NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

ISBN 0-315-56415-6

CORRIDOR ASYNCHRONOUS DELTA MODULATION

BY

TERRY TAI-WING KWAN

A Thesis
presented to the School of Graduate Studies
and Research, University of Ottawa
in partial fulfillment of the
requirements for the degree of
Master of Applied Sciences (M.A.Sc.)
in
ELECTRICAL ENGINEERING

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

ABSTRACT

An algorithm based on delta modulation for the transmission of nonuniformly sampled signals through digital channels by time quantizing and block coding of the sampling intervals has been investigated and simulated on a digital computer. The trade-offs between the system parameters such as the unit time quantum, the run-length constraint, the buffer size, the transmission rate and signal distortion have been analyzed and evaluated.

Two different schemes, one using run-length codes and the other using vector quantization, are proposed for encoding the signal samples together with the sample interval which are generated by a non-linear (non-uniformly sampled) delta modulator. The performance of the two schemes is evaluated. The algorithms have been tested with computer-simulated speech signals. A 3 to 5 dB improvement in segmental-signal-to-quantization-noise (SSQNR) is obtained over conventional non-linear delta modulation techniques, operating in the same range of transmission rate, with the use of run-length codes. A bit rate compression ratio of magnitude of more than 2 is achieved with the application of vector quantization compared to the results obtained with the run-length codes in the medium transmission rate of 16 to 24 kbits/second.

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my thesis supervisor, Professor W.S. Steenaart, for his generous encouragement, understanding and invaluable guidance throughout this work, without which this thesis would have not been possible.

I would like to thank Professor B. Sankur for his very helpful suggestions, comments, discussions and encouragement for the early part of this work.

And last but not least, the author is highly indebted to his wife Roseline for her enormous patience and constant encouragement during this period.

To my wife Roseline, my sons Maxime and Olivier

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

1.0 INTRODUCTION

With the rapid development of the technology of high speed integrated digital circuits, digital techniques have become standard in commmunication systems. These systems offer certain advantages over their analog counterparts, such as an ability to operate at lower signal-to-noise ratios, power requirements, cost and yet smaller size. The major disadvantage of digital transmission is the requirement of a wider transmission channel bandwidth. Hence, data compression is required to reduce the necessary transmission bandwidth. The development of optimal data compression techniques has become the main task in the design of digital communication systems.

Data compression can be achieved either by performing a block transformation on the data, and then quantizing in the transform domain [1] (Transform Coding), for example, the Fourier, Hadamard, discrete cosine, or Karhunen-Loeve transforms, or by predictive or interpolative methods in the time domain (Predictive Coding).

It is often difficult to choose a specific data compression technique for a given application, because the quality of the algorithm depends on the statistical properties of a given source. Most of the comparison techniques investigated so far perform data compression on

uniformly sampled signals [3][4][11]. In such cases, there is always a risk of oversampling or undersampling the signals of unknown or time varying statistics, (speech and image signals). If predictive (waveform) nonuniform sampling techniques are employed, then only the sampling intervals and the prediction polarities are needed to be transmitted, since the predictive rule is also known at the receiving end [2]. As a result, the transmission rate can be reduced considerably. Although some source encoding systems [2][3] have been proposed with the latter techniques, no complete analysis and application has been developed. It is the goal of this thesis research to derive such an encoding scheme. Figure 1.1 shows the approximate performance levels of several quantization techniques used for speech coding [4].

In this thesis, a time domain model of a comparison digital encoding system has been developed and simulated. This system is based on a delta modulation technique with a nonuniformly sampling method where information is generated only on those samples which differ from their predictions by a certain preselected threshold. Buffering is employed due to nature of the asynchronous sampling technique. The encoding procedure can be factored into two separate steps, the source coding and the channel coding. Studies of data compression often lump the channel encoder with the communication link. To evaluate the signal encoder itself,

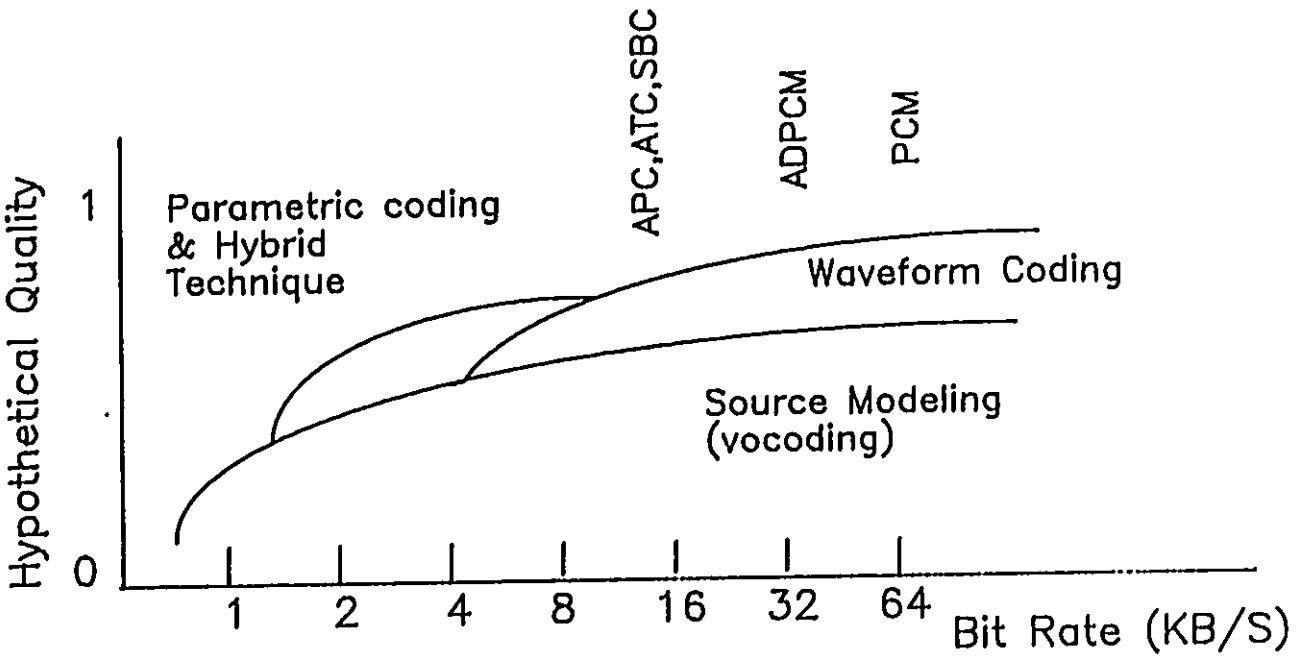


Figure 1.1 Quality vs Bit Rate for Speech Coding

a noiseless channel is often assumed. This study is focused on the problem of source coding while assuming that the channel errors contribute no degradation to the system performance. In practice, an error correcting code is likely to be employed for the recovery of the signal in noisy channels.

To verify the validity of the simulation method, the system performances are compared to different conventional source encoding systems (Constant Factor Delta Modulation, CFDM [14][42], Linear Delta Modulation, LDM [6]) with the same input signals, which in this case is synthesized speech.

The following paragraphs present an overview of the thesis.

In chapter two, a general model of an asynchronous source encoding scheme is outlined. The minimum bandwidths required for transmission are derived for both sinusoidal and Gaussian signals. The adaptation schemes in generating nonuniform information flow are also described. The sampling information is transmitted with the aid of a store-and-forward device, such as a buffer.

In chapter three, the proposed corridor asynchronous delta modulation system is presented. Following the system description, the sampling interval statistics are derived for the asynchronous system. The transmission rate, the

channel capacity, and various sources of distortion, such as quantization noise, buffer overflow and run-length limits, inherent to the system are presented.

In chapter four, the application of run-length coding of the nonuniform sampling intervals of the asynchronous delta modulator is presented. The application of run-length encoding with Huffman code, differential run-length code and differential run-length code with synchronization word are investigated.

In chapter five, the application of widely popular vector quantization is described. The application of vector quantization to the nonuniform sampling intervals generated by the asynchronous delta modulator is introduced. The performance is then compared with that obtained by using run-length coding described in chapter four. A noticeable improvement, in terms of transmission rate reduction, is obtained by the use of vector quantization.

The simulation results are discussed in chapter six, where comparisons of the different coding schemes on the system and performances with LDM and CFDM are made.

This thesis is concluded by a suggestion for what the author feels to be an interesting field.

CHAPTER II

2.0 ASYNCHRONOUS SOURCE ENCODING SYSTEMS

2.1 INTRODUCTION

In conventional coding methods like Pulse Code Modulation (PCM), and Differential Pulse Code Modulation (DPCM), the sampling interval is fixed and restricted to an upper limit of $1/2f_{max}$, where f_{max} is the highest frequency component of the modulating signal, according to the Nyquist sampling theorem [5]. It would be interesting to devise a system where the sampling intervals could be made to vary to take advantage of the redundancy of the signals. Hence, the number of samples to be transmitted would potentially be reduced, and a substantial bit rate reduction would be achieved.

It is the aim of this research work to realize data compression with the transmission of nonuniformly sampled data over the synchronous binary channel. In this chapter a general description of asynchronous source encoding systems is outlined. The output of such a system consists of positive and negative pulses nonuniformly spaced in time, and the information required for transmission is the pulse interval duration and the polarity of each pulse.

Bandwidth compression can be achieved by reducing the redundancy of the encoded signal. The results obtained are

shown in Chapter 3. The higher the correlations between the message samples, the greater the redundancy, and in turn, the required information rate to be transmitted can be reduced. In asynchronous source encoding systems, information is transmitted only when required, according to a particular criterion which aims to remove or reduce redundancy [6].

2.2 ASYNCHRONOUS SOURCE ENCODING SYSTEM

The general block diagram of an asynchronous source encoding system is shown in Figure 2.1.

The difference signal, e(t), between the input signal, x(t), and the reconstructed signal, y(t), is continuously updated and sampled at appropriate instants according to the sampling control algorithm. At the nonuniformly sampling instant, t_i, a positive or negative pulse is generated and conveyed for transmission,

$$b(t_i) = \pm 1 \qquad (2.1)$$

where $b(t_i)$ is the pulse polarity at time t_i

The reconstructed signal y(t) is generated by a function derived from the information on the duration of the

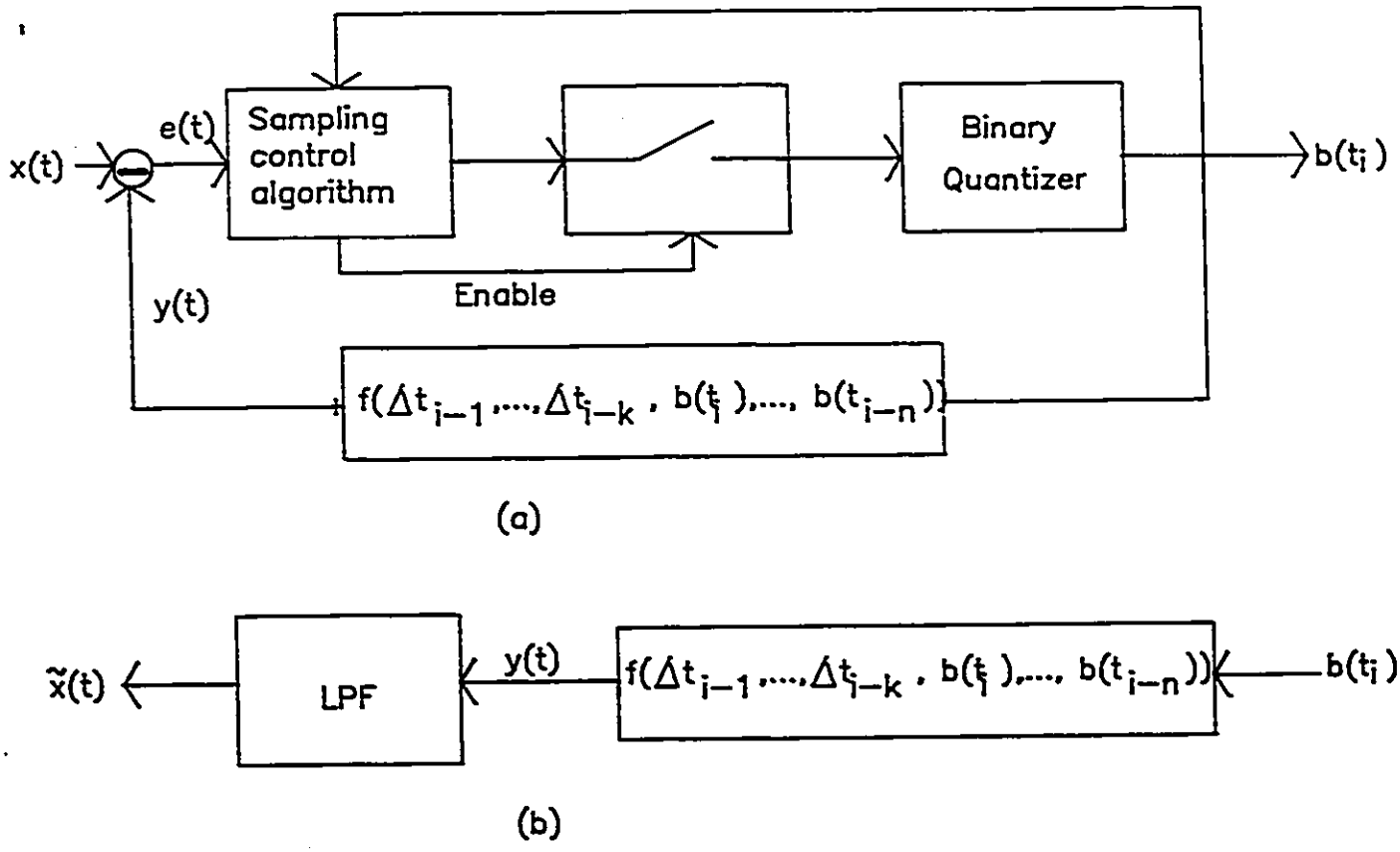


Figure 2.1 Asynchronous Source Encoding and Decoding System
(a) Encoder, (b) Decoder

preceding intervals and the pulse polarities, i.e.

$$y(t) = f(\Delta t_{i-1}, \ldots, \Delta t_{i-(n+1)}, b(t_i), \ldots, b(t_{i-n}))$$

for $t_i \leq t \leq t_{i+1}$

where $\Delta t_{i-1} = t_i - t_{i-1}$ (2.2)

In practice, the sampling control algorithm generally consists of a simple threshold detector applied to the difference signal, $e(t) = x(t) - y(t)$. For the case of fixed thresholds, e.g. at values of $-a/2$ and $+a/2$, the system is called a fixed corridor type asynchronous source encoding system with corridor width, "a", which is also called the step-size [7].

2.2.1 Asynchronous Delta Modulation Systems

Due to the simplicity of realization of Delta Modulation (DM), numerous refinements and variations [3] of the basic invention [8] have been developed. Asynchronous Delta Modulation (ASDM) systems, which can be easily implemented, have become the best known asynchronous source encoding systems. ASDM systems have a digital output which is quantized in amplitude but not in time. Consequently uniform sampling processes are not required. The system arrangement is shown in Figure 2.2.

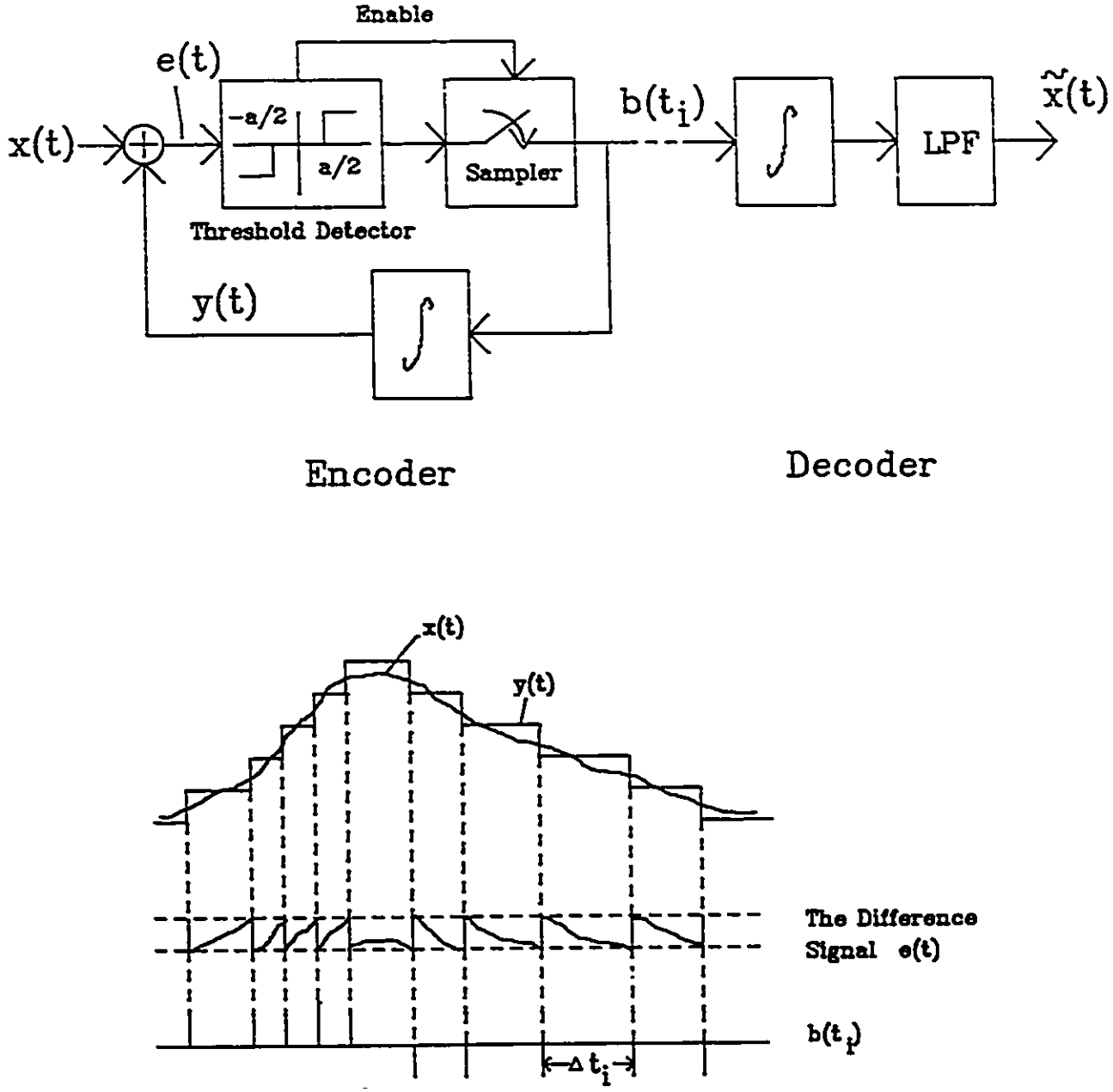


Figure 2.2 The ASDM System

In Figure 2.2, the encoder generates an output waveform b(t) consisting of short pulses with magnitude ± 1. Each pulse is generated when the variation of the signal exceeds the basic [-a/2, +a/2] interval. The polarity is determined from the difference signal e(t) by applying the following rules:

$$\begin{aligned} b(t) &= +1, \text{ if } e(t) \geq a/2 \\ &= -1, \text{ if } e(t) \leq -a/2 \\ &= 0, \text{ if } -a/2 \leq a(t) \leq a/2 \end{aligned} \qquad (2.3)$$

In the feedback loop of figure 2.2, the integration of the generated pulse trains yields the staircase approximation, having irregular horizontal intervals, of the input signal.

To obtain satisfactory (Nyquist I) transmission, the minimum bandwidth, B_{min}, required is $2/\Delta t_{min}$, where Δt_{min} is the minimum value of Δt. Note that $\Delta t_{min} = a/|x'(t)|_{max}$, where $|x'(t)|_{max}$ is the maximum absolute slope of x(t).

Consider a sinusoid x(t) equal to $A\sin 2\pi f_m t$. In a quarter of a period, the number of steps required to reach the amplitude of |A| from zero is equal to A/a, where a is the step size of the delta modulator. Thus, the number of

pulses generated in one period is 4A/a, or the average number of steps per second is

$$E(n) = \frac{4A\ 2\pi f_m}{a\ 2\pi} = \frac{4A\ f_m}{a} \qquad (2.4)$$

where E(.) is the expectation operator.

The average absolute slope of x(t) is

$$E(|x'(t)|) = 4A\ f_m \quad [\text{sec.}^{-1}] \qquad (2.5)$$

and the average number of steps per second is

$$E(n) = \frac{E(|x'(t)|)}{a} \qquad (2.6)$$

If buffering is assumed, then the minimum bandwidth, B_{min}, required for this sinusoidal signal becomes $4Af_m/a$.

