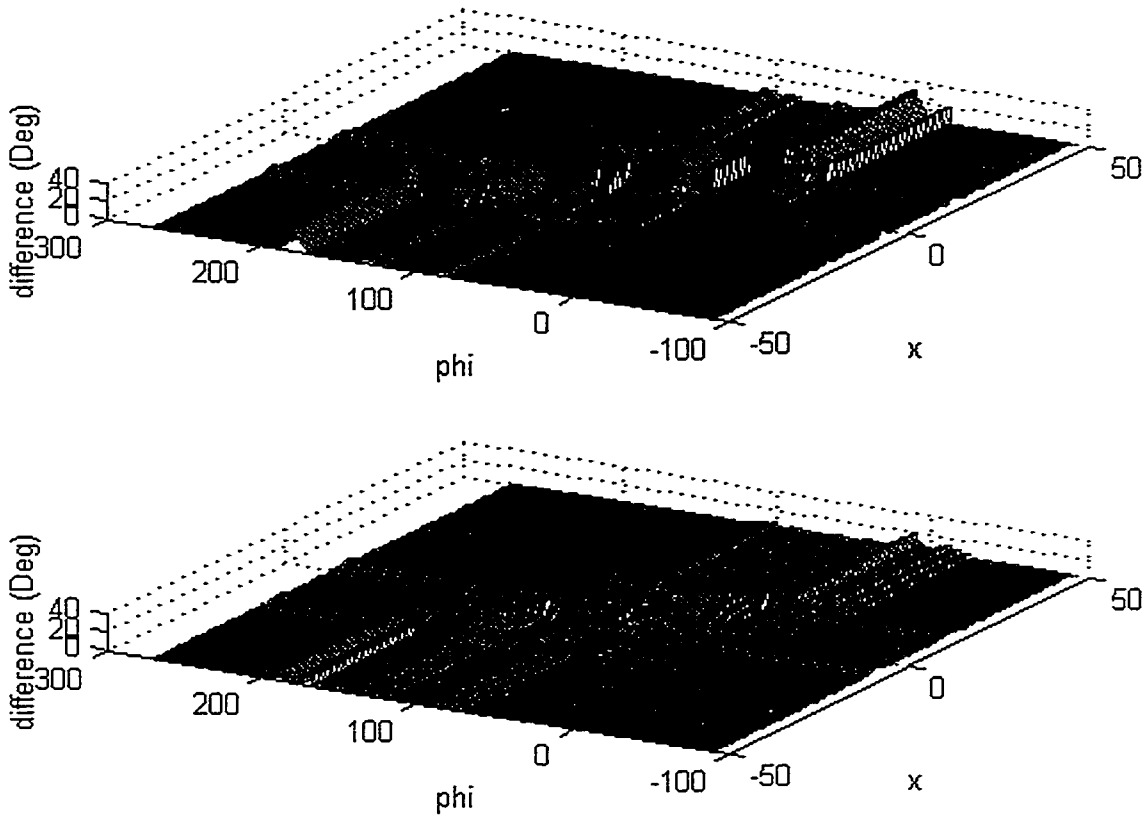


Figure 4.46 Differences in the FLC's output θ

Upper: difference in the case of 4-bit/4-bit inputs relative to analog inputs

Lower: differences in the case of dithered 4-bit/4-bit relative to analog inputs

Fig. 4.46 illustrates comparatively the behavior of the FLC with 4-bit/4-bit inputs and the FLC with 4-bit/4-bit dithered inputs relative to the analog FLC. The

maximum absolute error in the former case is 0.692 *rad*, while the maximum absolute error in the other case is just 0.521 *rad*. As expected, dithering leads to a notable improvement in the FLC's output resolution.

4.4 Virtual Prototyping Environment for Fuzzy Truck Backer-Upper

We have simulated the truck backer-upper in MATLAB. In this section, we simulate the truck backer-upper in a 3-Dimensional environment by using VRML 97.

VRML is an acronym for the *Virtual Reality Modeling Language*. There are two versions of VRML: VRML 1.0 and VRML 97. VRML 97 is the second and most recent revision of the VRML specification [55]. Using VRML, it is easy to build the three-dimensional virtual world on the Internet. The most exciting feature of VRML is that it enables you to create dynamic worlds and sensory-rich virtual environments on the Internet, including the ability to:

- Animate objects in the virtual worlds, making them move.
- Play sounds and movies within the worlds.
- Allow users to interact with the worlds.
- Control and enhance worlds with scripts, small programs users create to act in the VRML world.

In order to view the VRML document on the Internet, the Web browser (we are using Netscape Communicator 4.05.) needs a VRML helper application or plug-in called VRML browser. The list of available VRML browsers is growing rapidly.

Some of the most common VRML browsers are Silicon Graphics's CosmoPlayer, Sony's Community Place, Intervista's World View, VREAM's Wirl and Dimension X' s Liquid Reality. In our application, we use Silicon Graphics's CosmoPlayer, which is integrated in Netscape Communicator.

However, we still need an interface between the virtual truck in VRML and the work engine --- FLC. Though VRML provides a *Script* node, which enables user to create a node in JavaScript or Java to perform complex actions, it is hard to use the Script node to set up the interface between a VRML world and its external environment. Silicon Graphics provides an interface called *External Authoring Interface* (EAI) [56]. It defines a set of functions on the VRML browser that the external environment can perform to affect the VRML world. Fig. 4.47 shows the relationship between this interface and the rest of the VRML environment.

Java EAI allows Java applets (external program) embedded on an HTML page to communicate with a VRML scene on that same page. The EAI allows 4 types of access into the VRML scene:

1. Accessing the functionality of the Browser Script Interface.
2. Sending events to EventIns of nodes inside the scene.
3. Reading the last value sent from EventOuts of nodes inside the scene.
4. Getting notified when events are sent from EventOuts of nodes inside the scene.

Therefore, we have an interface between a Java applet on an HTML page and an embedded VRML world on that same page. Thus, we can develop other Java classes, which can be called by the Java applet. And this is the basic idea through our truck backer-upper simulation in VRML.

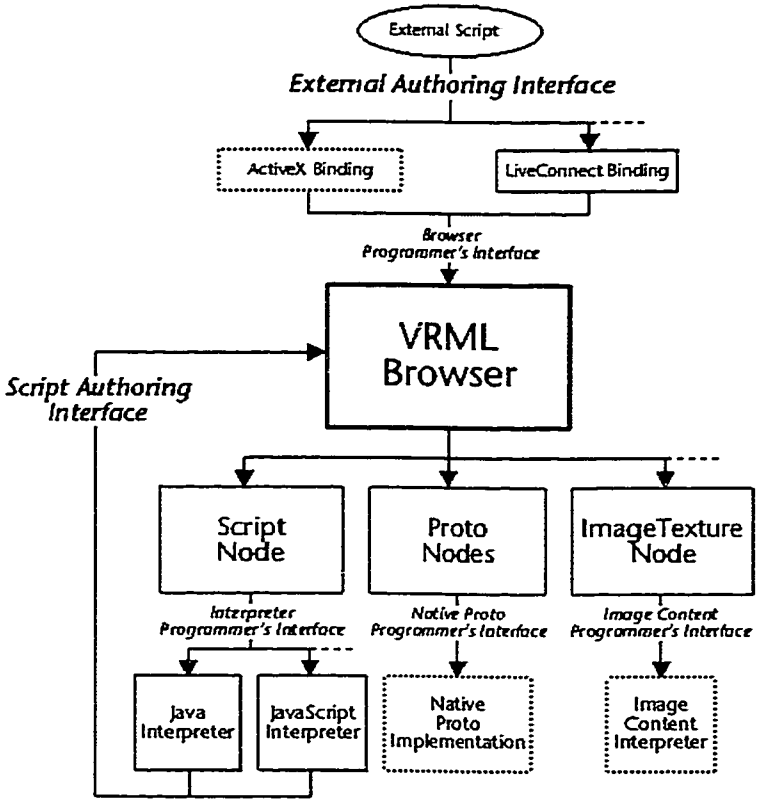


Figure 4.47 Anatomy of a VRML Browser (adapted from [56])

Before we start discussing how to implement the fuzzy truck backer-upper in VRML, we will introduce some concepts about computer animation [16]. In computer animation, the task of specifying the motion of an animated object to the computer is surprisingly difficult and even animating a simple object like a bouncing ball can present problems. This task is difficult because humans are very skilled at observing motion and quickly detect motion that is unnatural or implausible.

A number of techniques have been developed for specifying motion, but all the available tools require a tradeoff between automation and control. *Keyframing* allows fine control but does little to automatically insure the naturalness of the result. *Procedural methods* and *motion capture* generate motion in a fairly automatic fashion but offer little control over fine details.

Borrowing its name from the traditional hand animation technique, keyframing requires the animator to specify key positions for the objects being animated. The computer then interpolates to determine the positions for the in-between frames. The interpolation algorithm is an important factor in the appearance of the final motion. The simplest form of interpolation, *linear interpolation*, often results in motion that appears jerky because the velocities of the moving objects are discontinuous. But due to its simplicity, linear interpolation is still widely used in animation.

Current technology is not capable of generating motion automatically for arbitrary objects; however, algorithms for specific types of motion can be built. These techniques are called *procedure methods* because a computer procedurally follows the steps in an algorithm to generate the motion. Procedure methods have two main advantages over keyframing techniques: they make it easy to generate a family of

similar motions, and they can be used for systems that would be too complex to animate by hand, such as particle systems or flexible surfaces.

A third technique for generating motion, *motion capture*, employs special sensors, called *trackers*, to record the motion of a human performer. The recorded data is then used to generate the animation.

As we mentioned before, VRML can animate the objects in the VRML world. Actually, it uses the keyframing techniques. VRML provides some useful nodes to help users generate animation [1]. It provides the *TimeSensor* node, which will act as a clock to control the playback of the animation. This node provides features for starting and stopping the animation, and controlling how quickly the animation plays. To describe the changes during an animation, you can use VRML' s *PositionInterpolator* and *OrientationInterpolato*r nodes. Users can specify a few key frame values (position or rotation values) in their *keyValue* field and the key fractional times for each key values in their *key* field. The VRML interpolator nodes use these key fractional times and values as a rough sketch of the animation and fill in the values between those specified by using linear interpolation. Using keyframing techniques and linear interpolation, user can describe an animation independent of the playback speed of the animation.

VRML also provides some useful sensors to sense viewer actions, make the world interactive. The common sensor node is *TouchSensor*, which can be added to any group. When in such a group, the TouchSensor node senses when the viewer moves over, clicks, or drags on any shape built in that group.

So far, we have discussed the important VRML nodes we use in our VRML-truck. Those nodes make the truck animate and interact with the viewer. Next, we

will discuss how to implement EAI and FLC in Java. Their relationships are described in Fig. 4.48.

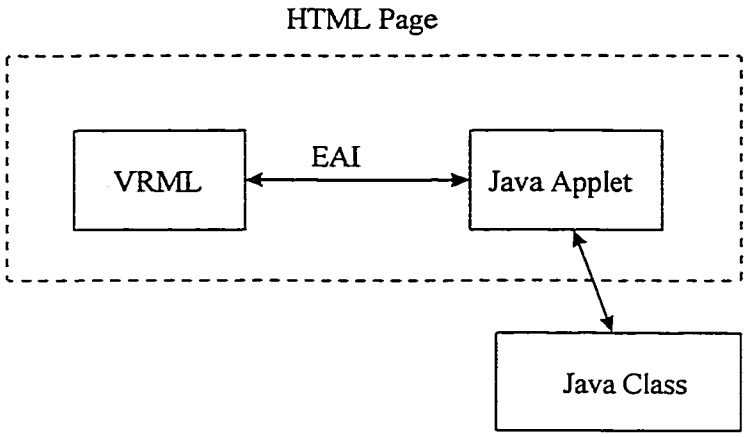


Figure 4.48 Relationship among VRML, applet and Java class

In Fig. 4.48, it shows that a VRML file and a Java applet are embedded in the same HTML page, both can communicate with each other through EAI. A Java applet communicates with a VRML world by obtaining an instance of the *Browser* class [56]. This class is the Java encapsulation of the VRML world. It contains the *getNode()* method, which returns a *Node* when given a DEF name string in VRML. Only DEF names in the base file are accessible.