In the case of a zero mean Gaussian input signal x(t) band-limited to f_{max} with mean square value σ^2, the probability density function of the slope of x(t) is also a Gaussian function with a mean square value given in [10].

$$E\{(x'(t))^2\} = 4\pi^2 \int_0^{f_{max}} f^2 \frac{\sigma^2}{f_{max}}\, df = (2\pi f_{max})^2\, \sigma^2/3 \qquad (2.7)$$

The average absolute slope, $E(|x'(t)|)$ is found to be $\sqrt{8}/3\pi \cdot f_{max}$ [10]. The average pulse rate can be found by substituting this value into (2.6). Thus

$$E(n) = \frac{\sqrt{8}\pi/3 \cdot f_{max}}{a} \tag{2.8}$$

From (2.7), if the maximum slope is assumed to be N times the root-mean-square value of the derivative, i.e. the square root of (2.7), then

$$|x'(t)|_{max} = N(2\pi f_{max})/\sqrt{3} \tag{2.9}$$

and consequently the minimum bandwidth:

$$B_{min} = \frac{2|x'(t)|_{max}}{a} = \frac{2N}{\sqrt{3}\,a} 2\pi f_{max} \tag{2.10}$$

if no buffering is assumed.

Considering the fact that $2/\Delta t_{min}$ is the necessary transmission bandwidth and $1/E(\Delta t)$ is the average pulse transmission rate, one can use the ratio $E(\Delta t)/\Delta t_{min}$ or $|x'(t)|_{max}/E(|x'(t)|)$ to represent the efficiency of the channel utilization. It is found that (recall (2.9), (2.8), (2.5) and (2.4)) the ratio for a Gaussian signal is $N\sqrt{\pi/2}$ and that for a sinusoidal signal is $\pi/2$. For a Gaussian signal with N greater than $\sqrt{\pi/2}$, it is necessary to find an

algorithm for further band-compression rather than transmit the output pulse itself.

2.2.2. Adaptive Schemes

Recall (Figure 2.2) that the sampling instants are determined solely by the threshold detector. However, more promising performance improvements are likely to be achieved by adaptive sampling techniques. Hawkes and Simonpieri [2] have described a sampling algorithm for ASDM systems, where the sampling interval is a function of the past intervals. It is necessary to limit the maximum and minimum sampling interval lengths, which are denoted as Δt_M and Δt_m respectively.

The sampling interval is determined by applying the following rules:

$$\Delta t_i = \begin{cases} \Delta t_{i-1}\, F(b(t_i), b(t_{i-1}), \ldots); & \text{if } \Delta t_m \le \Delta t_i \le \Delta t_M \\ \Delta t_m & ; \text{ if } \Delta t_i < \Delta t_m \\ \Delta t_M & ; \text{ if } \Delta t_i > \Delta t_M \end{cases} \quad (2.11)$$

where F is a specific function.

In the simplest form, memorization would be limited to only the previous sample. Thus F depends on b(t) and $b(t_{i-1})$ only. A reasonable logic for F would be:

$$F(+1,+1) = F(-1,-1) = A \le 1$$

$$F(+1,-1) = F(-1,+1) = B \geq 1 \qquad (2.12)$$

In order to follow rapid changes in the signal, a change in the step size can also be adopted.

Jayant [11] has described a step-size adaptation scheme based on a fixed rate delta modulation system. The step size Δ can be generally represented as:

$$\Delta_i = \begin{cases} \Delta_{i-1}\, G(b(t_i),\, b(t_{i-1}),..) & , \text{ if } \Delta_m \leq \Delta_i \leq \Delta_M \\ \Delta_m & , \text{ if } \Delta_m > \Delta_i \\ \Delta_M & , \text{ if } \Delta_i > \Delta_M \end{cases} \qquad (2.13)$$

where Δ_M and Δ_m are the maximum and minimum step sizes respectively and G is a specific function.

Note that Δ_M should be limited to avoid excessive signal overshoot while Δ_m should correspond to the minimum quantization interval.

(Weiss et al [12] and Oliver [13] have discussed a method of overshoot suppression for delta modulation systems. This method and is described in chapter six.)

In a similar way as for (2.12), if G depends on b(t) and $b(t_{i-1})$ only, then the logic G can be stated as follows:

$$G(+1,+1) = G(-1,-1) = C > 1$$
$$G(+1,-1) = G(-1,+1) = D < 1 \qquad (2.14)$$

2.3. UNIFORMLY SAMPLED COMPARISON TECHNIQUES

Because of their similarity with the systems discussed above, the zero-order interpolator and the fan interpolator [42], both well known data compression schemes, will be mentioned briefly.

The Zero-Order Interpolator: First the signal is uniformly sampled and a horizontal aperture is set around the samples in such a way that the number of samples staying in the aperture space is maximized. After the last sample, the aperture is moved vertically to a new value as shown in Figure 2.3.

The compressed message includes the aperture levels and the number of samples that stay within the aperture before it is readjusted. One variation of such an interpolator is described in [14], which proposes a finite length, exponential converging-aperture procedure for speech coding at an average bit rate of 9.6 to 14.4 kb/s.

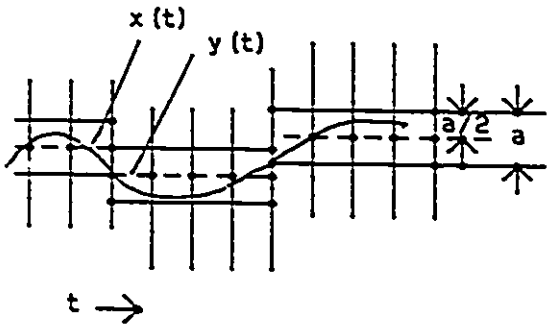


Figure 2.3 The Zero Order Interpolator

The Fan Interpolator: The fan interpolator operates in a similar way except that the aperture is not constrained to be horizontal. It tries to approximate the signal by the longest straight line joining two sample points in such a way that all intermediate sample values fall within the aperture shown in Figure 2.4.

As shown in figures 2.3 and 2.4, the fan interpolator is sensitive to slope changes whereas the zero order interpolator is sensitive to amplitude changes.

In the next chapter, the analysis of the ASDM system is presented. The polarities of the output pulses of the encoder and the durations between them are run-length encoded so as to achieve synchronous transmission.

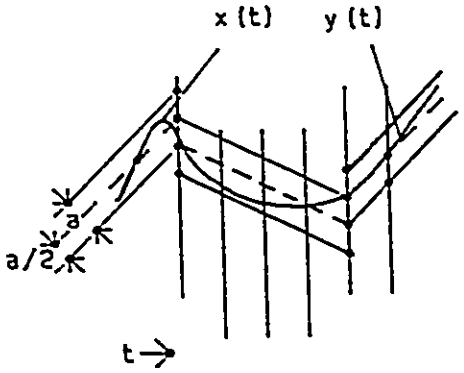


Figure 2.4 The Fan Interpolator

This Page is Intentionally Left Blank

CHAPTER III

3.0 ASYNCHRONOUS DELTA MODULATION SYSTEMS

3.1 INTRODUCTION

In this chapter, the proposed asynchronous delta modulator is described from a practical point of view. This is followed by the derivation of the probability density function (pdf) of the sampling intervals of an ASDM. The required transmission rate, the quantization noise characteristics of the system and the channel bandwidth required for transmission are also analyzed.

The commonly used data compression algorithms, such as the zero-order interpolator and the fan interpolator as described in chapter 2, require the transmission of nonredundant sample values, and the time information to locate them. However, in an asynchronous source encoding system, instead of transmitting the quantized level of each nonredundant sample, the differences between the samples are transmitted. Thus, only one bit is required to represent the polarity of the output pulse.

To operate on irregular sampling intervals, it is necessary to include information such as run-length, in order to permit the reconstruction of the transmitted signal. It is advantageous to run-length encode the time intervals of the asynchronous source encoding system output

in two ways. First, synchronous transmission can be achieved by the use of a buffer. Secondly, long run-lengths can be encoded to represent long idle periods, such as conversation pauses or silence periods in speech signals and long black and white scans in video signals. Considerable reduction is attainable in transmission rates for run-length encoding in multiplexed channels.

Scuilli [15] has experimentally determined the run-lengths of the quantized amplitude values for speech signals sampled at 8 KHz and quantized to 64 levels, as seen in Figure 3.1.

In a practical data compression system, the information of the compressed data is usually generated nonuniformly. In the case where run-length encoding is employed, the code words are generated at the end of the intervals to be encoded. Hence, the compressed data has to be fed into a buffer so that it can be transmitted at a uniform rate and in the right order.

3.2. SYSTEM DESCRIPTION

The block diagram of the complete data compression is shown in Figure 3.2.

The output of the asynchronous source encoder consists of nonuniformly spaced positive and negative pulses. The

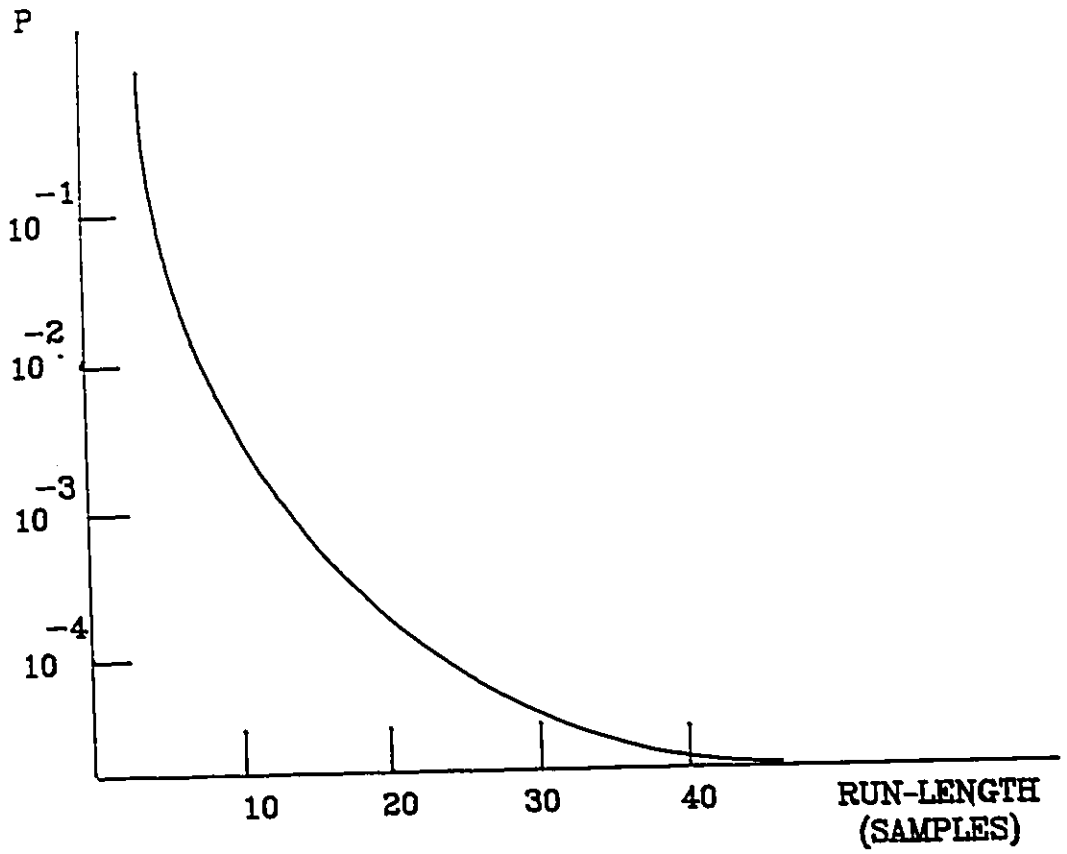


Figure 3.1

Typical Run-Length pdf of a Speech Signal

*From reference [15]

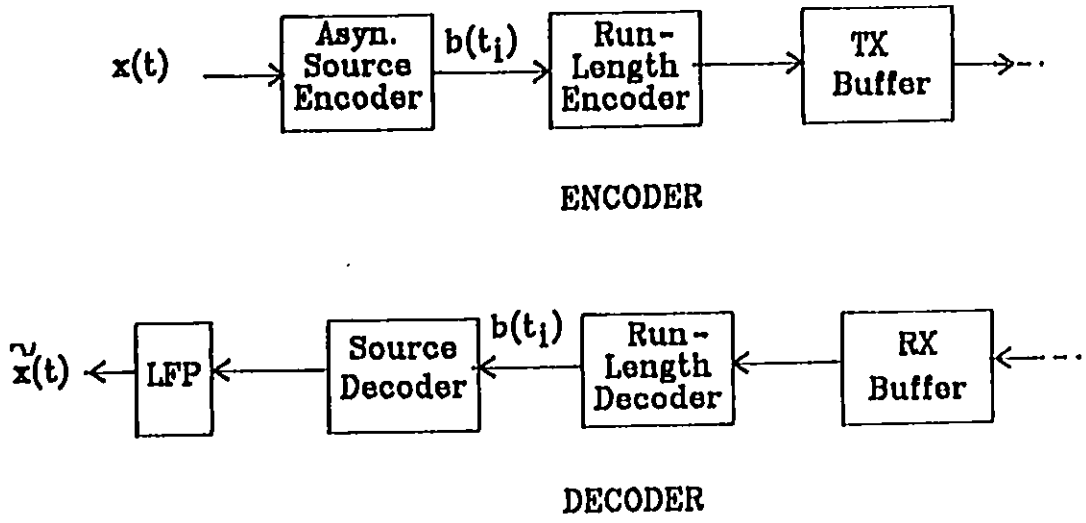


Figure 3.2 Source Encoding System for Non-Uniformly Sampled Signals

counter control triggers and resets the two counters alternately during successive pulse intervals. The run-length codes assigned for the time intervals appear at the counter outputs. One period of the counter clock, τ sec. corresponds to a run-length of one. In our system simulation, as described in chapter six, τ is chosen to be equal to Δt_{min}.

Figures 3.3 and 3.4 shows the run-length encoder and decoder respectively. In the encoder, the data selector at counter output selects the run-length code words to feed into the buffer. The asynchronous source encoding system output consists of both positive and negative pulses to indicate upward or downward steps. A sign bit (information bit) is carried along with each code word.

The counter frequency generators of the encoder and decoder each oscillate at the same frequency. When the decoder counter matches the word in the buffer, the comparator triggers the source decoder. Subsequently, the word is discarded from the buffer and the counter is reset.

The demodulated signal is subsequently passed through a low-pass filter (LPF) to remove the edges due to quantization.

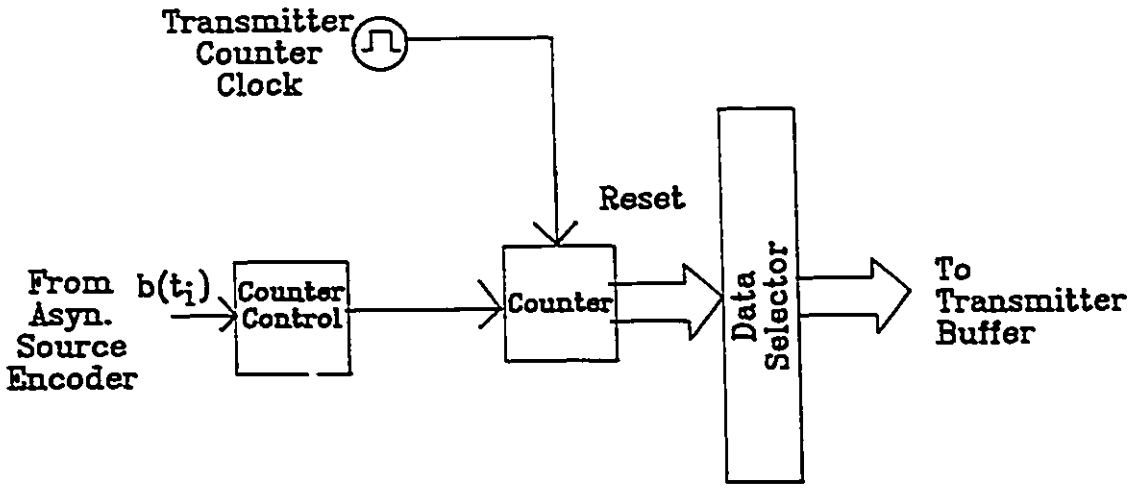


Figure 3.3 Run-Length Encoder

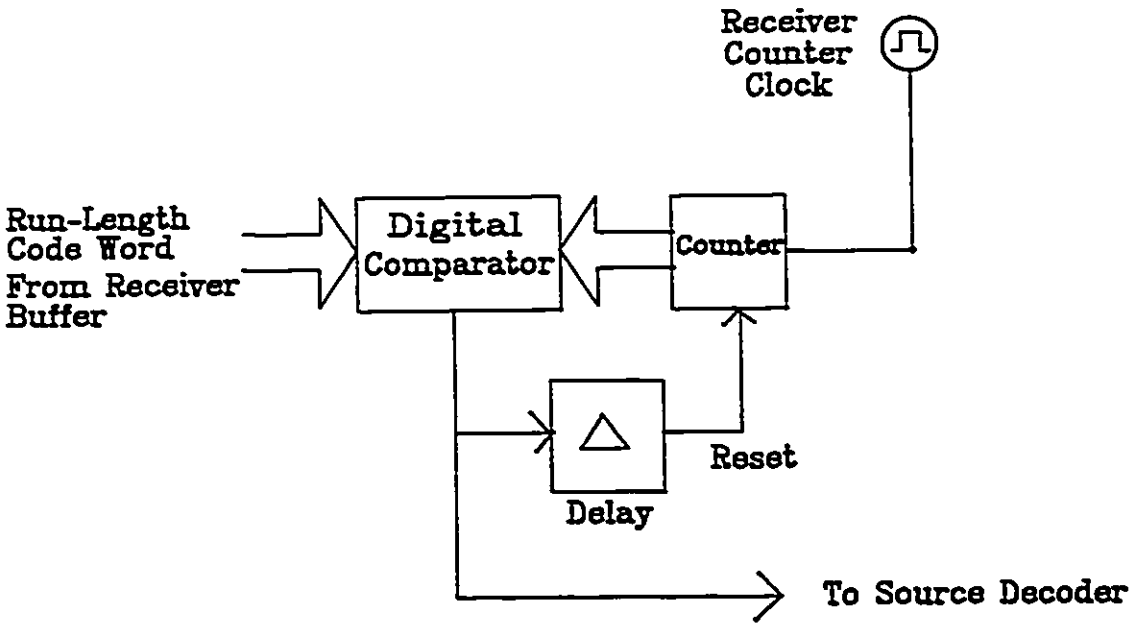


Figure 3.4 Run-Length Decoder

3.3. RUN-LENGTH ENCODING ERRORS DUE TO THE RUN-LENGTH CONSTRAINT

If the minimum inter-bit-interval time (IBI), Δt_{min}, is the run-length of one, then the minimum length of the code word, l_c (bits), would be the smallest integer greater than $\log_2(\Delta t_{max}/\Delta t_{min})$, where Δt_{max} is the longest IBI allowed, so that all time intervals between arrivals can be encoded correctly. In the case of asynchronous delta modulated speech, excessively long IBI's are likely to appear during the silence periods. Therefore, it would be more appropriate to consider Δt_{max} for the active speech regions.

When a certain time interval Δt_i is longer than Δt_{max}, an encoding error occurs. This type of error can be prevented by employing the encoding scheme shown in Figure 3.3. In this model, the counter starts to recycle when the interval length exceeds Δt_{max}, and Δt_i is assigned to the code word corresponding to $\Delta t_i - k\Delta t_{max}$, where k is an integer. Hence, the transmitted interval length becomes Δt_i modulo(Δt_{max}). In such a case, two kinds of error due to run-length constraint may occur:

(1) If the buffer storage is not emptied during the long time interval, the interval measurement error will be $k\Delta t_{max}$ seconds, where k is the number of Δt_{max} counted before the buffer is empty.

(2) If the storage is emptied, then the time intervals decoded by the receiver will be equal to the

durations inwhich the buffer remains empty. In other words, the received time interval is S_i/R_o sec., where S_i is the number of locations occupied in the buffer at the start of Δt_i and R_o is the transmission rate. Consequently, the interval measurement error will be $\Delta t_i - (S_i/R_o)$ sec.

3.4 CHARACTERISTICS OF THE INTER-BIT-INTERVAL (IBI) DISTRIBUTION

The statistical behavior of the interval time between two consecutive samples, Δt, is one of the factors affecting the performance of the system. In an ASDM system, Δt is the period between two consecutive corridor crossings. The analysis of this case can be based on the previous studies of level-crossing problems [17]. However, due to the mathematical difficulties confronted in level-crossing problems, analytical derivations of the IBI are available only for a few kinds of signal distributions [17].

In this section, the IBI characteristics of Gaussian and Laplacian distributed signals are derived.