Once a Node instance is obtained from the get Node() method of the Browser class, its EventIns and EventOuts can be accessed by using *getEventIn()* and *getEventOut()* methods. Once an instance of the desired EventIn is obtained, an event can be sent to it. Once an instance of a desired EventOut is obtained, 2 operations can be performed. The current value of the EventOut can be got and a

callback can be set up to be invoked when the EventOut is generated. The *advise()* method on EventOut is called to set the callback. To implement the *callback()* method, the applet must subclass the *EventOutObserver* class. Then whenever an event is generated for that EventOut the callback() method is executed and passed the value and timestamp of the event. The advise() method is also passed a user defined object. This value is passed to the callback() method and can be used by the applet author to pass user defined data to the callback.

In our simulation, we built a virtual truck in VRML called "truck_back.wrl" ; a Java applet called "TruckBack.java" ; an HTML page including them called "truckback.html" ; one Java classes called "Fuzzy.java" and another three classes called "GetTrace.java", "GetTrace_N.java" and "GetTrace_D.java", respectively. Fig. 4.49 shows the structure of the truck backer-upper simulation program.

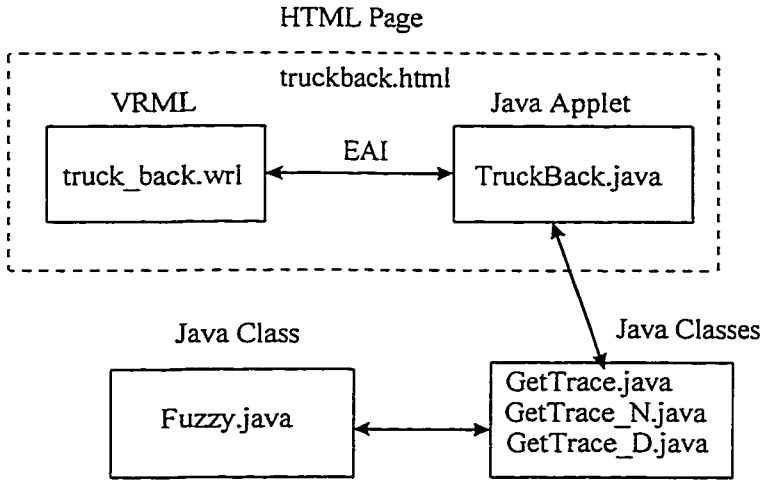


Figure 4.49 Structure of the simulation program

The Java class *Fuzzy* performs as FLC, also it has the truck kinematics part. Its main function is to determine the fuzzy outputs (steering angle), and use truck kinematics to determine the next state, then send the state to the Java classes *GetTrace, GetTrace_N* and *GetTrace_D.*

These three Java classes control the whole procedure of truck backer-upper and record the truck trace from which VRML will get the key frame values of truck position values, truck rotation values and the rotation values of the two front wheels. And these key frame values will be used to animate the truck backer-upper. The difference among them is that *GetTrace* deals with analog FLC, *GetTrace_N* deals with digital FLC, while *GetTrace_D* deals with dithered digital FLC.

We also built a friendly graphic user interface (GUI) so that it is easy for users to run the simulation. Users need input the initial position of the truck including the truck position (x, y) and the truck orientation value (Φ). To the right of these input boxes, there are three buttons, "AnalogFLC", "DigitalFLC" and "DitheredDFLC". Click on any of them, VRML will display the backing up trace corresponding to different FLC. We also set up two viewpoints to make viewer observe the simulation easily and clearly. One viewpoint is fixed called "Docking Point". The other is not fixed called "Truck Point", which allows the user to have a closer look at the truck backer-upper.

The virtual environment is captured and shown in Fig. 4.50 - 4.53. Fig. 4.50 shows the truck at an initial position (-30, 25, 2.1) from the docking point of view. Fig. 4.51 shows the virtual truck built in VRML. It is observed from the other viewpoint, "Truck Point". In this figure, you can see the detail of the truck. Fig. 4.52

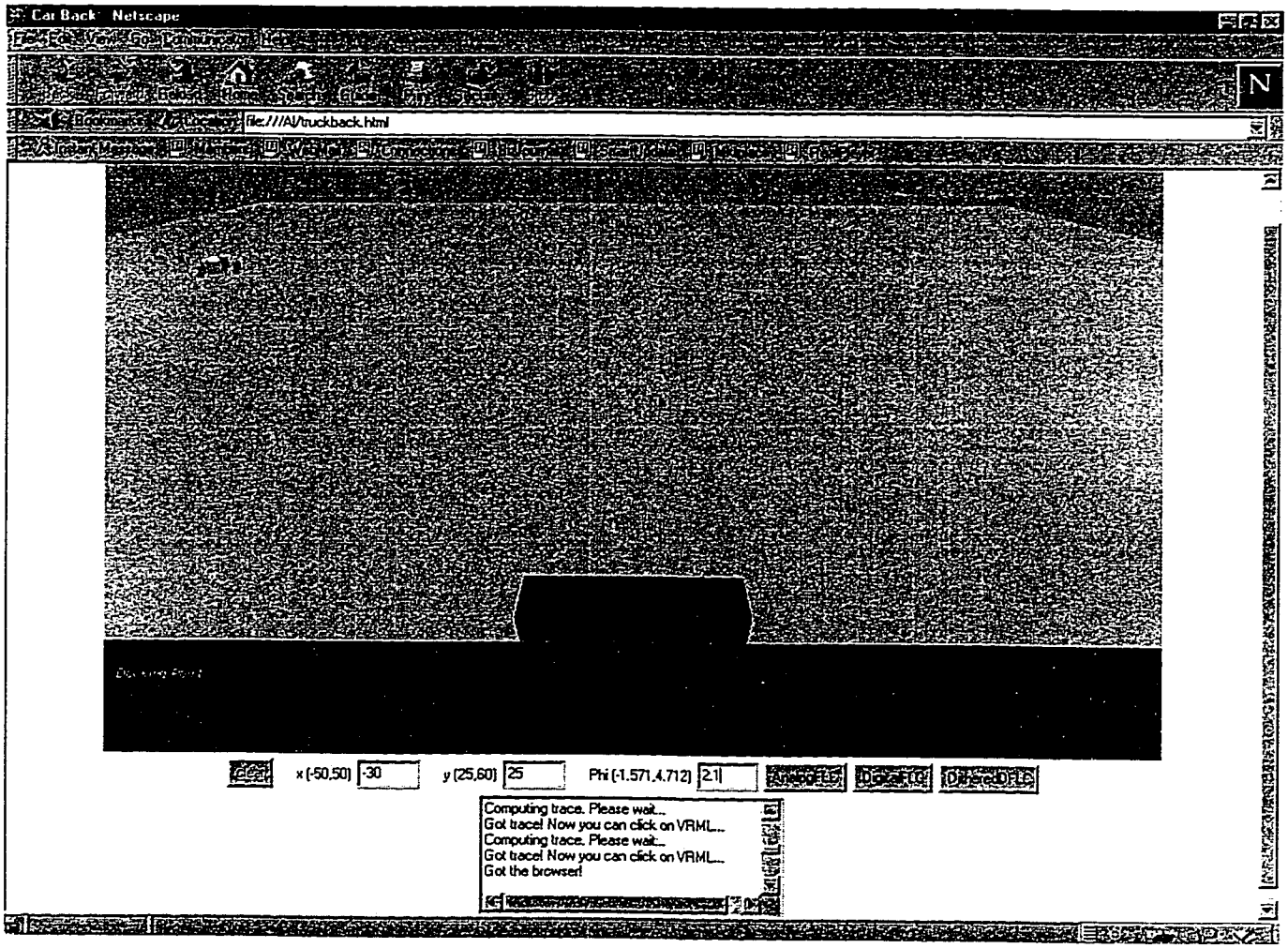


Figure 4.50 Virtual Environment of the Fuzzy Truck Backer-upper

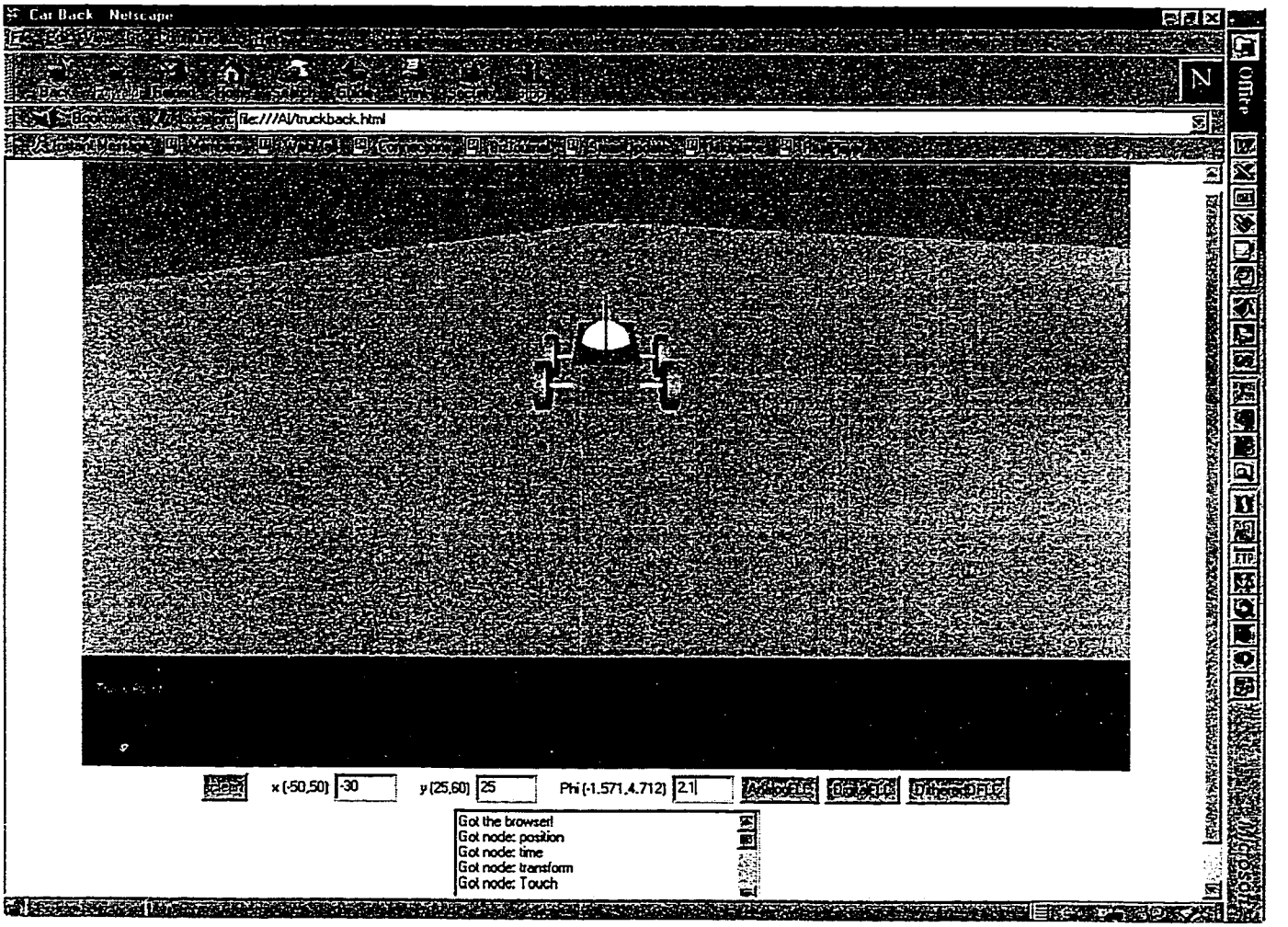