Consider a zero-mean Gaussian signal. The average number of level crossings in 1 second is given by Rice's formula [18]:

$$E(n_j) = E(n_o) \exp - \{\frac{(2j-1)^2 a^2}{8 R(0)}\} \qquad (3.1)$$

n_j is the number of level crossings by the signal of the j'th level, which is of amplitude a·j, each level having a separation of 'a'. $E(n_o)$ is the average number of zero crossings, and R(0) is the zero-autocorrelation coefficient of the signal, and is the signal power.

The probability density function, pdf, of Δt, $p(\Delta t)$, can be determined approximately from the pdf of the first derivative of the source signal, given that

$$|x'(t)| \cong \frac{a}{\Delta t_i} \quad \text{or} \quad \Delta t_i \cong \frac{a}{|x'(t)|} \qquad (3.2)$$

where Δt_i is the i'th interval in time.

This approximation is based on the assumption that the signal is linear during the inter-bit-interval (IBI), which is acceptable for small values of Δt_i and a, and also that there is no maximum or minimum within the Δt_i period.

The relation between Δt and $x'(t)$ expressed in (3.2) is a double-sided function as shown in Figure 3.5 and $p(\Delta t)$ can be expressed as

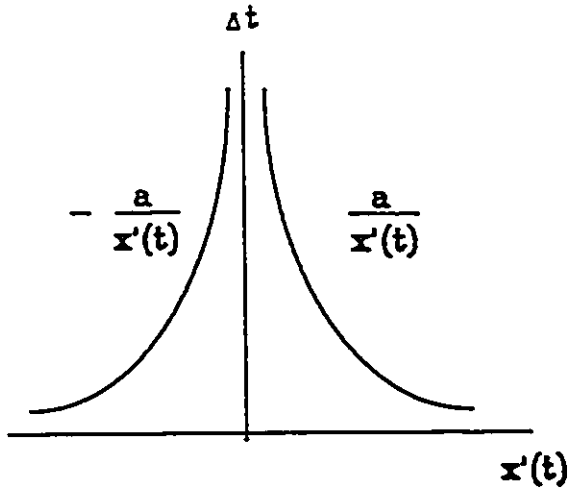


Figure 3.5 Relationship between Δt and $x'(t)$
Δt = Interbit–bit–interval time
$x'(t)$ = slope of input signal
a = step size

$$p(\Delta t) = P[x'(t)]\left(\frac{-a}{\Delta t}\right)\frac{d}{d(\Delta t)}\left(\frac{-a}{\Delta t}\right) + P[x'(t)]\left(\frac{a}{\Delta t}\right)\frac{d}{d(\Delta t)}\left(\frac{a}{\Delta t}\right) \qquad (3.3)$$

Considering that pdf of the slope is symmetric for positive and negative values as in the zero-mean Laplacian and Gaussian distributions, (3.3) can be simplified as

$$p(\Delta t) = \frac{2a}{(\Delta t)^2} P[x'(t)]\left(\frac{a}{\Delta t}\right) \qquad (3.4)$$

Therefore, for signals with a zero-mean Laplacian distribution of

$$P[x'(t)](\xi) = \frac{1}{\sqrt{2}\sigma_o}\exp\{-\sqrt{2}|\xi|/\sigma_o\}, \quad -\infty < \xi < \infty$$

the pdf of Δt is then (by substituting the equation above into (3.4)),

$$p(\Delta t) = \frac{\sqrt{2}\,a}{(\Delta t)^2\sigma_o}\exp\{-\frac{\sqrt{2}\,a}{\Delta t\ \sigma_o}\}, \quad 0 \le \Delta t < \infty \qquad (3.5)$$

Now consider a band limited, zero-mean Gaussian distributed signal with a slope that is also Gaussian distributed [16] and the same power σ_o, where

$$P[x'(t)](\xi) = \frac{1}{\sigma_o \sqrt{2\pi}} \exp\{-\frac{\xi^2}{2\sigma_o^2}\} \qquad (3.6)$$

By substituting equation (3.6) into equation (3.4), we obtain

$$p(\Delta t) = \frac{2a}{(\Delta t)^2/2\pi\ \sigma_o} \exp\{-\frac{a^2}{2(\Delta t)^2\ \sigma_o^2}\} \qquad (3.7)$$

Since a/σ_o is the only variable in (3.5) and (3.7), and the corridor width can always be expressed as a function of the standard deviation of the signal, one can conclude that the only parameter that the IBI depends on is the corridor width, "a".

In Figure 3.6 the normalized pdf of the IBI are drawn for different width factors, a/σ_o, using the results obtained in (3.5) and (3.7). The Laplacian slope approximation is found to have a higher average IBI compared to that of the Gaussian. In other words, the Gaussian approximation is better for shorter IBI's or steeper slope, whereas the Laplacian approximation is better for longer IBI's or the tails of the slope.

3.5 TRANSMISSION RATE

In this section, a control algorithm for the transmission buffer for the proposed asynchronous delta

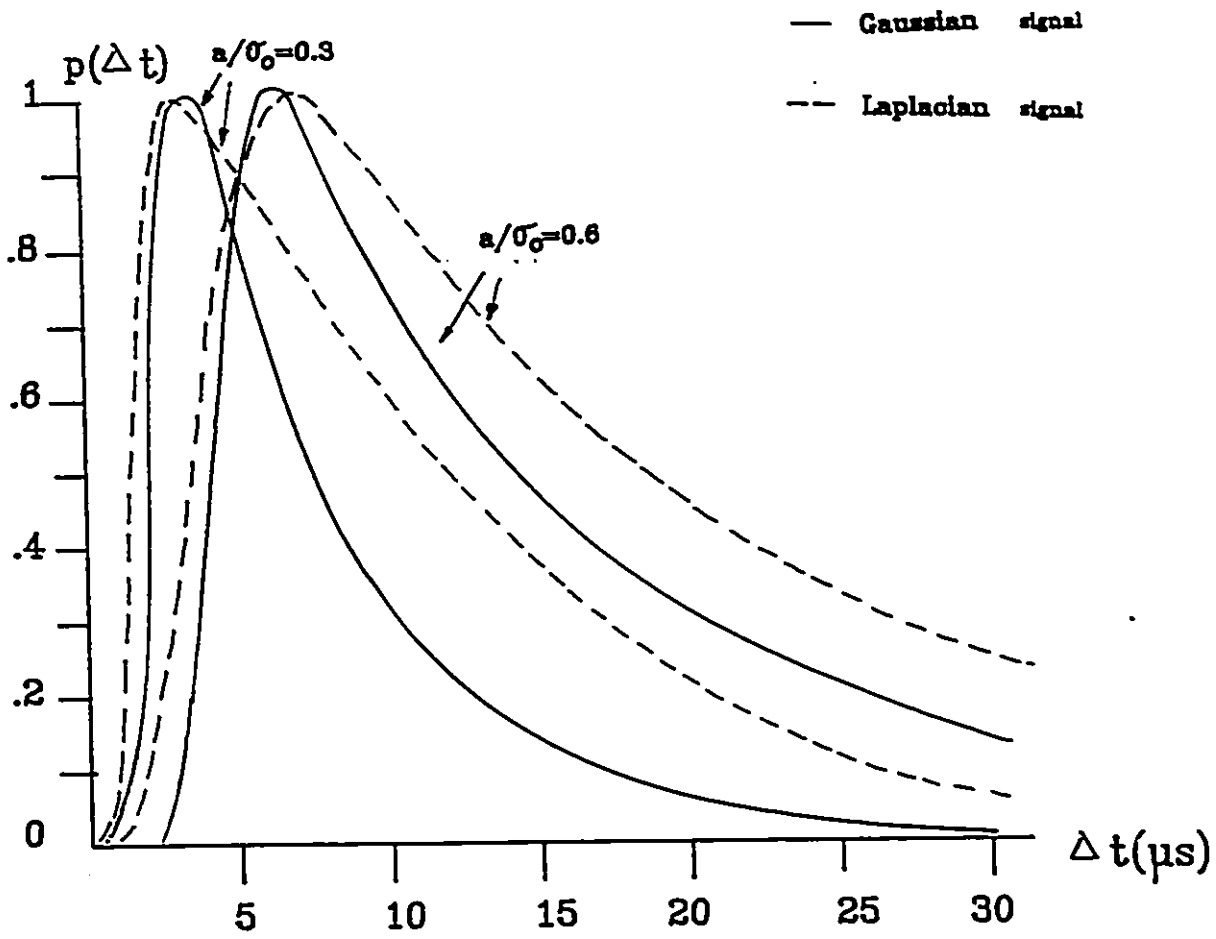


Figure 3.6 The PDF of the IBI for Gaussian and Laplacian Slope Distribution

modulator is investigated. The transmission rate chosen affects directly the buffer occupancy level distribution. If the transmission rate R_o is chosen to be greater than the average pulse transmission rate, $1/\overline{\Delta t}$, then the buffer is expected to underflow frequently, whereas if $R_o < 1/\overline{\Delta t}$ then it will result in frequent buffer overflow. Therefore, the transmission rate is a compromise between the overflow and underflow conditions.

If the transmission rate R_o is chosen to be equal to $1/\overline{\Delta t}$, then the probabilities of buffer underflow and overflow are the same. However, it is more desirable to chose R_o which is slightly greater than $1/\overline{\Delta t}$, since the underflow situation is more tolerable than that of overflow [19].

The average input rate into the buffer, $1/\overline{\Delta t}$, can be found if $p(\Delta t)$ is known. This was derived for asynchronous delta modulated speech considering only the active regions. However, one must also consider the silence intervals which exist for as much as 50% of the time.

Exponential functions have been used to model the talk spurts and the silence gaps. Examination of the speech signal has revealed that besides the numerous short gaps, there exist gaps as long as 1 to 2 seconds. This suggested that the silence intervals can be modeled by a combination of two exponential pdf's with means approximately 10 msec. and 1 second [20].

$$p_g(t) = Ab_1 \exp(- b_1 t) + (1-A)\, b_2 \exp(- b_2 t) \qquad (3.8)$$

$p_g(t)$ is the total probability of occurrence of gaps, 'A' is the probability of occurrence of short gaps, whereas '1-A' is the probability of occurrence of the long gaps. It is found by Sherman [20] that only one out of 20 gaps is a long one, that is A = 0.95, and the average gap length is

$$\frac{1}{b} = \frac{A}{b_1} + \frac{(1-A)}{b_2} \qquad (3.9)$$

that is, around 50-60 msec.

The talk spurts also have an exponential density function with an average duration of 50 msec.

$$p_s(t) = c \exp\{- ct\} \qquad (3.10)$$

The ratio of the silence gaps to the entire talk period is:

$$c/\ (c+b) = 50\% \qquad (3.11)$$

Then the expected valued of (Δt) to be used in the calculation for the transmission rate becomes

$$E\{\Delta t\} = \Delta t_a\ (1 + c/b) \qquad (3.12)$$

where Δt_a is the average time interval obtained by considering only the active regions.

Unfortunately, if the transmission rate is kept constant, then large buffers are necessary to maintain low probabilities of overflow. This has a disadvantage of long system delay and high realization costs. However, if the transmission rate can be varied, considerably shorter buffer storage would be sufficient to keep the same probabilities of overflow and underflow. This can be done by using a control mechanism which monitors the state of the buffer and regulating the transmission rate accordingly. If the buffer is excessively full, the transmission rate should be increased. Conversely, the transmission rate should be reduced if the buffer is almost empty.

It is necessary both to prevent underflow and to keep the average transmission rate equal to the average data arrival rate to the buffer to make the best use of the transmission channel.

A general form of the control algorithm is,

$$R(S) = R_o \cdot H(S) \qquad (3.13)$$

where R(S) is the transmission rate when there are S words in the buffer, R_o is the average transmission rate (bits/sec), and H(S) is the coefficient of adaptation rate dependent on the number of word in the transmission buffer.

H(S) must satisfy the following restrictions: first, it must be an increasing function, second, the average buffer

occupancy level about which S fluctuates is half of the buffer size, i.e. H(L/2) =1, where L is the buffer size.

Since it is required that the average transmission rate is R_o,

$$\sum_{i=0}^{L} |Pr(S=i)/R(i)| = \frac{1}{R_o} \tag{3.14}$$

or

$$\sum_{i=0}^{L} \frac{Pr(S=i)}{H(i)} = 1 \tag{3.15}$$

In practice a piecewise linear approximation to H(S) is used. A typical example for the case

$$Pr(S \leq L/10) = 1/10 \quad \text{and} \quad Pr(S \geq 4L/5) = 1/5$$

would be as shown in Figure 3.7 so that (3.14) is satisfied.

3.6 CHANNEL BANDWIDTH

If the run-length codes have a fixed length of l_c bits, the channel capacity required for the system employing the proposed algorithm will be a fixed increase of the average transmission rate from R_o to $(l_c+1)R_o$ bits/sec, where one bit is added for pulse polarity information. In the case of asynchronous source encoding schemes, the necessary channel capacity is determined by the minimum inter-bit-interval

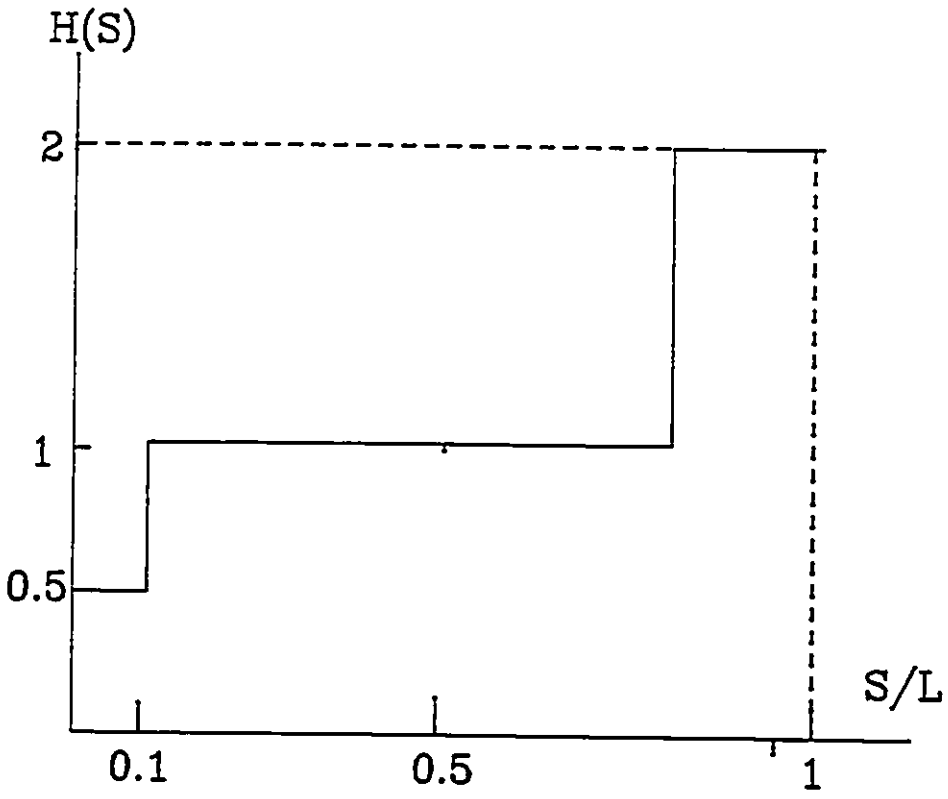


FIGURE 3.7 A Typical Example of H(S)

time of the pulses. Therefore, the necessary channel capacity is $2/\Delta t_{min}$ bits/sec.

In the proposed system, if fixed length codes are used, and only the reduction in the bit transmission rate is considered, the improvement in speed over a synchronous system achieved can be measured by the following ratio:

$$\frac{2/\Delta t_{min}}{l_c/E(\Delta t)} = \frac{E(\Delta t)}{l_c \Delta t_{min}} \tag{3.16}$$

In the case of speech signals where idle period and pauses constitute 50%, the average information provided by the code words is:

$$H(\Delta t) = -\sum_j p(\Delta t_j) \log_2[p(\Delta t_j)] \text{ bits} \tag{3.17}$$

where $j = 1, 2, \ldots, L_c$. L_c is the nearest integer to $\Delta t_{max}/\Delta t_{min}$. In other words, L_c is the longest encodable run-length with l_c bit-long code word, i.e. $L_c = 2^{lc}$.

Since IBI's are subject to time quantization, the asynchronous source encoder output pulses can be assumed to appear at instants which are integer multiples of the unit run-length. Consequently, one can consider the pulses to be nonredundant samples, and the intervals, which are filled with zeros, between the pulses to be the redundant samples. A Markov model can be defined with the following parameters:

P_r = Probability of a redundant sample

P_n = Probability of a nonredundant sample

$P_{n/r}$ = Probability of a transition from a redundant to a nonredundant sample

$P_{r/n}$ = Probability of a transition from a nonredundant to a redundant sample

where $P_{r/n}$ is always equal to 1 in our system because each nonredundant sample is assumed to be followed by at least one redundant sample which implies a run-length of at least 1 in all cases.

Consider the following:

$P_n + P_r = 1$

$P_{n/r} + P_{r/r} = 1$, $P_{n/n} + P_{r/n} = 1$

$P_{r/n} = 1$ and $P_{n/n} = 0$

Let l_r denote the length of a redundant run. The run-length probabilities are expressed as follows:

$$P_r(l_r = j) = P_{r/r}^{j-1} P_{n/r} = P_{r/r}^{j-1} (1-P_{r/r}) \qquad (3.18)$$

then

$$H(\Delta t) = -\sum_i P_{r/r}^{i-1}(1-P_{r/r}) \log \left| P_{r/r}^{i-1}(1-P_{r/r}) \right| \quad (3.19)$$

$$= -\sum_i P_{r/r}^{i-1}(1-P_{r/r})(i-1) \log P_{r/r} - \sum_i P_{r/r}^{i-1}(1-P_{r/r}) \log(1-P_{r/r})$$

$$= -\frac{P_{r/r}}{(1-P_{r/r})} \log(P_{r/r}) - \log(1-P_{r/r}) \quad (3.20)$$

The run-length has a geometric distribution [21] and has a mean of

$$l_{av} = \frac{1}{(1- P_{r/r})} \quad (3.21)$$

By substituting (3.21) into (3.20), we obtain

$$H(\Delta t) = l_{av}\frac{(l_{av}-1)}{l_{av}} \log(\frac{l_{av}-1}{l_{av}}) - \log \frac{1}{l_{av}}$$

$$= l_{av} \log l_{av} - (l_{av}-1) \log(l_{av}-1) \quad (3.22)$$

This is the same result as obtained by Huang [22], it gives an upper bound on the entropy per run, where each run-length was discretized independent exponentially distributed random variables.

P_n can be represented as:

$$P_n = P_r P_{n/r} = (1-P_n)(1-P_{r/r})$$

$$P_n = \frac{1-P_{r/r}}{2-P_{r/r}} \qquad \Rightarrow \qquad P_{r/r} = \frac{2P_n-1}{P_n-1} \tag{3.23}$$

Therefore, by substituting (3.23) into (3.21), we obtain

$$l_{av} = \frac{1}{P_n} - 1 \tag{3.24}$$

The system has only one independent parameter, $P_{r/r}$. When $P_{r/r} = 1/2$, then $l_{av} = 2$ and $H(\Delta t) = 2$. If $P_{r/r}$ increases above 1/2, the entropy also increases but not as fast as l_{av}, and if $P_{r/r}$ decreases below 1/2, entropy also decreases but faster than l_{av}. $1 - H(\Delta t)/l_{av}$ is plotted in Figure 3.8 as a function of l_{av}.

The average rate of information is $H(\Delta t)/\Delta t_{av}$ where as the transmission rate is $l_c/\Delta t_{av}$ if fixed-length coding is used. Therefore if $H(\Delta t) < l_c$, less information than that which is transmitted could be sufficient to resolve the uncertainty. $H(\Delta t) = l_c$ only when the inter-bit interval time has a uniform distribution. However, the examination of $p(\Delta t)$ for various cases shows that long and short IBI's are quite less probable than those of medium length.