Figure 4.51 Virtual Fuzzy Truck

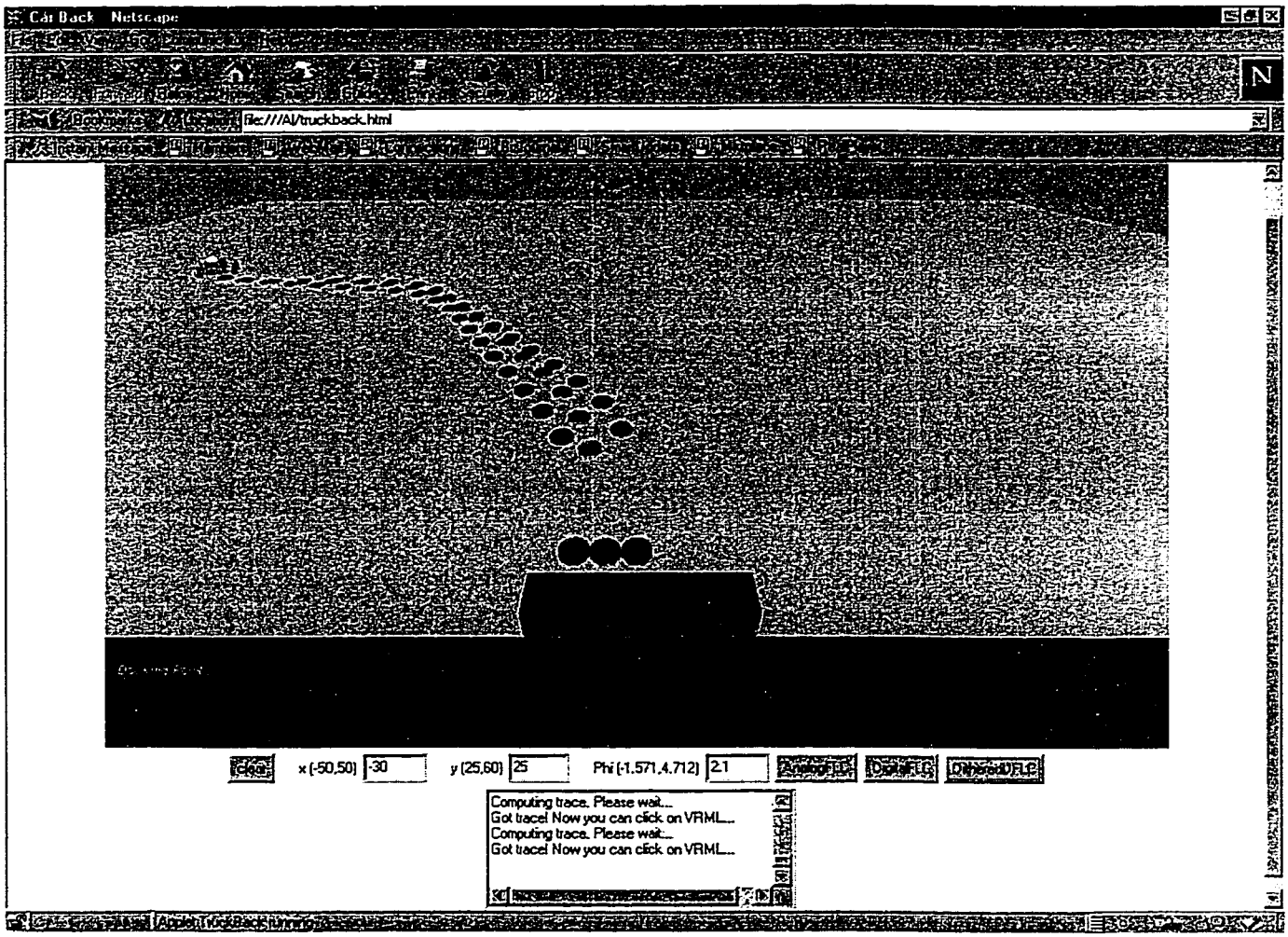


Figure 4.52 Backing Up Trace from Initial Position (-30, 25, 2.1)