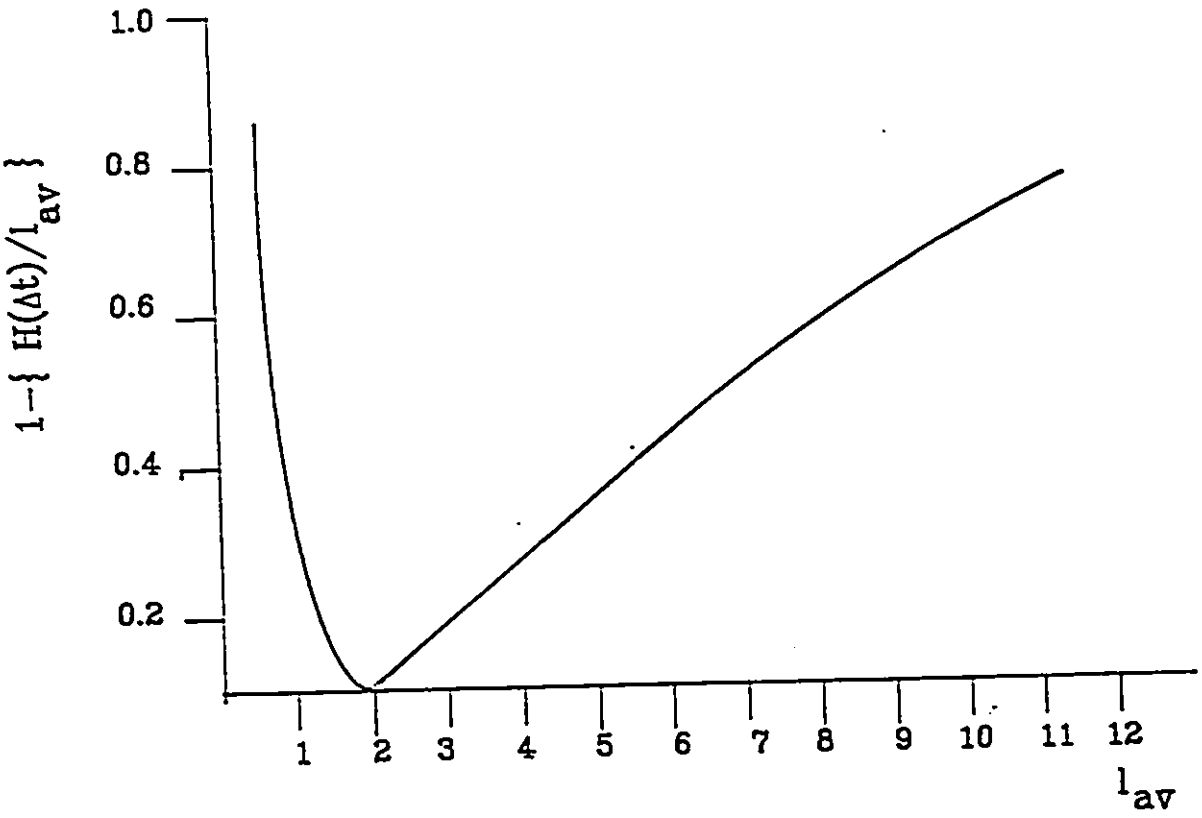


Figure 3.8 The Upper Bound to the Entropy of a Run Length Code

This suggests that variable length coding should be used, in which case our primarily interest will be in the average code length, n_{av}.

$$n_{av} = \sum_j p(\Delta t_j)\ l_j \tag{3.25}$$

where Δt_j is the j'th sample interval and l_j is the number of bits of the code word associated to that interval.

Because of certain restrictions on the variable length codes, in order to guarantee unique decodability, most commonly prefix codes are used. It is possible to assign prefix codes so that [22],

$$n_{av} < H(\Delta t) + 1 \tag{3.26}$$

where $H(\Delta t)$ is the entropy of the sample interval, Δt.

By assigning code words to sequences of $K\Delta t$'s , where K is > 1, rather than to each Δt, then we can obtain a better average code word length.

$$n_{av} < H(\Delta t) + 1/K \tag{3.27}$$

In order to find the optimum variable length code, the Huffman coding procedure can be applied. First, all $p(\Delta t_j)$ should be found, then it is a very systematic procedure to obtain the Huffman codes [23].

If we expected that $n_{av} < l_c$, that is $rn_{av} = l_c$, with r being the compression ratio achieved by using Huffman codes instead of fixed length codes, then the necessary channel capacity will be

$$\frac{\log_2(\Delta t_{max}/\Delta t_{min})}{r\ \Delta t_{av}} \qquad (3.28)$$

One may try reduce further the necessary channel capacity by decreasing the number code words. This is possible either by increasing Δt_{min} or decreasing Δt_{max}. However, one must note that increasing Δt_{min} introduces more time quantization error, whereas decreasing Δt_{max} means that more IBI's will exceed Δt_{max} and will be encoded incorrectly.

3.7 NOISE CHARACTERISTICS

In all digital modulation methods, amplitude quantization is one of the major sources of quantization error. In the case where run-length encoding is employed, an additional noise component due to time quantization is added to the total distortion. A list of the distortion sources for the system encoder investigation is shown below:

(1) Amplitude quantization

(2) Time quantization

(3) Run-length limit

(4) Buffer overflow

Expressions for both the amplitude and time quantization errors are derived in this section, while the nature of the distortions due to the run-length constraint and the buffer overflow are to be discussed in chapter 4.

In an asynchronous source encoder, the intervals between samples, IBI, are measured in multiples of the unit time quantum, τ. Different run-length codes are then assigned to different IBI's. If l_i is the run-length corresponding to Δt_i and is allowed to take on integer values only, then it is always true that

$$(l_i - 1)\tau < \Delta t_i \leq l_i\tau \qquad (3.29a)$$

Alternatively,

$$l_i\tau = \Delta t_i + \tau - \Delta\tau, \qquad 0 \leq \Delta\tau < \tau \qquad (3.29b)$$

where $\Delta\tau$ is the resulting time displacement of the reconstructed signal in the receiver. In order to prevent the accumulation of these time displacements on the receiver side, a negative bias of $\tau/2$ can be added to the run-length decoder.

The effects of the time quantization for an ASDM is shown in Figure 3.9.

The mean squared error for the j'th level can be expressed as,

$$E(e_j^2) = \int_{x_j - a/2 + \overline{\Delta a}}^{x_j + a/2 + \overline{\Delta a}} (x - x_j)^2 p(x)\, dx, \text{ for } \frac{dx}{dt} > 0 \qquad (3.30a)$$

and

$$E(e_j^2) = \int_{x_j - a/2 - \overline{\Delta a}}^{x_j + a/2 - \overline{\Delta a}} (x - x_j)^2 p(x)\, dx, \text{ for } \frac{dx}{dt} < 0 \qquad (3.30b)$$

where $\overline{\Delta a}$ is the average absolute amplitude displacement. For $a << 1$, one can assume that $p(x)$ is constant for the interval $[(j - 1/2)a, (j + 1/2)a]$.

Let $p(x) = p(x_j)$ for $(j - 1/2)a < x \leq (j + 1/2)a$

and $\Pr[(j - 1/2)a < x \leq (j + 1/2)a] = p_j$

then, $p_j = p(x_j)a$; therefore

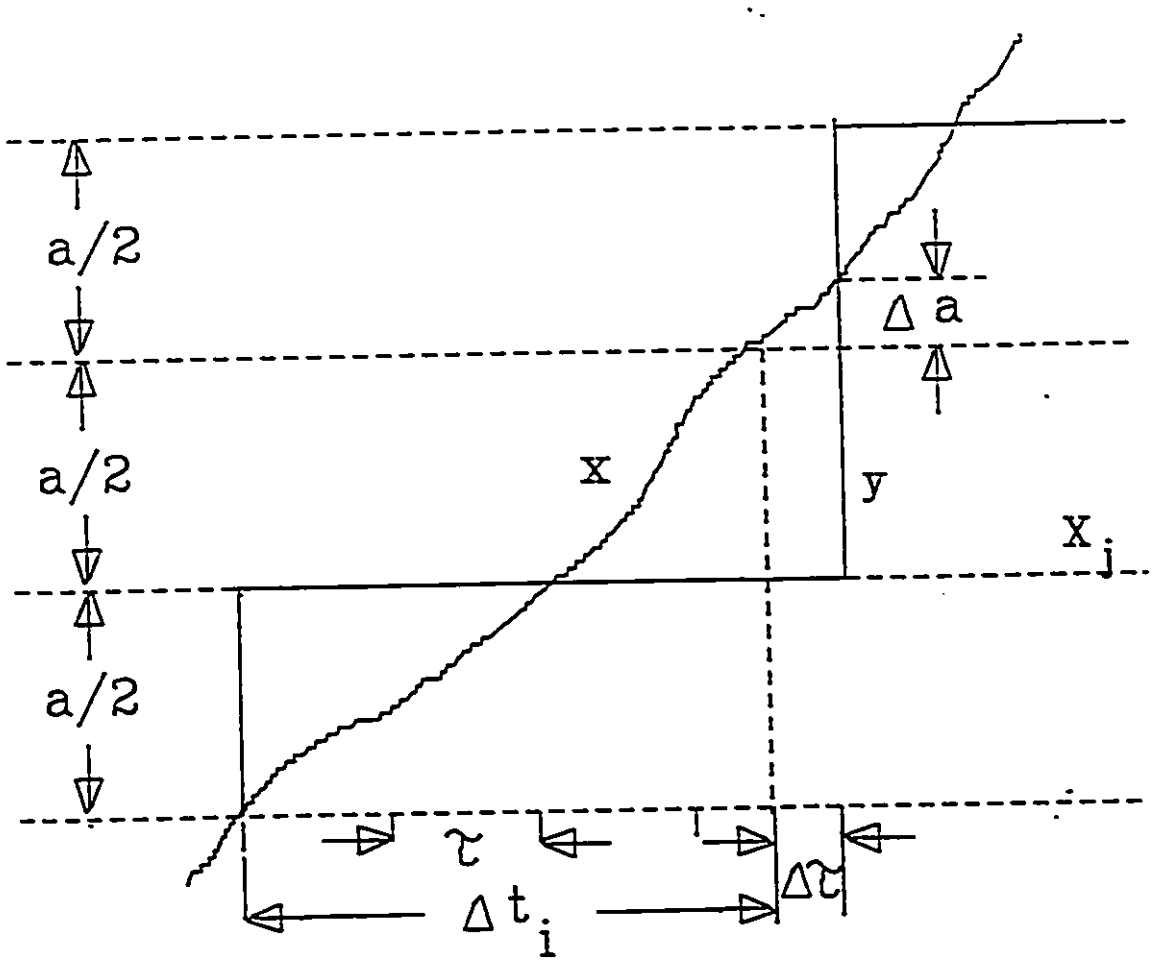


Figure 3.9 The Effect of Time Interval Quantization

$$E(e_j^2) = p(x_j) \int_{x_j - a/2 + \overline{\Delta a}}^{x_j + a/2 + \overline{\Delta a}} (x - x_j)^2 \, dx, \quad \text{for } \frac{dx}{dt} > 0 \tag{3.31}$$

By substituting $n = x - x_j$

$$E(e_j^2) = p(x_j) \int_{-a/2 + \overline{\Delta a}}^{a/2 + \overline{\Delta a}} \eta^2 \, d\eta = \frac{p(x_j)}{3} \left(\frac{a^3}{4} + 3a \, \overline{\Delta a}^2 \right) \tag{3.32}$$

The same result is obtained if $dx/dt < 0$.

The total mean squared error can be found by summing all the error terms,

$$E(e_j^2) = \sum_j e_j^2 = \sum_j \frac{p(x_j) a^3}{12} + \sum_j a \cdot p(x_j) \, \overline{\Delta a}^2$$

since

$$\sum_j p_j = \sum_j a \, p(x_j) = 1$$

we obtain,

$$E(e_q^2) = \frac{a^2}{12} + \overline{\Delta a}^2 \tag{3.33}$$

The first term of (3.33) is the familiar amplitude quantization error, while the second term represents the contribution of the time quantizing or displacement error.

Recall from Figure 3.9, that if τ is chosen such that $\tau < \Delta t_{min}$, one can write,

$$|x'(t)|_{t=t_i} = \frac{a_i}{\Delta\tau_i}$$

or

$$\Delta a_i = |x'(t)|_{t=t_i} \Delta\tau_i \qquad (3.34)$$

Assuming Δt is independent of $|x'(t)|$ and uniformly distributed between 0 and t. Let $x'(t)$ be μ, then

$$\overline{\Delta a} = E\{\Delta t\} \; E\{|x'(t)|\} \qquad (3.35a)$$

$$= \frac{\tau}{2} \int_0^{|\mu|_{max}} |\mu| \; p_{|\mu|}(|\mu|) \; d(|\mu|) \qquad (3.35b)$$

if $p_\mu(\mu) = p_\mu(-\mu)$, then

$$\overline{\Delta a} = \tau \int_0^{\mu_{max}} \mu p_\mu(\mu) \; d\mu \qquad (3.36)$$

For a bandlimited signal, if f_{max} is the highest frequency component, then

$$|x'(t)|_{max} = 2 X_{max} 2\pi f_{max}$$

where X_{max} is the highest amplitude value. Let m be the number of steps used for the approximation

$$m \triangleq \frac{2X_{max}}{a} \qquad \text{or} \qquad |x'(t)|_{max} = f_{max} \pi ma.$$

By using (3.36) and the above, (3.33) becomes

$$E(e_q^2) = \frac{a^2}{12} + \left| \tau \int_0^{f_{max} \pi ma} up_u(u)\, du \right|^2 \qquad (3.37)$$

For a Gaussian input with variance σ^2 and bandlimited to f_{max},

$$E\{ |x'(t)| \} = \sqrt{\frac{8\pi}{3}}\, \sigma f_{max} \qquad (3.38)$$

Assuming that the maximum value of the slope, $|x'(t)|_{max}$, is N times its rms value, and $\tau = \Delta t_{min}$, then (recall (2.7) and (3.34))

$$\tau = \Delta t_{min} = \frac{a}{|x'(t)|_{max}} = \frac{\sqrt{3}\ a}{2N\pi f_{max}\sigma} \qquad (3.39)$$

Combining (3.35a), (3.35b), (3.38) and (3.39), (3.33) becomes

$$E(e_q^2) = \frac{a^2}{12} + \left| \frac{\sqrt{3}\ a}{4N\pi f_{max}\sigma} \sqrt{\frac{8\pi}{3}}\, \sigma f_{max} \right|^2$$

$$E(e_q^2) = \frac{a^2}{12} + \frac{a^2}{2N^2\tau} \qquad (3.40)$$

This result, which is valid for small value of a, indicates that the total time quantization noise power is also a function of step size and is less than that of the total amplitude quantization noise power for a bandlimited Gaussian input, if $N > \sqrt{6}/\tau = 1.38$.

As seen in Figure 3.10, the maximum amplitude quantization error is a/2, while the maximum total error is $a/2 + \Delta a_{max}$. The time quantization error signal consists of positive and negative pulses of height (a) and (-a) with uniformly distributed random width between 0 and τ. The time difference between leading edges of the pulses are integer multiples of τ, namely the corresponding run-length.

In this chapter, the characteristics, such as the quantization noise, the IBI characteristics, the transmission rate and the channel bandwidth, on the corridor asynchronous delta modulation system have been described. In chapter 4, the application of run-length encoding on the proposed asynchronous delta modulator output is described. The non-uniform sampling intervals and the polarities of the steps are run-length encoded.

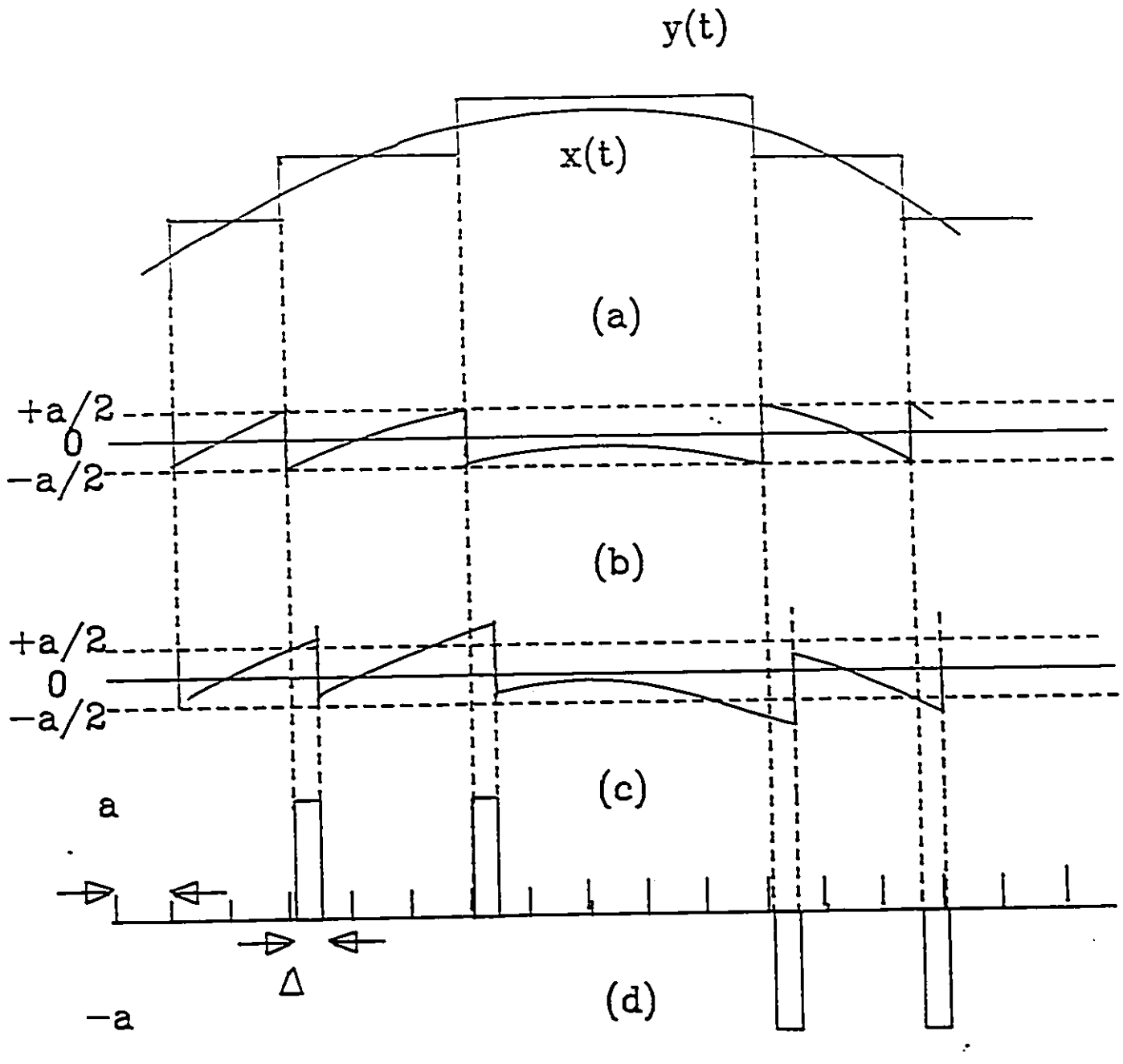


Figure 3.10 Error Waveforms

(a) Asynchronous Delta Modulated Signal

(b) The Amplitude Quantization Error

(c) The error signal resulting from both amplitude and time quantization

(d) The error signal due to time quantization only

CHAPTER IV

4.0 RUN-LENGTH CODING

4.1 INTRODUCTION

In this chapter, a brief overview of the time codes is given. It is followed by the description of the application of run-length encoding to asynchronously sampled speech like signals. Coding techniques which prevent run-length overflow are also presented. Conditions of buffer overflows and underflows are also studied.

4.2 GENERAL TIME CODES

Before discussing run-length coding, we briefly discuss the properties of general time codes.

If there exists a certain predictability among the sampling sequences constituting the message then the samples can be classified into two categories, namely, redundant and non-redundant samples. Part of this redundancy can be eliminated by the use of predictive transformations aiming at reducing the transmission rate. In some cases, such as run-length encoding, the parameters derived from this transformation together with time codes which carry the

timing information of the sample sequences are sent along with non-redundant samples to achieve data compression.

The time codes can be arranged in a way that the timing information is sent separately from the nonredundant sample values. Such codes are known as total information codes. Whereas, if the nonredundant sample values are required to be sent in conjunction with the time codes, then these codes are known as partial information codes.

Basically there are three different coding techniques for the sample sequences which have been determined to be redundant and nonredundant samples by the predictive scheme selected:

(a) Total Information Codes:

Time Sequence codes that encode a certain fixed number of samples in such a way that it is uniquely decodable. The codes are of fixed length and the redundant samples can easily be distinguished from nonredundant samples by assigning different code words.

(b) Partial Information Codes:

Either the sampling time of each nonredundant sample or the beginning of a nonredundant run, such as a sampling pulse in the ASDM, is transmitted within a fixed time frame. The number of nonredundant samples have to be counted at the

receiving end in order to specify the length of the redundant run.

(c) Run-length Codes:

In this case, it is possible to encode either both the redundant and nonredundant runs or only the redundant runs, i.e. redundant samples together with the sampling intervals. For a binary source, a predictive transformation can be placed before the run-length coding for run-length generation. In Figure 4.1, successive N-bit sequences are compared, i.e. added.

It is highly probable that if repeated N-bit sequences follow each other, then long strings of zeros, redundant samples, will appear at the output of the modulo 2 adder. The sequence can be recovered eventually when a one appears at the output of the adder given that the same logic is provided at the receiving end. Hence large compression ratios can be achieved by run-length encoding the redundant sample sequences.

If the run-length code has fixed maximum length, i.e. n bits, an upper limit of 2^n code words is imposed on the encoded run-length.

The three coding techniques described above assume that nonredundant samples may appear consecutively. If it is not probable that consecutive samples are nonredundant, then

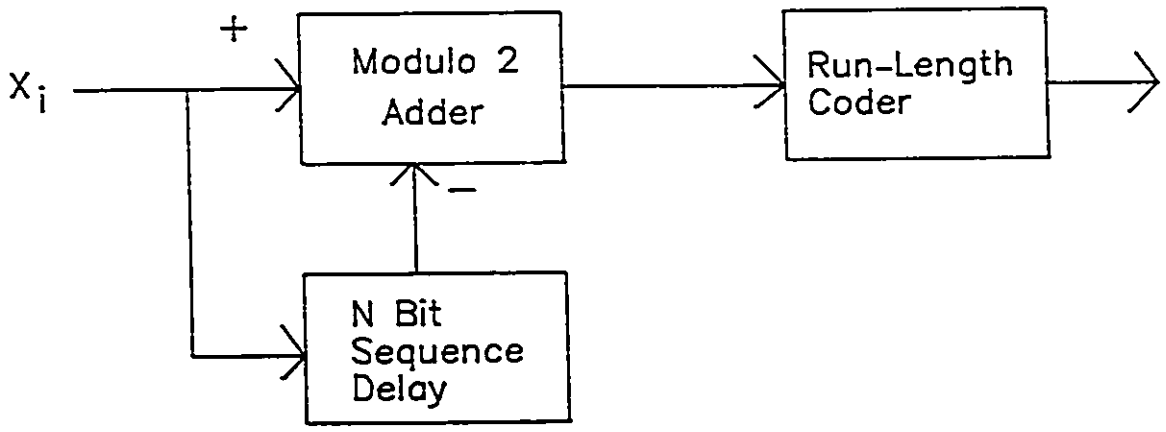


Figure 4.1 Predictive Transformation for Run-Length Encoding

each redundant run can be coded after being added to 1 to indicate the end of the sequence. Thus, if occasionally a nonredundant sample is followed by another one, then the corresponding redundant run-length will be encoded as 1. This procedure will result in an efficient total information coding, since all run-lengths are uniquely decodable and the redundant sample sequences, which contain strings of zeros, can easily be distinguished from the nonredundant ones.

The advantage of run-length coding on the ASDM sampling intervals is apparent due to the saving from the redundant samples through the ASDM process and the simplicity of the coding techniques that can be employed.

In the next section, we describe the application of run-length coding on the proposed ASDM.

4.3 RUN-LENGTH CODING FOR ASYNCHRONOUS SAMPLING SYSTEMS

If the inter-bit-interval times are continuous random variables, choosing the unit time quantum for the run-length coding becomes a matter of compromising between the noise introduced by the time quantization process and the given channel bandwidth. However, in (3.36), time quantization noise power is given by:

$$E(e^2_{tq}) = [\tau \int_0^{\mu_{max}} \mu p_\mu(\mu)\, d\mu]^2$$

where τ is the unit time quantum, μ is the slope of the processed signal. It is shown in (3.40) that the time quantization noise is negligibly small with respect to the amplitude quantization noise. Therefore it may be advantageous to choose the unit time interval in run-length, τ, larger than the minimum inter-bit-interval (IBI), Δt_{min}. In order to make a rational decision on the choice of τ, one would need to analyze the time quantizing noise spectrum and find the portion that lies within the signal bandwidth, which directly affects the signal to noise ratio, SNR.

In this section, we concentrate on the determination of the maximum allowable run-length for an assumed value of the unit time quantum. The curves in Figure 4.2 show the probabilities of the erroneous encoding, p(e), for different values of τ due to the occurrence of time intervals longer than L_c, where L_c is the fixed maximum run-length $N \cdot \tau$, and N is an integer. (Note that, in this case, the p(e) is associated to the error of an upward or downward step in DM, due to the run-length encoding the maximum allowed, and not to the encoded signal amplitude as often defined in PCM). For example, if 5 bit (1 sign bit and 4 run-length bits) coding is employed, we assume that all IBI's longer than $2^5\tau$ seconds will be encoded erroneously since linear coding is

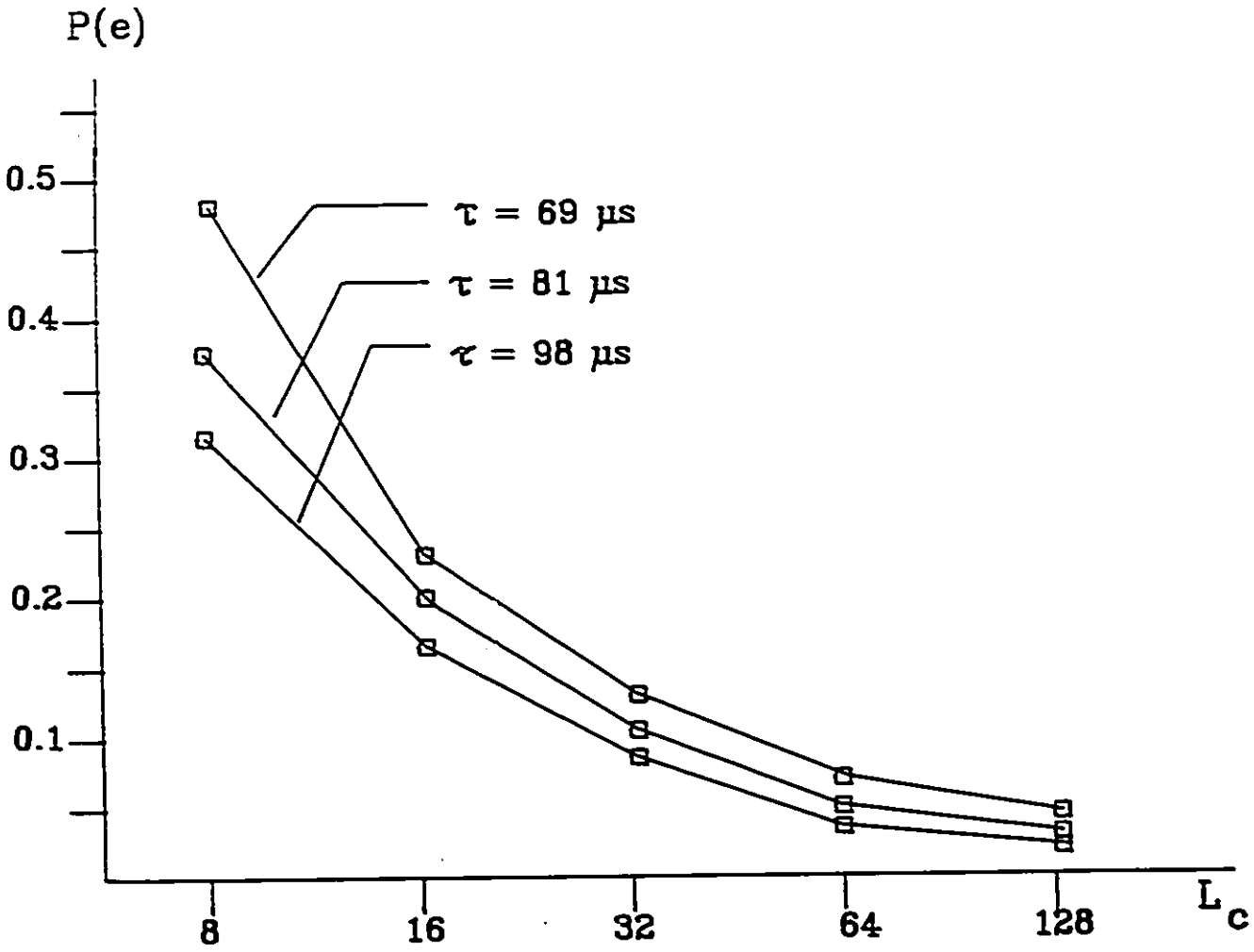


Figure 4.2 P(e) vs L_c for different value of unit time quantum τ

assumed. The curves in Figure 4.2 are obtained with a Gaussian signal whose inter-arrival-time pdf is as defined in (3.7):

$$p(\Delta t) = \frac{2a}{(\Delta t)^2} \frac{1}{2\,\sigma^2} \exp\{ - \frac{a^2}{2(\Delta t)^2 \sigma^2}\}$$

As shown in Figure 4.2, as the coder capacity increases, the p(e) decreases, at the cost of longer code words. Lower p(e) is obtained with longer τ due to the fact that the encoder is capable of handling longer IBI's, however, at the same time contributing a higher time-quantizing noise.

The measure of the encoding error depends on how the run-lengths that are longer than L_c are being handled. If the counters of the run-length coder are forced to stop counting at the instant when the run-length exceeds L_c, it is suspected that only the code words corresponding to L_c may be erroneous since a sign bit is always included in the run-length code. Thus an upward or downward step is interpreted. In this case, code words corresponding to L_c can be decoded as the expected value of the run-length given that it is greater than or equal to L_c. This precaution is expected to reduce the timing errors. This is described in more detail in section 4.5.

The timing errors due to confining the run-length cause the signal to be displaced in time since the time interval between two level crossings is effectively shortened or lengthened. To compensate for this distortion, synchronization words can be inserted and transmitted between blocks of code words [22]. The synchronization word bears knowledge about the cumulative difference between the actual run-lengths and the run-lengths corresponding to the code words. Thus a correction is made in the demodulator by shifting the signal by the appropriate amount. With this scheme some degree of freedom is also achieved in decreasing the maximum run-length by decreasing the synchronization interval. In Figure 4.3 the signals before encoding and after decoding in the presence and absence of synchronization are shown, in case timing errors occur because of excessively long IBI's. In Figure 4.3 (a), the solid line shows the original signal before encoding, whereas the dashed line is the decoded signal where the shift results from encoding errors within the interval $|t_1,t_2|$. Figure 4.3(b) shows the decoded signal if synchronization words are used. t'_1 is the synchronization time and $t'_2-t'_1$ is the shift produced by the synchronization word.

The application of a sychronization word has been simulated and a slight improvement is obtained and presented in chapter 6.

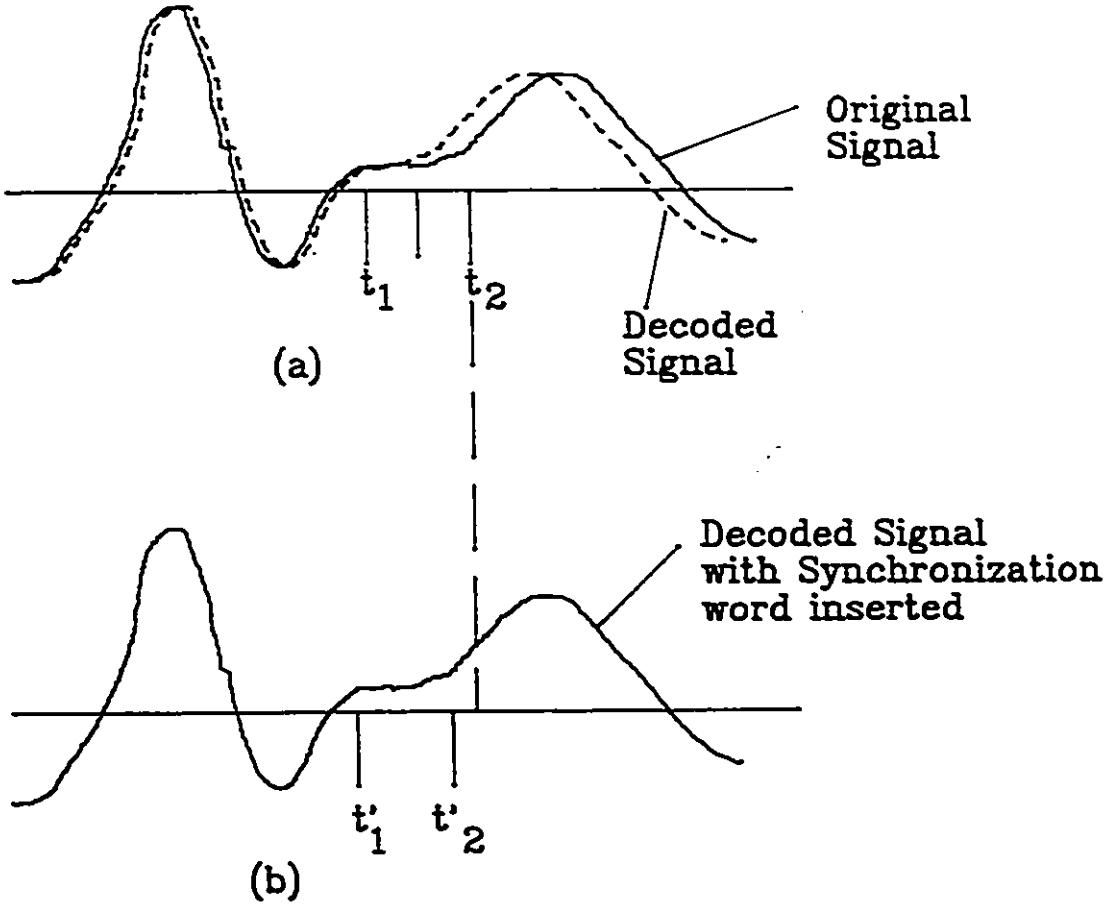


Figure 4.3 The Effect of Applying Synchronization Word

4.4 Differential Run-length Coding

Although most of the redundancy in a signal can be removed by zero-order prediction, the time intervals between the corridor crossings of an ASDM can still be correlated. Since the time intervals between the corridor crossings are directly related to the slope signal, Δt remains more or less the same if the slope of the signal does not change very rapidly. This suggests that a further reduction in the transmission rate can be achieved if differential run-length encoding is employed.

In the simulation presented in chapter 6, certain run-lengths are assumed to be transmitted for synchronization in certain intervals. Upon the reception of the sychronization word, the decoder recovers the correct run-length value. This prevents the random time displacements in the reconstructed signal. Since both forward and backward time shifts are likely to occur, the building up of time displacements in a single direction is not considered.

The simulation results in chapter 6 show that no improvement is achieved by the use of differential run-length codes.

4.5. PREVENTION OF RUN-LENGTH OVERFLOW

Three different coding techniques for run-length coding are described below for preventing data loss due to excessively long IBI's.

(1) If nonredundant sample values are also transmitted, together with redundant samples the sampled data stream can be coded in frames of length M symbols, where M is less than or equal to 2^{l_c}, where l_c is the code word length in bits. The first symbol of each frame is always considered to be nonredundant.

(2) Only when a run exceeds 2^{l_c}, then the sample at position 2^{l_c} will be nonredundant.

(3) In this scheme, rather than regarding redundant samples values as nonredundant, the run-lengths are encoded with a variable number of code words. This method is called run-length coding with standard run-lengths. With this scheme the run-lengths can be encoded effectively, no matter how long they are.

Bradley [23] and Molinder [24] have discussed such techniques, which are worth restating. The codes generated by these techniques are considered to be optimal for run-length encoding, if certain parameters are properly set.

Consider a binary source where each redundant sample is represented by a "0" and each nonredundant sample is represented by a "1". For runs of 0 or 1, one code word from a set of K code words is assigned, where each code word represents a unique but fixed-length run-length. The optimal choice of K will be the one that provides the maximum compression ratio.

Certain output sequences are encoded with code words C_i, for example Huffman, as shown below:

$$
\begin{aligned}
&01 \longrightarrow C_{1,0}\\
&001 \longrightarrow C_{2,0}\\
&\vdots\\
&\underbrace{00\ldots01}_{i} \longrightarrow C_{i,0}\\
&\underbrace{000\ \ 000}_{K} \longrightarrow C_{K,0}\\
&\underbrace{1\ldots110}_{i} \longrightarrow C_{i,1} \qquad 1 \le i \le K-1\\
&\underbrace{1\ldots111}_{K} \longrightarrow C_{K,1}
\end{aligned}
$$

A 2K standard run-length is chosen in the above and the code symbols C_i can be of variable length.

Now consider the asynchronous source encoder output sequence where sample occurrences are represented by 1's and the intervals are filled with 0's. The above procedure can be modified as shown below:

$$
\begin{array}{lll}
1 & \longrightarrow & C_1 \\
01 & \longrightarrow & C_2 \\
\vdots & & \\
\underbrace{0\ldots01}_{K-1} & \longrightarrow & C_K, \quad K \text{ zeros} \rightarrow C_{K+1} \\
 & & \quad 2K \text{ zeros} \rightarrow C_{K+2} \\
 & & \quad \vdots \\
 & & \quad NK \text{ zeros} \rightarrow C_{K+N}
\end{array}
$$

Let (K+N) be chosen such that $\log_2(K+N) = n$, where n is an integer. Each code word have fixed length $\log_2(K+N) = n$. Run-lengths less than K are coded with a single code word whereas longer ones are made up of more than one code word. As an example, if K = 5 and N = 3, then a run-length of 25 will be encoded as

$$C_8 + C_6 + C_5 = 15 + 5 + 5$$

By using one set of code words, the maximum length that can be encoded is K. With a combination of two sets code words, it becomes NK+K. Let W_i be the maximum encodable run-length with i code words, then

$$W_i = K + (i-1)KN \tag{4.2}$$

If i code words are necessary to encode a set of run-lengths then the average number of bits required is B = (ni), where $n=\log_2(K+N)$. If P_j is the probability that the run-length is equal to j, then the average number of bits required per run-length is

$$B_{av} = n \sum_{j=1}^{W_1} P_j + 2n \sum_{j=W_1+1}^{W_2} P_j + \ldots$$

$$= \sum_{i=1} ni \sum_{j=W_{i-1}+1}^{W_i} P_j, \qquad \text{where } W_o = 0 \qquad (4.3)$$

This code can be optimized by finding the values of N and K for which B_{av} is minimized.

4.6 BUFFER OVERFLOW AND UNDERFLOW

If data flow into the buffer storage is at a higher rate than the data readout rate for a long enough period, then the buffer would eventually overflow. Since during an overflow period no data can be allowed into the buffer, the data is lost, causing a signal distortion and a loss of synchronism.

If, on the contrary, the data read-in rate is lower than data read-out rate, then the buffer would underflow. In other words, attempts are made to remove data from the buffer even when the buffer is empty. In many ways, buffer underflow is not as detrimental as overflow. It is only of concern in terms of redundancy. During underflow periods the channel is occupied dispute there being no data in the buffer.

For smaller buffer sizes, overflow and underflow both occur more frequently. In most applications buffer overflow is the most important design consideration. The necessary buffer size is determined for a given overflow probability, which is expected to distort the specific input signal by no more than an acceptable amount. The buffer length is limited mainly by the delay introduced and the implementation cost.

Overflow and underflow probabilities are affected severely by the transmission rate. A slight increase of the transmission rate reduces the overflow probability appreciably and this is usually recommended to prevent data losses, although it brings along the disadvantage of frequent underflow.

An increase in buffer size leads to the reduction of both overflow and underflow. Altering the transmission rate increases one, while decreasing the other. The data transmission rate is determined for a given buffer size only by the relative importance of overflow and underflow.

An infinite buffer model is considered for the derivation of analytical expressions of overflow probability. The overflow probability is defined as the probability that the buffer exceeds a prescribed threshold, L. In fact this probability is an upper bound to the overflow probability that one would obtain by considering a finite buffer.

For a finite buffer the probability of overflow is

$$P_I(\text{overflow}) = \sum_{S \geq L} P_I(S) = 1 - \sum_{S=0}^{L-1} P_I(S) \qquad (4.4)$$

where $P_I(S)$ is the probability that S words are present in the buffer queue.

Considering a finite buffer, assuming the state probabilities $P_I(S)$, S=0, 1,...,L-1, are expressed recursively and are approximated for a finite buffer with the normalization condition

$$\sum_{S=0}^{L-1} P_F(S) = 1 \qquad (4.5)$$

where the subscript F indicates the finite buffer, such that the work presented hereafter can be applied.

The overflow occurs when the sum of the data in the queue during the last transmission and the new word (z) that arrives in the buffer before the transmission time, exceeds the buffer length, L. Therefore the conditional buffer overflow probability for a given buffer length S can be expressed as:

$$\Pr(\text{overflow} \mid S) = \Pr [z > (L-S)]$$

and the probability of overflow is

$$\Pr(\text{overflow}) = \sum_{i=0}^{L-1} \Pr(i)\Pr[z > (L-S)] \qquad (4.6)$$

This probability alone is not a good measure of the subjective quality of the recovered message, especially in the case of asynchronous waveform encoding. A characteristic of overflow is a dc shift in the reconstructed signal voltage if ASDM pulse intervals are encoded and buffered. The amplitude of this dc shift is directly proportional to the number of the sampling pulses arrive during an overflow period. Although, the probability of overflow may be the same for two different signals, the one which has more samples missed due to the overflow will be affected more. It is obvious that the distortion caused by overflow is closely related to the statistical properties of the IBI of the processed signal. The maximum possible dc shift in one overflow period can be found if the maximum slope of the band limited signal is known. Figure 4.4 shows the signal distortion due to overflow for ASDM. Δt_1 is the only IBI lost due to overflow, therefore the reconstructed signal is shifted by 'a' volts, i.e. one upward step is missed.