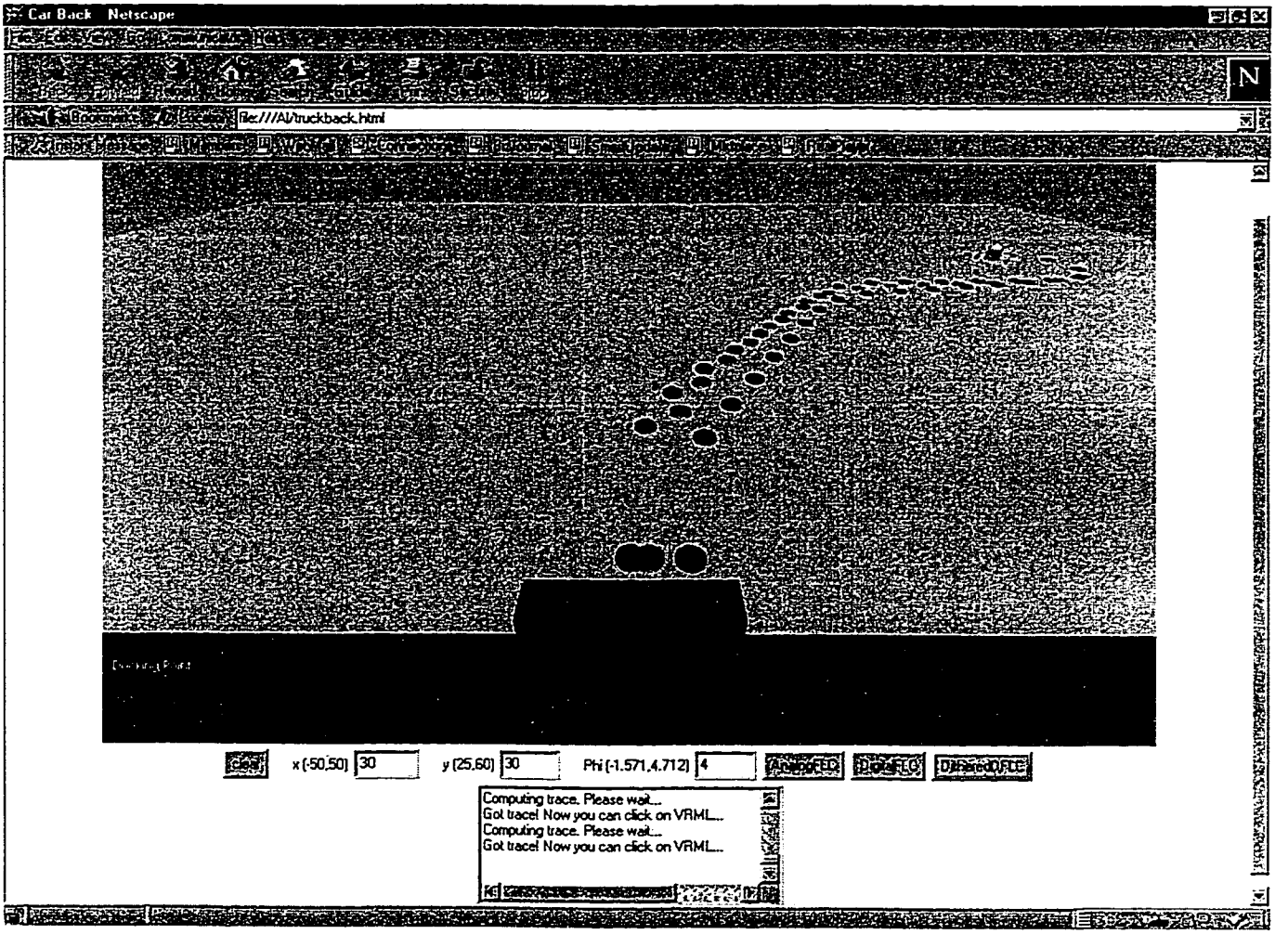


Figure 4.53 Backing Up Trace from Initial Position (30, 30, 4)

shows the three traces of the fuzzy truck backer-upper with the initial position (-30, 25, 2.1). The trace on the right is the backing up trace of analog FLC truck; the one on the left if the trace of digital FLC truck (4-bit quantization/without dithering); and the one in-between is the trace of dithered digital FLC truck (4-bit quantization/with uniform dithering). Fig. 4.53 shows a scenario with another initial position (30, 30, 4). Similar to Fig. 4.52, the dithered digital FLC has a better performance than the digital FLC without dithering.

4.5 Conclusion

Studying on the theories about quantization and dithering provides us theoretical knowledge that dithering technique can reduce the quantization error caused by quantization, one of the two steps in digitalization. This dithering technique becomes more and more important when digital techniques becomes more and more popular. In a sense, digital technique has penetrated in every area of modern life. Thus, a good technique that can reduce the inevitable quantization error due to the digitalization is of great interest.

Dithering technique has been applied to many digital processing areas, such as speech and visual signal processing, modified digital correlator and modified A/D converter, etc. Our research is to apply dithering technique to the fuzzy control area. Fuzzy control has been adopted by more and more control devices because of its simplicity compared to conventional control theory. No accurate mathematical

description of the system to be controlled is needed when fuzzy logic controller is employed. Like in many other fields, FLC also face the problem - how to reduce the quantization error caused by digitalization. Using high-resolution sensor is one of the clues; the other one is to employ dithering technique. We focus on two FLC applications: FLC for inverted pendulum and FLC for truck backer-upper. Implementation of these two applications in Simulink proves that applying dithering technique to digital FLC can improve its performance significantly. Using cheap, low-resolution sensor, you can achieve similar performance of the system with high-resolution sensor, while the cost is just employing a little more simple equipment for dithering and low-pass filtering. This has given our research practical meaning.

Although dithering may cost longer processing time and require more equipment, its advantages can overcome its drawbacks. Dithering technique is an efficient and promising technique to reduce the quantization error in digital area.

Bibliography

[1] Ames A.L., Nadeau D.R. and Moreland J.L., *VRML 2.0 Source Book,* second edition, NY: John Wiley & Sons, Inc., 1997

[2] Analog Devices, Inc., *Analog-Digital Conversion Handbook*, Englewood Cliffs, NJ: Prentice-Hall, 1986

[3] Bennett W.R., "Spectra of Quantized Signals," *Bell Syst. Tech. J.,* vol. 27, pp. 446-472, July 1948

[4] Bezdek J., "Fuzzy Models - What Are They, and Why?" *IEEE Trans. Fuzzy Syst.*, vol. 1, No. 1, Feb. 1993

[5] Blesser B.A. and Locenthi B.N., "The Application of Narrow-Band Dither Operating at the Nyquist Frequency in Digital Systems to Provide Improved Signal-to-Noise Ratio Over Conventional Dithering," *J. Audio Eng., Soc.,* vol. 35, pp. 446-454, June 1987

[6] Carbone P. and Petri D., "Effect of Additive Dither on the Resolution of Ideal Quantizers," *IEEE Trans. Instrum. Meas.,* vol. 43, No. 3, pp. 389-396, June 1994

[7] Carley L.R., "An Oversampling Analog-to-Digital Converter Topology for High-Resolution Signal Acquisition Systems," *IEEE Trans. Circuits Syst.,* vol. 34, Jan. 1987

[8] Chang K.Y. and Moore A.D., "Modified Digital Correlator and Its Estimation Errors," *IEEE Trans. Inform. Theory,* vol. 16, Nov. 1970

[9] Driankov D., Palm R. and Hellendoorn H., "Fuzzy Control With Fuzzy Inputs: The Need For New Rule Semantics," *Proc. IEEE Intl. Conf. On Fuzzzy Systems,* Orlando, FA, 1994, pp. 13-21

[10] Eatherly G.J. and Petriu E.M., "A Fuzzy Controller for Vehicle Rendezvous and Docking," *IEEE Trans. Instrum. Meas.*, vol. 44, No. 3, June 1995

[11] Finger R.A., "On the Use of Computer-Generated Dithered Test Signal," *J. Audio Eng., Soc.,* vol. 35, pp. 434-445, June 1987

[12] Gabel R.A. and Roberts R.A., *Signals and Linear Systems*, third edition, NY: John Wiley & Sons, Inc., 1987

[13] Gray R.M., "Quantization Noise Spectra," *IEEE Trans. Inform. Theory*, vol. 36, No. 6, pp. 1220-1244, Nov. 1990

[14] Gray R.M. and Stockham T.G., "Dithered Quantizers," *IEEE Trans. Inform. Theory,* vol. 39, No. 3, May 1993

[15] Hellendoorn H. and Driankov D., *Fuzzy Model Identification.* Springer, 1997.

[16] Hodgins J.K. and O'Brien J.F., "Computer Animation," *The Encyclopedia of Computer Science,* fourth edition, International Thomson Business Press, 1998.

[17] Jang J-S.R., "ANFIS: Adaptive-Network-Based Fuzzy Inference System", *IEEE Trans. Syst. Man Cybern.*, vol. 23, No. 3, May/June 1993

[18] Jang J-S.R. and Gulley N., *MATLAB Fuzzy Logic Toolbox,* the Math Works Inc., 1997

[19] Jang J-S.R. and Sun C-T, "Neuro-Fuzzy Modeling and Control," *Proc. IEEE,* vol. 83, No. 3, pp. 378-405, Mar. 1995

[20] Jayant N.S. and Noll P., *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984

[21] Jerri A.J., "The Shannon Sampling Theorem - Its Various Extensions ans Applications: A Tutorial Review," *Proc. IEEE,* vol. 65, No. 11, pp. 1565 - 1596, Nov. 1977

[22] Kosko B., *Neural Networks and Fuzzy Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1992

[23] Lea R.N., Jani Y. and Mica J.A., "Fuzzy Logic Control: Basics and Applications," *The Industrial Electronics Handbook*, pp. 1116-1126, 1997

[24] Lee C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller - Part I", *IEEE Trans. Syst. Man Cybern.,* vol. 20, No. 2, Mar./Apr. 1990.

[25] Lee C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller - Part II", *IEEE Trans. Syst. Man Cybern.,* vol. 20, No. 2, Mar./Apr. 1990

[26] Lotto I.D. and Paglia G.E., "Dithering Improves A/D Converter Linearity," *IEEE Trans. Instrum. Meas.*, vol. 35, No. 2, pp. 170 - 177, June 1986

[27] Mamdani E.H. and Assilian S., "An Experiment in Linguistic Synthesis with A Fuzzy Logic Controller," *Int. J. Man-Machine Studies,* vol. 7, No. 1, pp. 1-13, 1975

[28] Marks R.J., *Introduction to Shannon Sampling and Interpolation Theory.* New York: Springer-Verlag, 1991

[29] Miller A.J., Brown A.W. and Mars P., "A Study of An Output Interface for A Digital Stochastic Computer," *Int. J. Electronics*, vol. 37, No. 5, pp. 637 - 566, 1974

[30] *Fuzzy Logic Education Program*, Center of Emerging Computer Technologies, Motorola Inc.

[31] Nyquist, H., "Certain Topics in Telegraph Transmission Theory," *AIEE Trans.*, pp. 617-644, 1928

[32] Oppenheim A.V. and Willsky A.S., *Signals and Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1983

[33] Palm R. and Driankov D., "Fuzzy Inputs," *Fuzzy Sets and Systems 70*, 1995, pp. 315-335

[34] Petriu E.M., Wide P. and Eatherley G., "Sensor Resolution Effect on Fuzzy Logic Controller Behaviour," *IMTC-98,* St. Paul, MI, USA, May 1998

[35] Petriu E.M. and Eatherley G., "Fuzzy Systems in Instrumentation: Fuzzy Control," *IMTC-95*, Waltham, MA, April 1995

[36] Petriu E.M., Cornell A. and Eatherley G., "Fuzzy Control for Mobile Robot Backing-Up," *CONTI'98,* Timisoara, Romania, Oct. 1998

[37] Papoulis A., *Probability, Random Variables, and Stochastic Process*, McGraw-Hill, Inc., 1991

[38] Roberts L.G., "Picture Coding Using Pseudo-Random Noise," *IRE Trans. Inform. Theory,* vol. 8, pp. 145-154, Feb. 1962

[39] Salichs M.A., Moreno L., Gachet D., Escalera A.D.L. and Pimentel J.R., "Mobile Robots," *The Industrial Electronics Handbook*, pp. 773-784, 1997

[40] Shannon, C.E., "Communication in the Presence of Noise," *Proc. IRE,* vol. 37, pp. 10-21, 1949

[41] Schuchman L., "Dither Signals and Their Effect on Quantization Noise," *IEEE Trans. Commun. Technol.,* vol. 12, Dec. 1964

[42] Sripad A.B. and Snyder D.L., "A Necessary and Sufficient Condition for Quantization Errors to be Uniform and White," *IEEE Trans. Acoust. Speech Signal Processing,* vol. 25, pp. 442-448, Oct. 1977

[43] Sugeno M., *Industrial Application of Fuzzy Control*, Elsevier Science Pub. Co., 1985

[44] Vanderkooy J. and Lipshitz S.P., "Resolution Below the Least Significant Bit in Digital Systems with Dither," *J. Audio Eng. Soc.,* vol. 32, No. 3, pp. 106-113, Mar. 1984

[45] Vanderkooy J. and Lipshitz S.P., "Dither in Digital Audio," *J. Audio Eng. Soc.,* vol. 35, pp. 966-975, Dec. 1987

[46] Wagdy M.F., "Effect of Various Dither Forms on Quantization Errors of Ideal A/D Converters," *IEEE Trans. Instrum. Meas.,* vol. 38, No. 4, Aug. 1989

[47] Wagdy M.F. and Goff M., "Linearizing Average Transfer Characteristics of Ideal ADC's via Analog and Digital Dither," *IEEE Trans. Instrum. Meas.,* vol. 43, No. 2, Apr. 1994

[48] Wagdy, M.F. and Ng W.M., "Validity of Uniform Quantization Error Model for Sinusoidal Signals Without and With Dither," *IEEE Trans. Instrum. Meas.*, vol. 38, No. 3, June 1989

[49] White D.R. and Pickup C.P., "Systematic Errors in Digital Cross Correlators Due to Quantization and Differential Nonlinearity," *IEEE Trans. Instrum. Meas.*, vol. 36, pp. 47-53, Mar. 1987

[50] Whittaker E.T., "On the Functions Which Are Represented by the Estimation of Power Spectra," *Proc. R. Soc. Edinburgh,* vol. 35, pp. 181-194, 1915

[51] Widrow B., "A Study of Rough Amplitude Quantization by Means of Nyquist Sampling Theory," *IRE Trans. Circuit Theory,* vol. 3, No. 4, pp. 266-276, Dec. 1956