Considering that the expected number of words lost in a transmission period due to overflow as a distortion measure, we will express this quantity for the general case. Let P(i) be the probability that the buffer overflows and i words

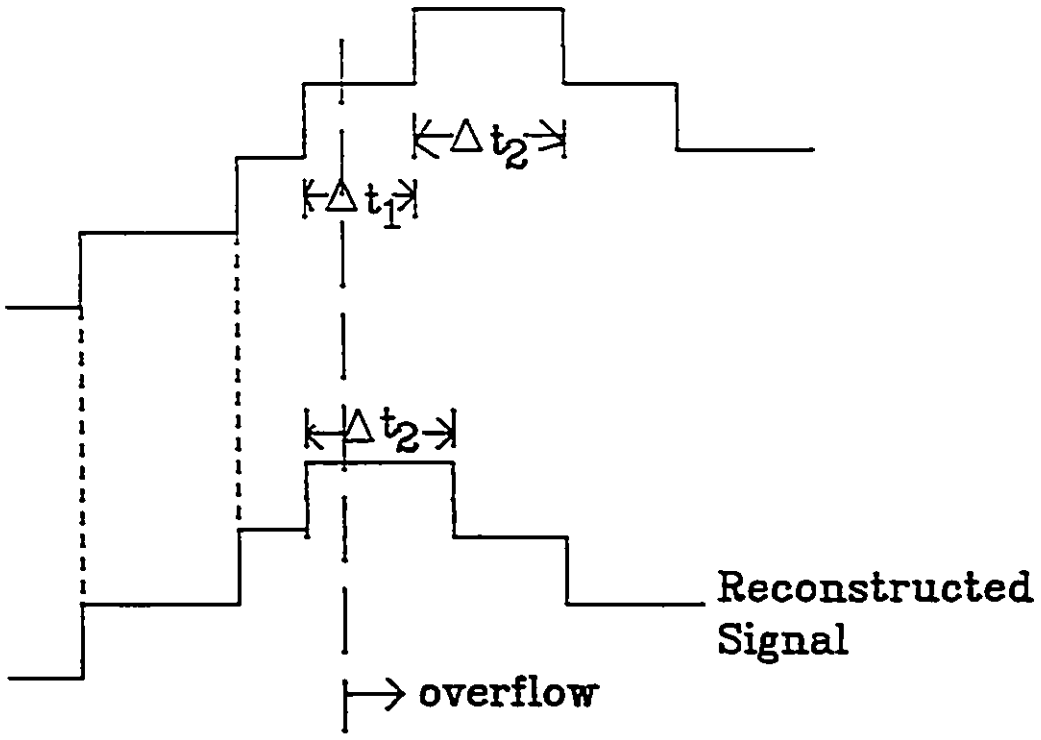


Figure 4.4 Effect of Buffer Overflow

are lost in a transmission period. Assuming that Pr(z>L) = 0,

$$P_o(1) = P(L-1)Pr(z=2) + \ldots + P(1)Pr(z=L)$$
$$P_o(2) = P(L-1)Pr(z=3) + \ldots + P(2)Pr(z=L)$$

or in short notation,

$$P_o(i) = \sum_{j=1}^{L-1} P(j)\ Pr(z = L+i-j) \qquad (4.7)$$

Thus the expected number of words lost in a transmission period is:

$$E\{k\} = \sum_{k=1}^{L-1} k \sum_{j=k}^{L-1} P(j)Pr(z= L+i-j) \qquad (4.8)$$

The fractional word loss R_L is

$$R_L = E\{k\} \ / \ E\{z\} \qquad (4.9)$$

where E{k} is equal to the fractional word loss when E{z}=1.

Now we wish to express R_L in terms of underflow probability, P_u. P_u can be alternatively defined as the probability of not removing a word in a transmission period. Assume that N samples are attempted to be stored in the buffer, and the transmission of all data that could be stored lasted MT seconds, where T is the transmission period. Thus there are M attempts to remove N sample data.

In the limit, as N goes to infinity, N/M is the mean data arrival rate E{z}. Further, assume that m attempts failed to remove data. Then m/M is the probability of not removing data in a transmission period, or P_u. On the average, $(1-P_u)$ words leave the buffer. Only (M-m) words are read out, and consequently N-(M-m) words are lost due to overflow.

Therefore, the fractional word loss R_L is:

$$R_L = \frac{N - (M-m)}{M} = 1 - \frac{M}{N}(1 - \frac{m}{M})$$

Thus

$$R_L = 1 - \frac{1 - P_U}{E\{z\}} \qquad (4.10)$$

P_U can also be expressed as

$$P_U = P(0) \cdot Pr(z=0) \qquad (4.11)$$

The results show that for given buffer size L and P(z), P_U and R_L can be determined independently, but they are related to each other as a function of E{z} only.

If E{z} = 1, $R_L = P_U$ for any for any buffer size. For higher data flow rates P_U is lower and R_L is higher, and vice versa for lower data flow rates.

4.7 OVERFLOW AND UNDERFLOW CONTROL

Rather than designing the buffer for a given probability of overflow and underflow, it may be more desirable to regulate information input to the buffer. One 88way of regulation is to introduce errors by adding bits in a controlled manner according to a practical criterion. The information rate to the buffer is monitored. If a condition of overflow is predicted, the least significant bit(s) of the run-length code words may be removed at a cost of distortion to the signal. In this case, the code words assigned to the run-lengths will have to be generated in a fashion that removal of the significant bit(s) will not introduce excessive distortion.

More effective procedures can be found, depending on the coding scheme and signal statistics. The method of overflow control investigated in the literature [25][26][27], concerning image and speech data compressors generally consisted of monitoring the queue length of code words within the buffer, and reducing the data rate into the buffer when necessary by relaxing the accuracy requirements on the compressed data. These algorithms have proven to be more effective than increasing the buffer size. Restricting overflow by merely increasing the buffer size is relatively inefficient.

When desired, the rate of data flow can be reduced by one of the following methods:

(1) Filtering:

If the signal is uniformly sampled and the run-length information can be recovered, moving average filtering can be employed. For asynchronous sampling, analog filters can be used. As the queue length increases, the filter parameters are made to vary in such a way that the equivalent low-pass filter cut-off frequency decreases. For example, the number of samples averaged together may be increased, in which case longer run-lengths are more likely.

(2) Sample Deletion:

One out of K uniform samples is retained and others are deleted. The quantity K is made to vary in such a manner that the resulting sampling rate, f_S/K (f_S is the original sampling rate), varies inversely with the queue length.

(3) Adaptive Tolerance:

The adaptive parameter is the aperture for synchronous data compressors, the step size for the ASDM. The aperture or step size is recomputed according to buffer fill immediately after the oldest word has been removed from the buffer.

The control method can be improved by examining other features, such as the most recent sample sent to the buffer or the time integral of the queue length. The disadvantage of this control method is that the compression processing error or signal quantization error increases during periods of high data activity. However, this method apparently gives better results for most applications, and so far it has received the most interest among the competing algorithms.

For multiplexed data compression systems, channel priority can be assigned for data that are too important to risk a loss to buffer overflow. In such a case, when the buffer overflow becomes imminent, the low priority data are more heavily compressed to allow channel usage for high priority data.

CHAPTER V

5.0 VECTOR QUANTIZATION

5.1 INTRODUCTION

Vector quantization (VQ) has recently emerged as a powerful and widely applicable coding technique. It was first applied to analysis and synthesis of speech, and has allowed Linear Predictive Coding (LPC) rates to be dramatically reduced to as low as 1500 b/s with a very slight degradation in quality and further compressed to rates as low as 800 b/s while retaining intelligibility [28] [29]. More recently, the technique has found its way to encoding the speech waveform and speech-like signals [30][31].

Because of its potential and versatility, we propose to use vector quantization as a building block in our asynchronous system. Vector quantization is applied to the CASDM, Corridor Asynchronous Delta Modulator, for a performance comparison with run length codes.

Vector quantization is a technique for mapping a sequence of continuous or discrete vectors into a digital sequence suitable for communication over, or storage in, a digital channel.

The encoding scheme is to select a reproduction vector (codeword) from a finite set of reproduction vectors (code book) for an input vector, which can be a vector of binary quantized samples. The selected codeword index, which is in binary format, is then transmitted over the channel. At the receiving end, the codeword is decoded by a simple "lookup table" for the correct reproduction vector. In such a way, instead of transmitting the signal sample itself, a codeword which represents a sequence of samples is sent over the digital channel. Thus data compression can be achieved.

5.2 VECTOR QUANTIZER DESIGN

Let $\{x_i\}$ be a sequence of vectors obtained by blocking a scalar source. A vector quantizer is defined by:

(a) A finite set of vector output points $\{y_i\}$, $i = 1,\ldots,N$, constituting a codebook.

(b) An encoding rule which assigns to any input vector x_i an output point or vector $y_i = Q(x_i)$.

A non-negative distortion measure, $d(x_i,y_j)$, defined between each input vector, x_i and each output point or vector, y_j, is used to evaluate or optimize performance.

Associated with any N-point vector quantizer in the R^k vector space is a partition of a disjoint set S_i where:

$$S_i = \{ x \in R^k : Q(x) = y_i \} \tag{5.1}$$

Given an input joint density, $p(x)$, the vector quantizer is optimal if it minimizes the average value of the encoding distortion, D:

$$D = E\{d(x,Q(x))\} \tag{5.2}$$

where $Q(x)$ is the mapping associated with the quantizer defined above. This optimality definition corresponds to a fixed-N constraint.

The necessary conditions for minimization of (5.2) are a generalization of the Lloyd-Max conditions for the scalar case [33]:

(1) Given a partition $\{S_j\}$, the optimal output points, y_j, should be chosen to minimize the distortion corresponding to the "cells" S_j:

$$E\{d(x,y_j) \quad x \in S_j\} \leq E\{d(x,u) \quad x \in S_j\} \text{ any } u \in R^k \tag{5.3}$$

(2) Given the output alphabet $\{y_i\}$, the optimal partition $\{S_j\}$ is defined by the nearest-neighbor rule:

$$x \in S_i \text{ iff } d(x,y_i) \leq d(x,y_m) \quad m=1,\ldots, N \tag{5.4}$$

The iterative algorithm for designing a quantizer which satisfies the necessary conditions for optimality was analyzed by Linde, Buzo and Gray [34], and is sometimes called the LBG algorithm. This is a generalization of the Lloyd algorithm [33] for the scalar case. Linde et al.[34] pointed out that there is no need for a variational approach or for taking derivatives of (5.2) to find the necessary conditions. Actually, the form presented above may be derived by the manipulation of inequalities between the centroids and their associated output points. Further, if the input has a probability of distortion and the fidelity criterion is the usual Mean Square Error (MSE), it can be shown that the first condition is equivalent to:

$$y_j = E\{x \mid x \in S_j\} \tag{5.5}$$

i.e. the output vectors should be the centroids or the centers of mass for each partition S_j.

The codeword generated with this algorithm achieves a local optimum (stationary point) under the given error criterion (distortion measure). One of the problems which frequently occurs when using this algorithm is that an output point vector may be assigned to a cluster containing only a few, if any, input vectors and hence may contribute very little to the reduction of the overall distortion. The main remedy for this "empty cell problem" is the choice of an initial codebook which would avoid empty cells. This

requires knowledge about the final codebook, which is generally unavailable.

An approach to solve this problem, is to begin a codebook of only two output points (one bit) and to build a codebook of b+1 bits from a codebook of b bits by splitting every output point into two new output points. Although this technique provides good experimental results [36], it does not guarantee the absence of empty cells from the resulting codebooks.

An alternate solution to the "empty cell problem" is to use some features of the ISODATA algorithm in pattern recognition [37]. This algorithm allows for splitting and lumping of clusters at every iteration at the cost of a substantial increase in complexity.

As a compromise for the design complexity and the empty cell problem, a modified LBG type algorithm is proposed. The flowchart of this algorithm is presented in Figure 5.1. As can be seen from the flowchart, after the clustering operation, the number of input vectors assigned to every cluster is checked. If the cell is empty, the corresponding output point is deleted and a new output point is defined by splitting the previously obtained output point representing the cluster with the highest distortion. The splitting is executed by multiplying all the components by a constant factor. The splitting operation preserves the highest distortion output point and replaces the empty cell output

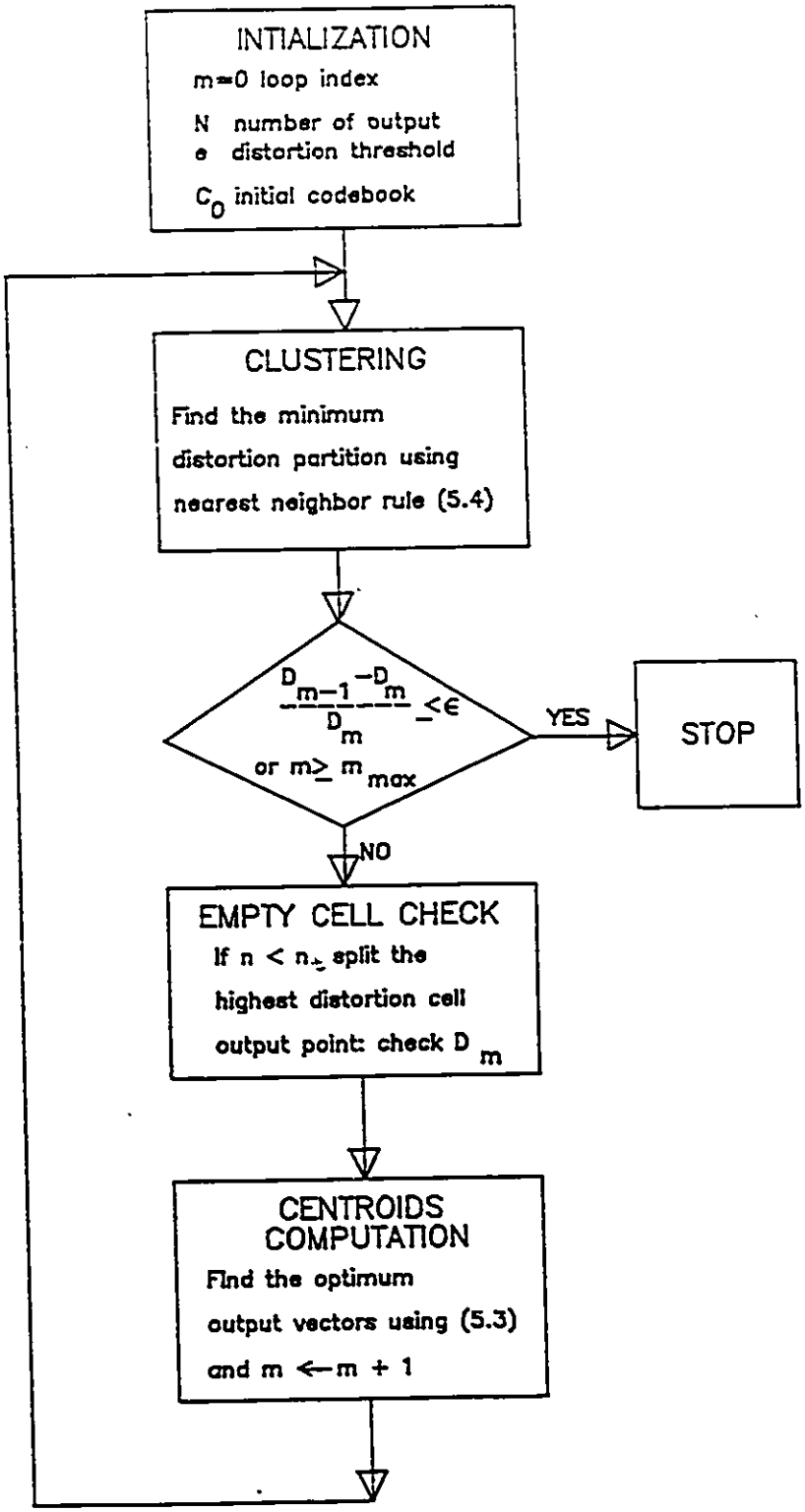


Figure 5.1 Codebook Design

by a perturbed version of the highest distortion output point (this operation splits the highest distortion cluster). This results in a decrease in the overall distortion so that the convergence of the modified algorithm is assured.

5.2 DISTORTION CRITERIA

A general distortion criterion commonly used to quantify the performance of a system is the average distortion $E\{d(x,y)\}$ between the input and the reproduction. In practice, the long-term sampling average or time average

$$\lim \frac{1}{n} \sum_{i=0}^{n-1} d(x_i, y_i)$$

is used to evaluate the system performance. If the vector process is stationary and ergodic, then, with probability 1, the limit exists and equals the expectation $E(d(x,y))$ [37].

In the case where the objective of the VQ design is to maximize the segmental SNR (SSNR) rather than the overall SNR, using criteria based on distortion would be more suitable than using the mean square error [38]. The energy-weighted mean-square-error (EWMSE) introduced by Cuperman [39] is one such distortion criterion. Minimizing the EWMSE is equivalent to maximizing the SSNR. This suggests that

higher SSNR can be obtained by designing the codebook using the EWMSE criterion instead of the MSE criterion. This was proven by Cuperman [39].

For an input signal extending over K consecutive frames, the EWMSE is defined as:

$$EWMSE = \frac{1}{K} \sum_{i=1}^{K} \frac{E\{ \| x_n^{(i)} - y_n^{(i)} \|^2 \}}{E\{ \| x_n^{(i)} \|^2 \}} \qquad (5.6)$$

where $x_n^{(i)}$ and $y_n^{(i)}$ are the input vector and the quantized vector in the i th frame. In practice the expectation is implemented by time averaging.

It should be noted that the nearest-neighbour codebook search in actual VQ coding is not affected by the use of the EWMSE and only the codebook is affected. Thus only the codebook design needs more computation and the encoding complexity remains the same as in the case where MSE is use as the distortion criterion.

5.3 COMPLEXITY IN VECTOR QUANTIZATION

Given a vector quantizer with N output vectors, the coding procedure assigned to each input vector x_i, the "closest" output point y_i which minimizes the distortion measure $d(x_i,y_i)$. The direct way to do this is to compute the distances between the input vector x_i and each output point y_j for j=1,2,...,N. This "full search" procedure requires N distance computations for each input vector. If the MSE criterion is used and N is expressed as $N = 2^{kr}$, where r is the rate in bits/sample, and K is the dimension, the number of multiplications per input vector, MUL, is given by:

$$MUL = kN = k2^{kr}$$

Hence, if the rate in bits/sample is fixed, then the computational complexity grows exponentially with the dimension. It is easy to see that the required memory, MEM, for storing the N output points is given by:

$$MEM = k2^{kr} \text{ words}$$

where a "word" denotes the amount of storage needed for a real number.

Thus, the computational and memory complexity of a full search vector quantizer both increase exponentially with the dimension for constant rate. The only known results in

waveform vector quantization of speech using full search quantizers are limited to dimension 4 for a rate of 2 bits/sample and 8 for 1 bit/sample [30]. The resulting quantizers compare favorably with linear PCM coding, but the achieved overall performance is not adequate for practical systems due to their high complexity.

Theoretically, it should be possible to reduce the number of computations in a full search vector quantizer by exploiting the geometric structure of the partition. For example, if the distortion criterion is the regular MSE, then the regions S_i are separated by hyperplanes, and by checking on which side of a hyperplane a given input vector lies, a part of the output points might be eliminated from the full search procedure. An alternative to reduce the complexity of vector quantization is presented by sub-optimal vector quantizers described in [28][34].

5.4 VECTOR QUANTIZATION WITH ASYCHRONOUS DELTA MODULATED SIGNAL

Although vector quantization is an asymptotically optimal coding procedure (for large dimension), the results obtained, a signal-to-quantization-noise ratio in the range 15 to 20 dB, are not competitive with the prior state of the art in speech coding. This is due to exponential growth of the complexity versus dimension, characteristic of optimal

vector quantization [28][29]. Suboptimal vector quantization reduces the complexity, but it results in a significant degradation in performance. The performances obtainable with this constraint are not adequate for applications where communications or toll quality speech coding are needed [38][39][40]. Nevertheless, the vector quantizer achieves an impressive improvement in performance over the conventional scalar quantizer: over 9 dB in dimension k=4 at a rate of 2 bits/sample [38][39].

To exploit the versatility of vector quantization, we propose to use a vector quantizer as a building block in our asynchronous system. The general configuration proposed for our vector quantized asynchronous delta modulation (VQASDM) system is presented in Figure 5.2. The asynchronous samples and their corresponding IBIs generated by the CASDM are translated by the time quantizer, with a built-in delay unit, into a sequence of synchronous signal samples whose signs, positive and negative, and amplitudes represent the upward/downward steps and IBIs of the ASDM samples respectively. These sequence samples are equally spaced with an interval equal to the average IBI's, $\overline{\Delta t}$, of the asynchronous samples generated by the CASDM. The sequences of such synchronous samples are then block coded with a vector quantizer.

The transmitter output is the index of the codebook entry chosen to represent a block of samples and IBI's

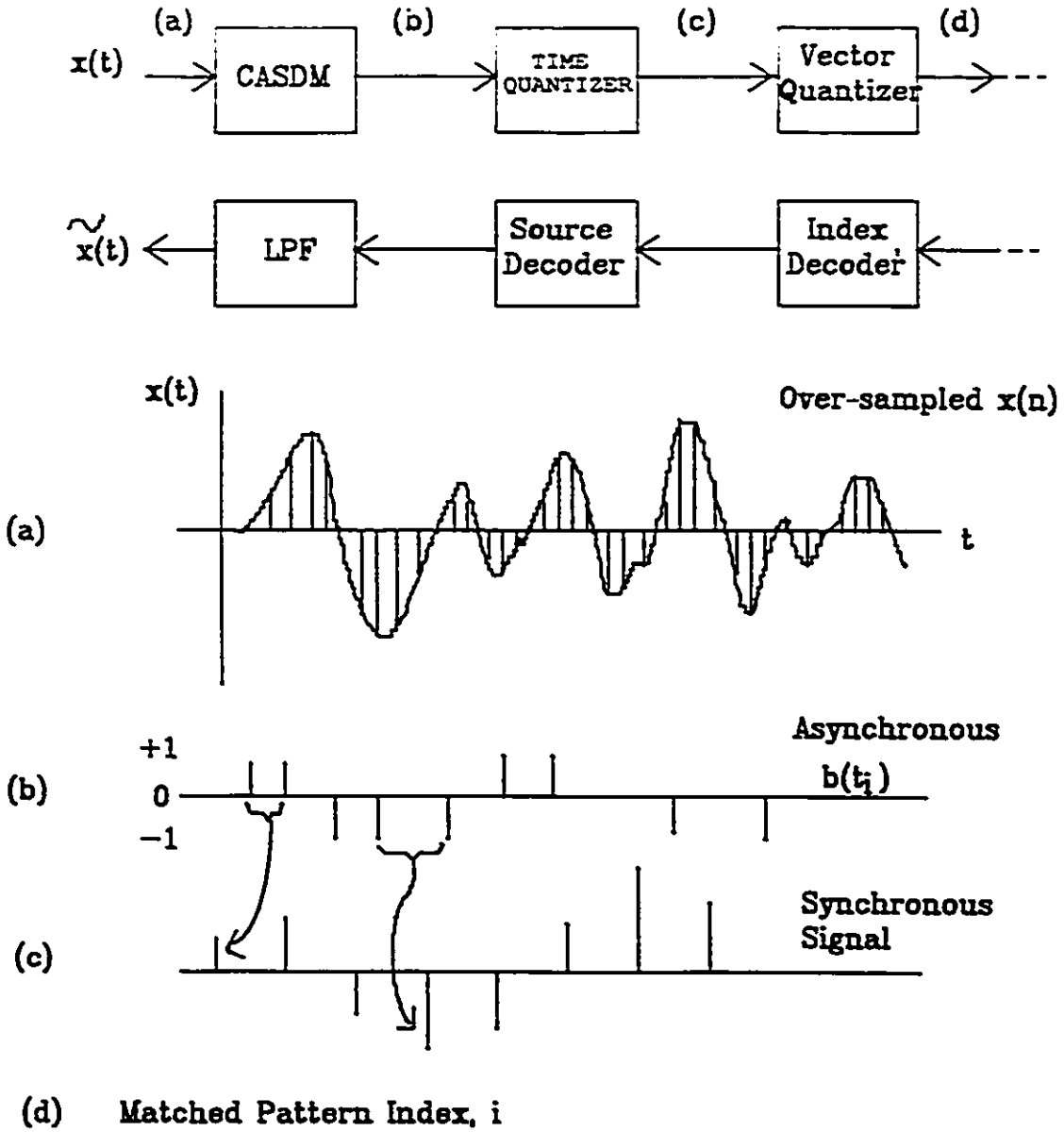


Figure 5.2 Application of Vector Quantization on CASDM

generated by the ASDM. Unless otherwise specified, a full search through the codebook is assumed for finding the nearest neighbor to the block under the chosen mean square error criterion.

The receiver retrieves from its codebook the vector corresponding to the transmitted index, i, and uses a reverse process to decode and reconstruct the signal.

The advantage of such composition is that a low transmission rate can be achieved. However, higher quantization noise is expected due to fact that a vector quantizer is being cascaded to the ASDM.

The simulation results are presented in chapter 6. Different values of *r*, bits/sample, and *k*, dimension, are used for the vector quantizer. A training sequence of 1024 samples is used to generate the code book. A transmission bit rate reduction of up to 30% over the Huffman coding scheme, as described in chapter 4, is obtained by the use of the vector quantizer.

CHAPTER VI

6.0 SIMULATION RESULTS

6.1 INTRODUCTION

In section 6.3, various run-length encoding schemes, Huffman coding, differential run-length coding and run-length coding with synchronization word, are compared with respect to transmission rate and the signal-to-quantization noise ratio. Error-free channels are assumed in all cases.

The following coding techniques, which are described in chapter 4, sections 4.3, 4.4 and 4.5, are simulated:

1. Huffman coding
2. Differential run-length coding
3. Run-length coding with synchronization word

Various adaptation schemes, step size, corridor size and overshoot suppression, are simulated on the Corridor Asynchronous Delta Modulator, CASDM.

In section 6.4, vector quantizers of different dimensions are simulated. A full search technique is employed, as our main interest is in reducing the transmission rate while maintaining the same signal-to-noise ratio as that obtained by using run-length codes.

A speech simulator consisting of both a pulse generator, for generating a voiced signal, and a noise generator, for producing an unvoiced signal, is used for speech-like signal generation. The model is as shown in Figure 6.1. In this model, samples of speech excitation-like waveform are assumed to be the output of a time varying digital filter that approximates the transmission properties of the vocal tract and the spectral properties of the glottal pulse shape. Since the vocal tract changes shape rather slowly, it is reasonable to assume that the band-pass filters, which model the vocal tract, have fixed characteristics over a time interval of on the order of 10 ms. For a voiced signal, an impulse train is fed through the filters. The spacing between the impulses corresponds to the fundamental period of the glottal excitation. For an unvoiced signal, the filters are excited by a random-number generator that produces flat-spectrum noise. In both cases, the amplitude control regulates the intensity of the input to the filters to provide smooth transitions from voiced signal to unvoiced signal, and vice versa. The simulated signal has a bandwidth of 4000 Hz. The signal is sampled at 8 times the Nyquist rate, i.e. 64 KHz, hence the sampling period is 15.6us. For each speech segment, we specify the pitch period, the voiced/unvoiced decision, and the bandwidth and the center frequencies of the four formants.

The ranges of the parameters chosen in our speech model are shown below:

Pitch period : $6 \text{ ms} \leq p \leq 12 \text{ ms}$

Voiced/Unvoiced time ratio : 1/1

Voiced/Unvoiced cycle : 100-120 ms

Voiced/Unvoiced power ratio : 30dB - 40dB

Formant Bandwidth :
$$50 \leq B_1 \leq 110$$
$$50 \leq B_2 \leq 110$$
$$100 \leq B_3 \leq 400$$
$$100 \leq B_4 \leq 400$$

Formant Center Freq :
$$400 \leq f_1 \leq 700$$
$$700 \leq f_2 \leq 1800$$
$$1800 \leq f_3 \leq 2900$$
$$2900 \leq f_4 \leq 3900$$

The choice of the particular values listed above is driven by a Gaussian distributed function.

The simulator is driven by a clock with period t_{min} = 15.6us. The system is allowed to step up or down only at each clock instant and the time intervals between the corridor crossings may assume only integer multiples of the clock period. Thus the resulting multiplicity becomes the run-length to be encoded for transmission.

Before stating the results, we have to define the parameters which evaluate the performance of the simulation.

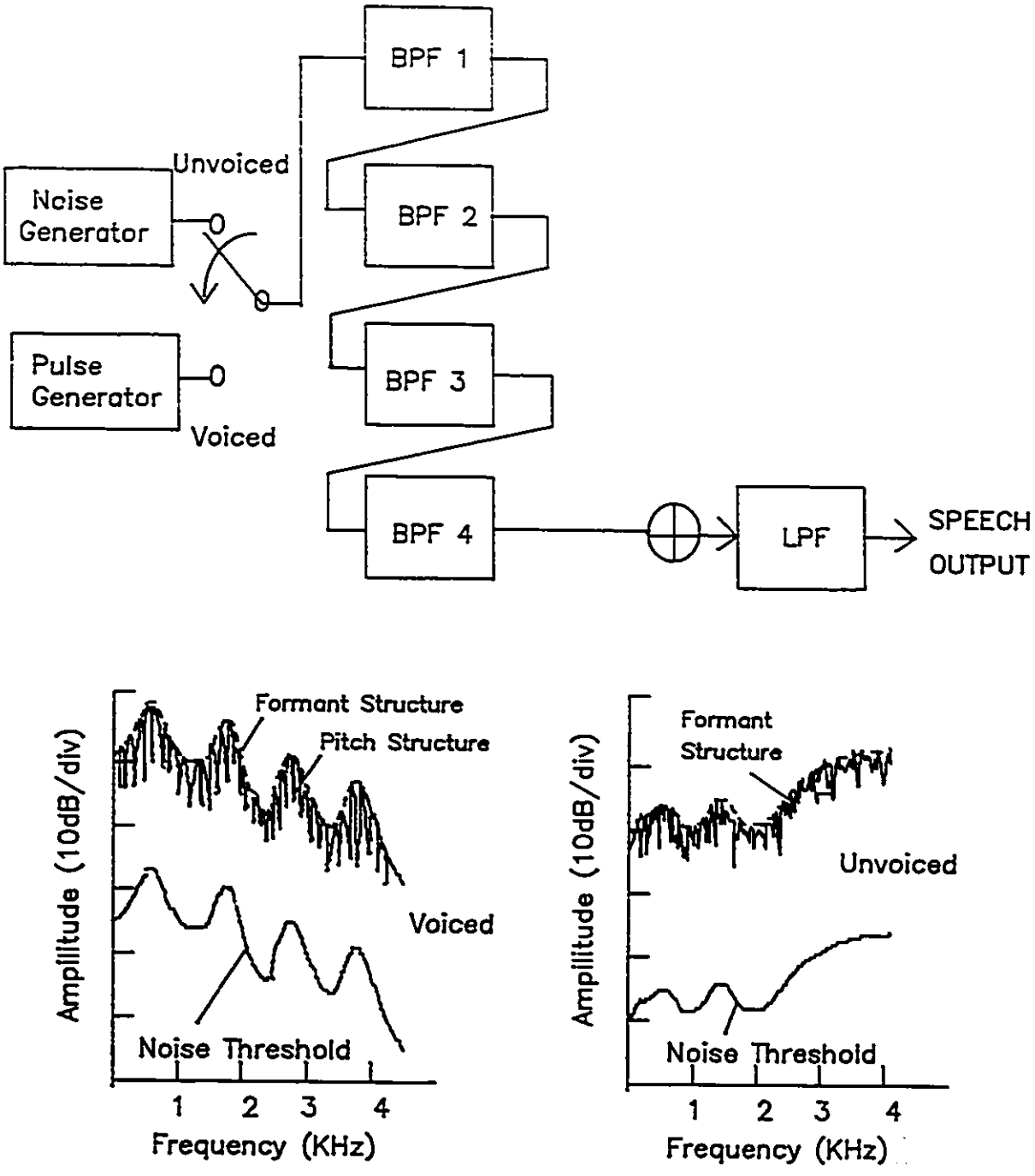


FIGURE 6.1 MODEL OF SPEECH-LIKE WAVEFORM SIMULATION

(1) Signal to noise ratio, SNR (dB):

$$SNR = 10 \log_{10} \frac{\Sigma x_i^2}{\Sigma (x_i - y_i)^2} \qquad (6.1)$$

where x_i is the input signal and y_i is the reconstructed signal.

However, a more suitable evaluation to the proposed model would be a subjective (listening) test with real speech signal applied.

(2) Oversampling factor, k :

$$k = \frac{\text{\# of samples taken by CASDM in the segment}}{\text{\# of Nyquist samples in the same segment}} \qquad (6.2)$$

where the Nyquist rate is 8 KHz.

(3) Transmission rate, B :

$B = k \times \text{Nyquist frequency} \times \bar{b}$

where $\bar{b}$ = average bit/signal sample.

A typical simulated CASDM waveform is shown in Figure 6.2. The estimated signal obtained at the output of the CASDM is shown tracking the input signal.

The relationship between the over-sampling factor, k, and the signal-to-noise ratio, SNR, of the CASDM is obtained through simulation and is shown in Figure 6.3. The result obtained is as expected; a higher value of k provides a higher SNR at a cost of a higher transmission rate.

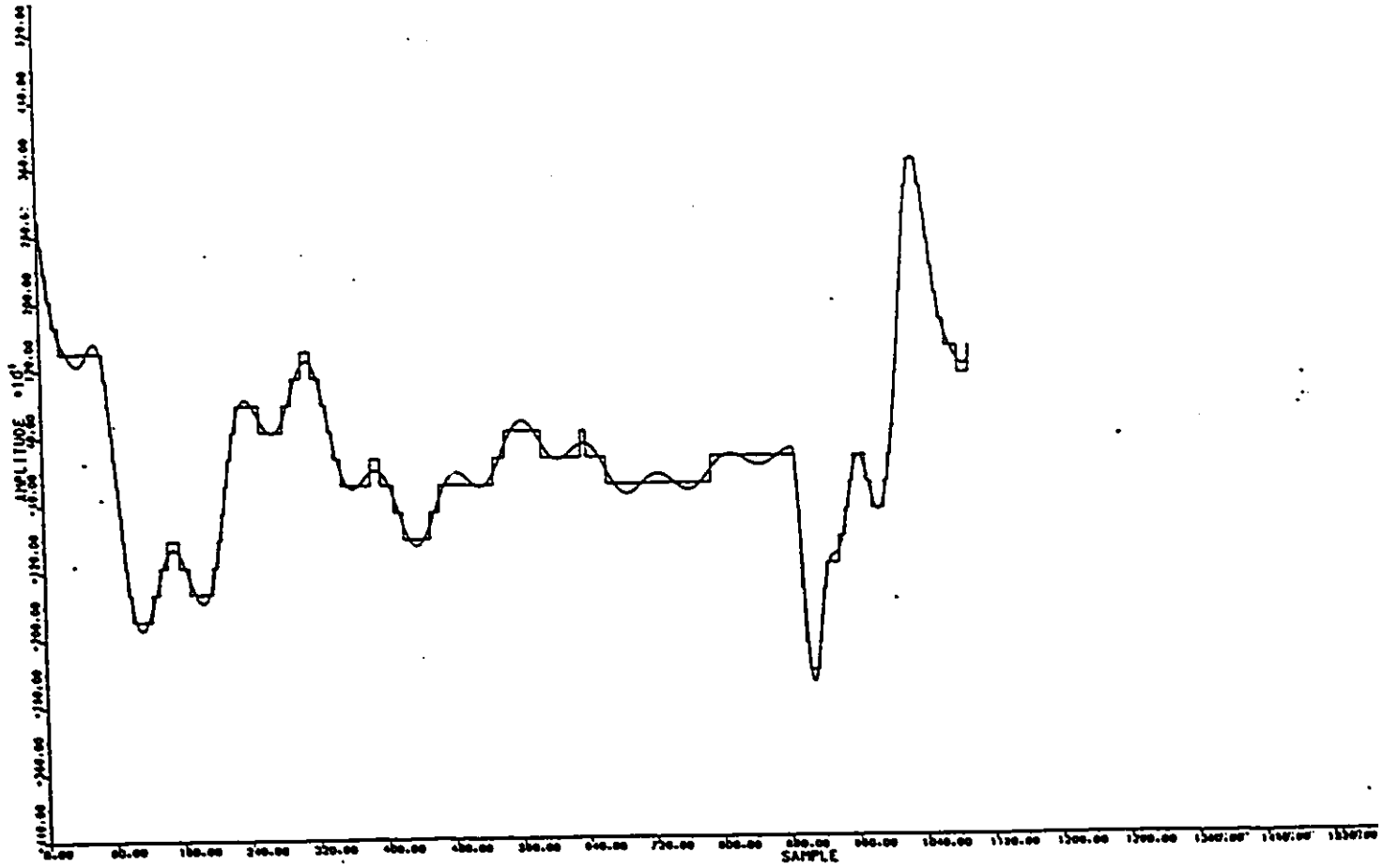


Figure 6.2 Simulation of the CASDM

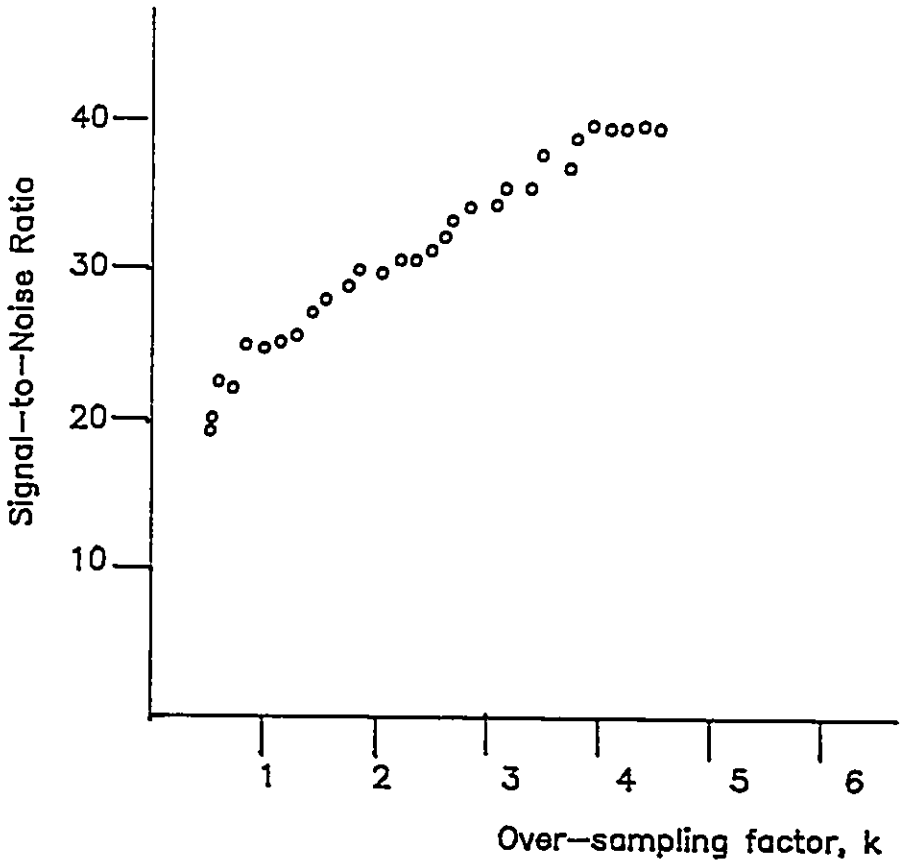


Figure 6.3 SSNR vs Over-sampling Factor, k for CASDM

6.2 ADAPTIVE SCHEMES

Recall that in section 2.2.2, several quantization adaptation schemes are mentioned. For the sampling time interval, the minimum sampling time is chosen to be 15.6μs whereas the maximum allowable sampling time interval is dependent on the run-length code employed. In section 6.3, the simulation results of different run-length encoding schemes are described.

6.2.1 Step Size Adaptation

In the case of step size adaptation, recall (2.14), the step size is increased by a factor of $C > 1$ when (+1,+1) or (-1,-1) occurs and is reduced by a factor of $D < 1$ if an alternating pattern (+1,-1) or (-1,+1) occurs. Different C,D pairs (1.5, 0.66), (1.40, 0.75), (1.15, 0.95), (1.95, 0.4) are used in our simulation, however no noticeable improvement is observed.

6.2.2 Corridor Width Adaptation

Another adaptive scheme for CASDM is to change the corridor size according to the signal amplitude after each sampling instant. For example one can assign a larger corridor size if the signal amplitude is large or vice

versa. With such a scheme one allows for rougher quantization at larger signal amplitudes, thus incurring into some SNR loss for a trade off of less samples for transmission. The corridor adjustment is implemented according to Max-Lloyd quantizer steps, i.e., the corridor size is expanded by the same ratio as that of the corresponding step size in the quantizer to its minimum step size. However, no significant improvements are brought about on the simulated signal by the use of this scheme.