[52] Widrow B., "Statistical Analysis of Amplitude-Quantized Sampled-Data Systems," *Trans. AIEE (Applications and Industry),* vol. 79, 1960 (Jan. 61' section)

[53] Widrow B., I. Kollar and M.C. Liu, "Statistical Theory of Quantization," *IEEE Trans. Instrum. Meas.*, vol. 45, No. 2, pp. 353-361, Apr. 1996

[54] Zadeh L.A., "Fuzzy Sets," *Informat. Control,* vol. 8, pp. 338-353, 1965

[55] "VRML 97 Specifications", http://www.vrml.org/

[56] "External Authoring Interface reference," http://cosmosoftware.com/products/player/developer/eai/

Appendix

Simulink - the Simulation Environment

As a toolbox in MATLAB, Simulink has something special compared with other application toolboxes in MATLAB, which is that it is built atop MATLAB. As a result, Simulink users have direct access to the wide range of MATLAB-based tools for generating, analyzing, and optimizing systems implemented in Simulink. These tools include MATLAB Application Toolboxes, specialized collections of M-files for working on particular classes of problems.

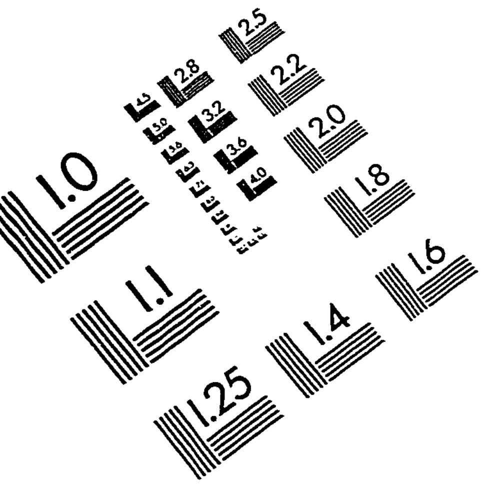
Simulink is a software package for modeling, simulating and analyzing dynamical systems. It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Systems can be also multirate, i.e., have different parts that are sampled or updated at different rates.

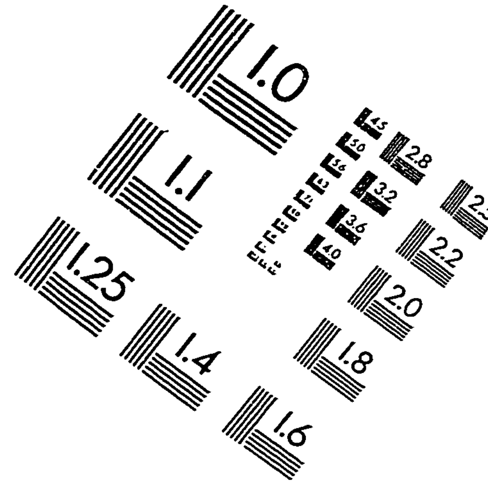
For modeling, Simulink provides a graphical user interface (GUI) for building models as block diagrams, using click-and-drag mouse operations. With this interface, you can draw the models just as you would with pencil and paper. This is a big advantage over the previous simulation packages that require users to formulate differential equations and difference equations in a language or program. Simulink includes a comprehensive block library of sinks, sources, linear and nonlinear components and connectors. Moreover, the users can also customize and create their own blocks.

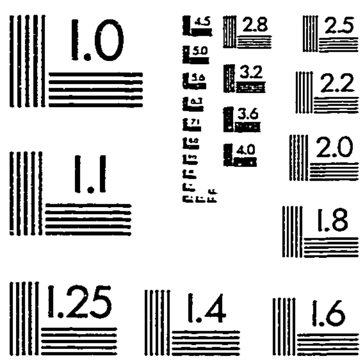
Models are hierarchical, so the user can build models using both top-down and bottom-up approaches. One can view the system at a high-level, then double-click on the block to go down through the levels to see increasing levels of model detail. This approach provides insight into how a model is organized and how its parts interact.

After defining a model, one can simulate it, using a choice of integration methods, either from the Simulink menus or by entering commands in MATLAB's command window. The menus are particularly convenient for interactive work, while the command-line approach is very useful for running a batch of simulations. Using scopes and other display blocks, one can see the simulation results while the simulation is running. In addition, one can change parameters and immediately see what happens, for "what if" exploration. The simulation results can be put in the MATLAB workspace or MAT file for post processing and visualization.

IMAGE EVALUATION TEST TARGET (QA-3)

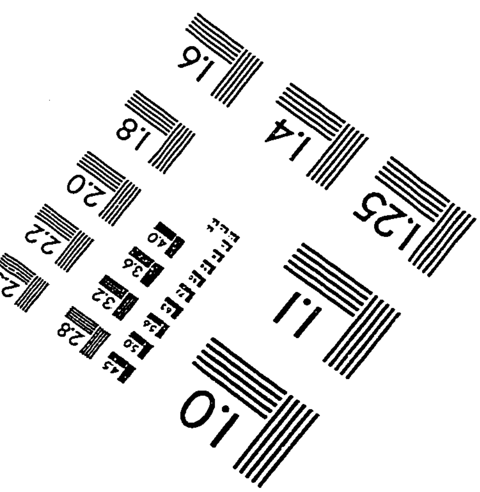


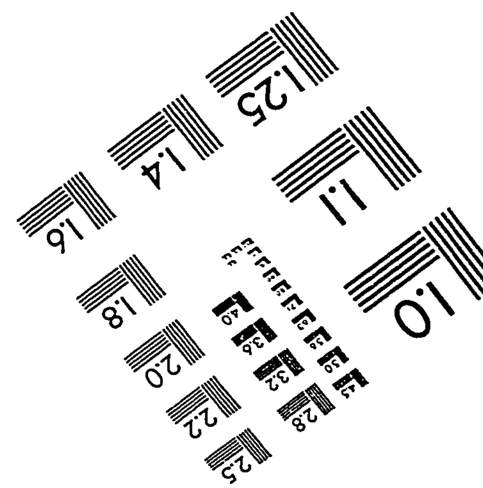




150mm

6"





APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989