Consider a corridor of fixed size within a speech segment as seen in Figure 6.4. The corridor aperture tends to be smaller tangentially to the signal when the slope of the signal is greater and vice versa. In other words, the tangential corridor width is inversely proportional to the slope of the input signal. One advantage of keeping the corridor aperture constant is that the estimated signal is forced to track the rapidly changing input signal and as a result the error due to slope overload can be reduced.

The sample signal is processed through a CASDM with a corridor width factor of 0.2, which is 20% of the variance of the input signal.

6.2.3 Overshoot Suppression

It is known that overshoot can cause considerable subjective annoyance, but since it has a limited effect on

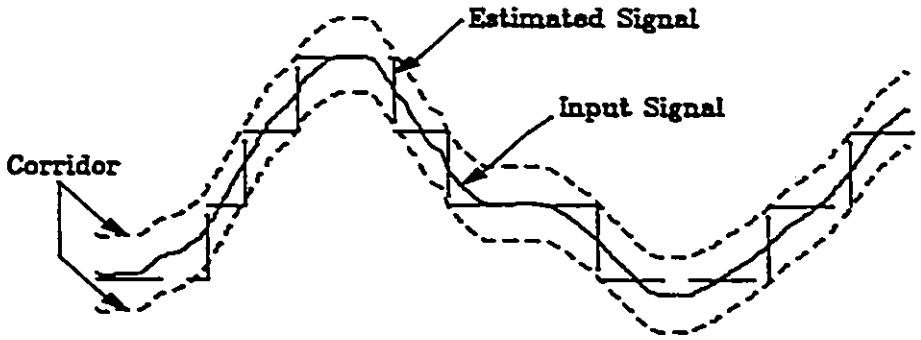


Figure 6.4 CASDM with Constant Corridor Aperture

the mean SNR [13], it is important to examine the effect of overshoot suppression on the estimated signal.

An example of overshoot is shown in Figure 6.5. In order to limit the overshoot, a digital comparator is used to detect the condition:

$$|X_n - Y_{n-1}| < (\Delta_{max} - J)$$

where X = input signal,

Y = estimated signal,

Δ max = maximum step size and

J = maximum overshoot permitted.

The simulation results show a slight improvement of less than 0.5 dB in the SNR over the scheme with no overshoot suppression.

6.3 RUN-LENGTH ENCODING

6.3.1 Huffman Coding

For the simulated speech signal, the average Huffman code word lengths are found for different maximum allowable run-length, L_c. Figure 6.6 shows that if the maximum encodable run-length is 128, an average saving of more than 2 bits (out of 7) can be achieved by the use of Huffman coding.

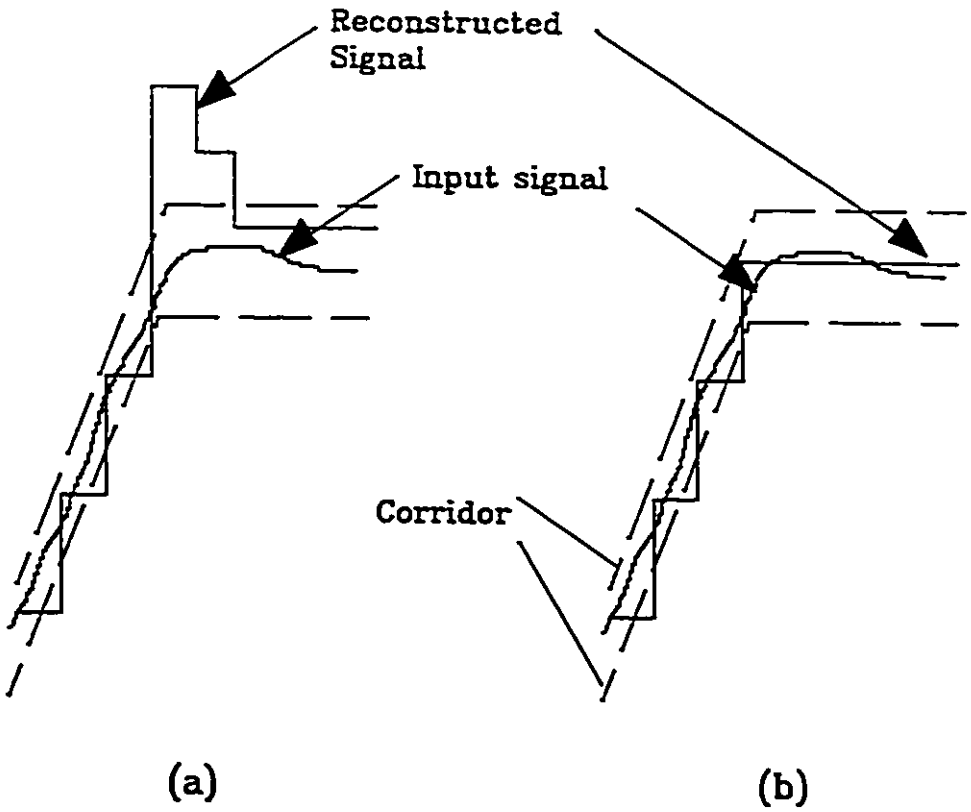


Figure 6.5 The Effect of Overshoot Suppression
(a) CASDM without overshoot suppression
(b) CASDM with overshoot suppression

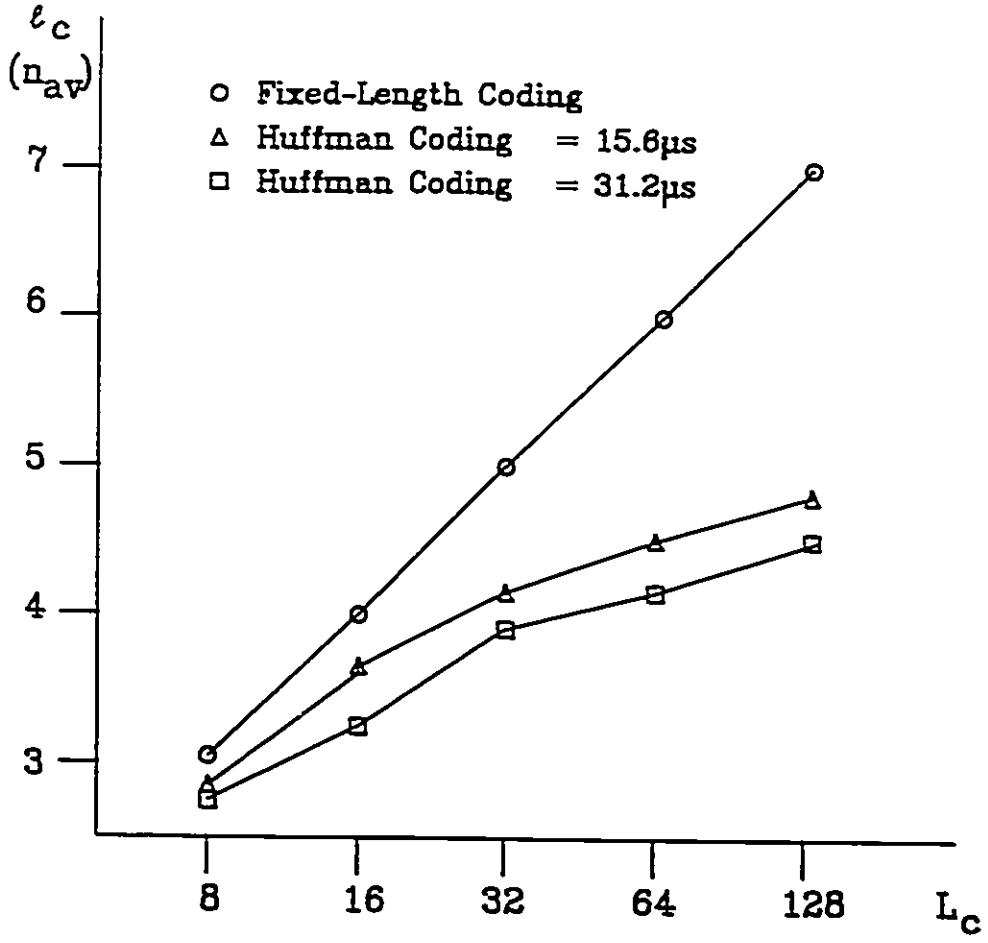


Figure 6.6 Average Number of Bits/Code Word versus Run-Length Constraints

If the minimum run-length quantum is chosen to be 15.6μs, in which case the probability that a run-length is equal to 1 is 0.001, then the Huffman code generated is listed as given below for L_c = 16:

Run-length	Code Word
1	000001
2	0010
3	011
4	010
5	101
6	111
7	0001
8	1000
9	1001
10	1101
11	00001
12	00110
13	00111
14	11000
15	11001
16	000000

The average transmission rates corresponding to different fixed maximum run-lengths, L_c, are shown in Figure 6.7. The transmission rates are obtained by using (6.2), (6.3) and the values of k from Figure 6.3.

6.3.2 Huffman Coding with Synchronization Word

As described in section 4.3, if a fixed maximum run-length is imposed, then the demodulated signal will be displaced in time. In the simulation of the CASDM, synchronization words are inserted between blocks of code words before transmission. The improvement, in terms of

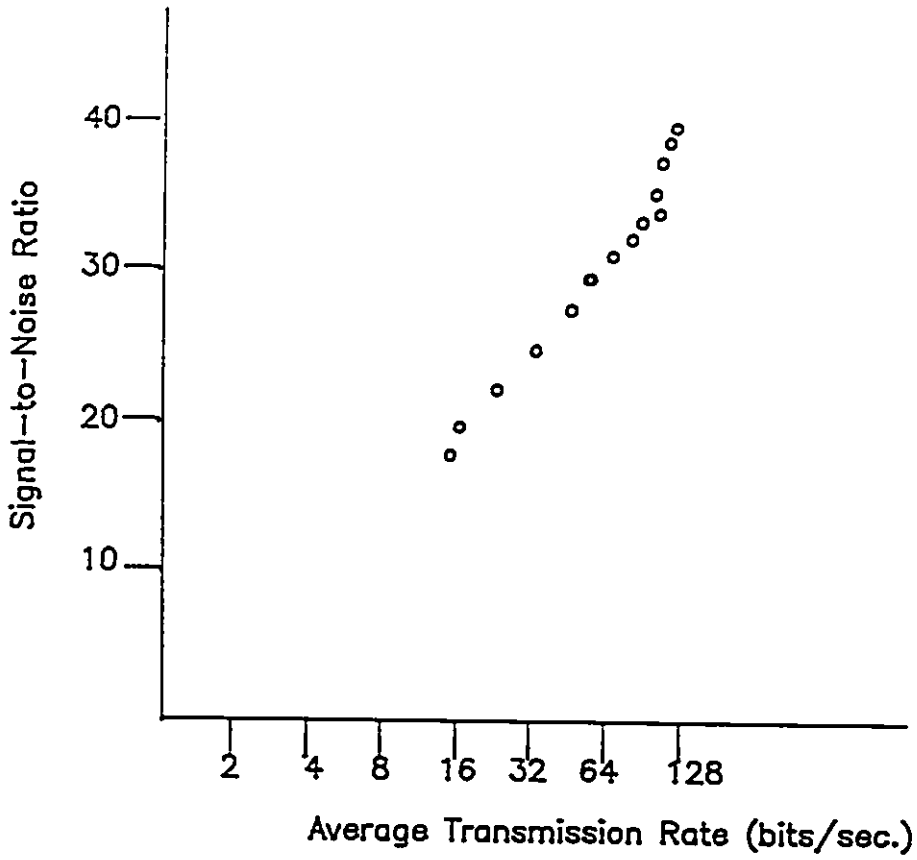


Figure 6.7 SNR vs Average Transmission Rate for Huffman Coding with $\tau = 15.6\mu s$

signal distortion is found for various values of n_s, the number of code words transmitted between two synchronization words.

The simulation results in Figure 6.8 show that the SNR is very low for $L_c \leq 32$ when no synchronization words are transmitted. However, the calculated SNR may not be a good criterion for the subjective performance. As shown in Figure 6.9, although low values of SNR are obtained as a result of the build-up of the time displacements, the shape of the estimated signal is not necessarily seriously distorted. The same SNR value is obtained whether or not synchronization words are employed when L_c is 128, because none of the simulated run-lengths have exceeded 128.

It is known that for speech signals, the intelligibility is less disturbed if the phase information is degenerated. A more useful criterion would be the absolute difference of the magnitude spectrum of the input signal and the reconstructed signal. However, for video signals, such a criterion will not be appropriate, since visual perception is very sensitive to phase distortion.

6.3.3 Differential Run-length Encoding

The simulation results presented in the following paragraphs are based on the use of fixed length encoding on the run-length differences. The run-lengths are assigned

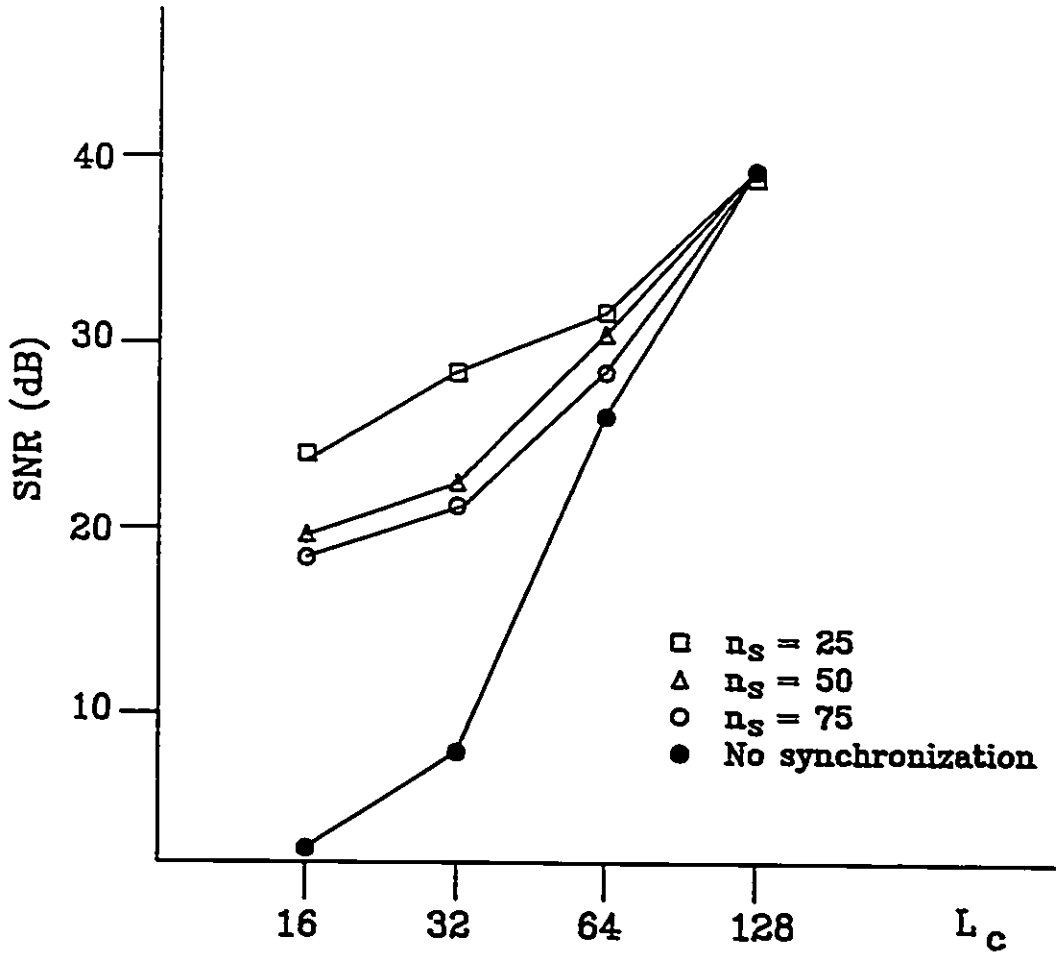


Figure 6.8 SNR as a function of Run-length constraints for different synchronization intervals

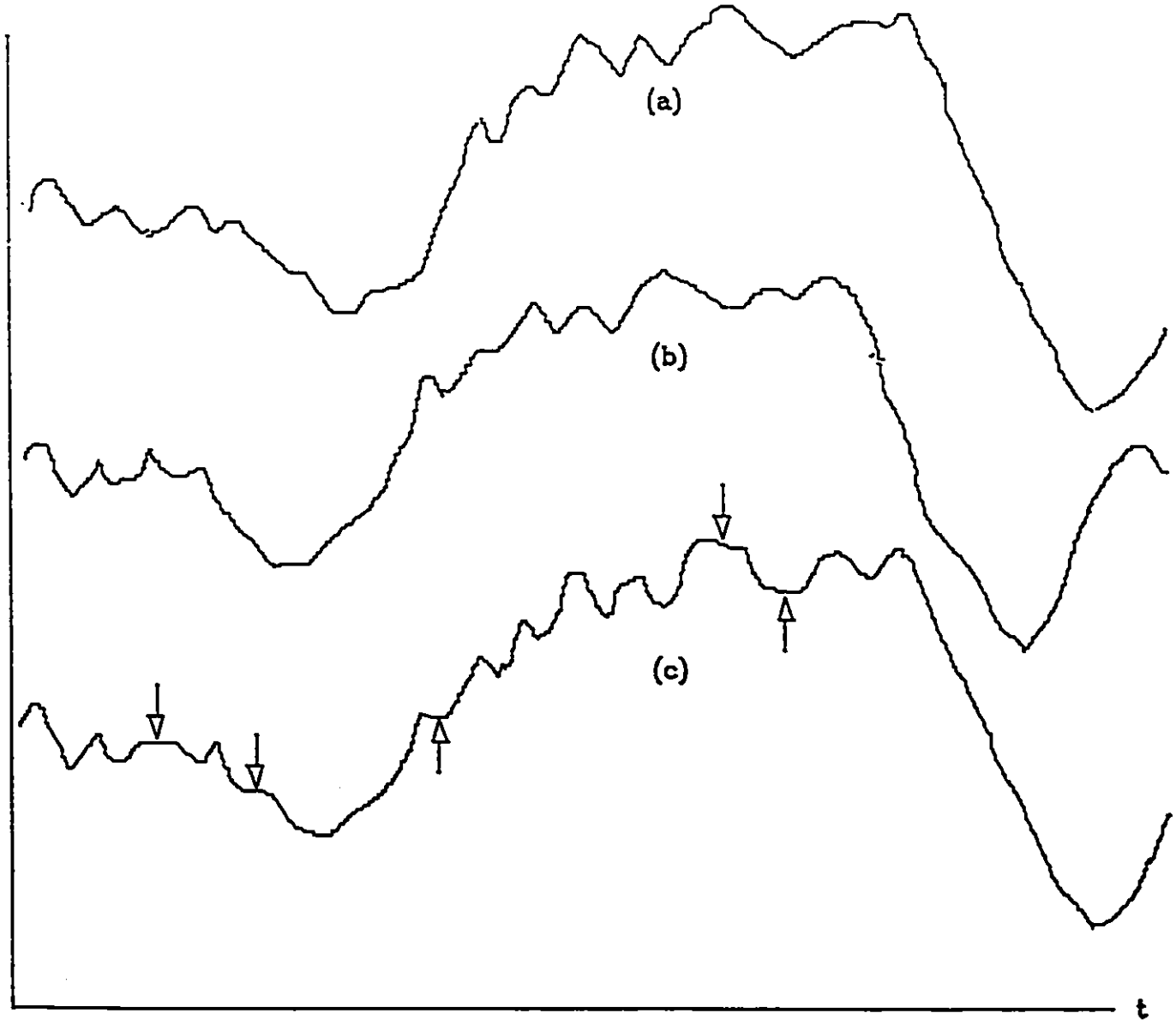


Figure 6.9 Signal Reconstruction

(a) Input Signal
(b) Demodulated signal with no synchronization words
(c) Demodulated signal with synchronization words inserted within blocks of 75 code words. (arrows indicate instants of synchronization)

positive values if the level change of the CASDM corresponds to an upward step, and negative values are assigned otherwise. The algebraic difference between two consecutive run-lengths is then encoded by a sign bit. If the absolute run-length exceeds the encoder capacity, then the difference is encoded as the greatest integer that could be expressed by the encoder. Unfortunately, if an error is made in encoding, then the true value of the run-lengths can never be recovered. Thus, a faster accumulation of distortion may occur.

In the simulation, certain run-lengths are assumed to be transmitted for synchronization in certain intervals. Upon the reception of the synchronization word, the decoder recovers the correct run-length value. This prevents the random time displacements in the reconstructed signal. Since both forward and backward time shifts are likely to occur, the building up of time displacements in a single direction is not considered.

For the simulated speech signal, the SNR values are shown in Figure 6.10. A SNR of 39.5 dB is obtained even if no synchronization words are transmitted when L_C=128, because none of the simulated differential run-lengths exceeded 128.

If the synchronization words carry information about both the actual run-lengths and the quantized signal amplitudes, then better values are obtained as seen in

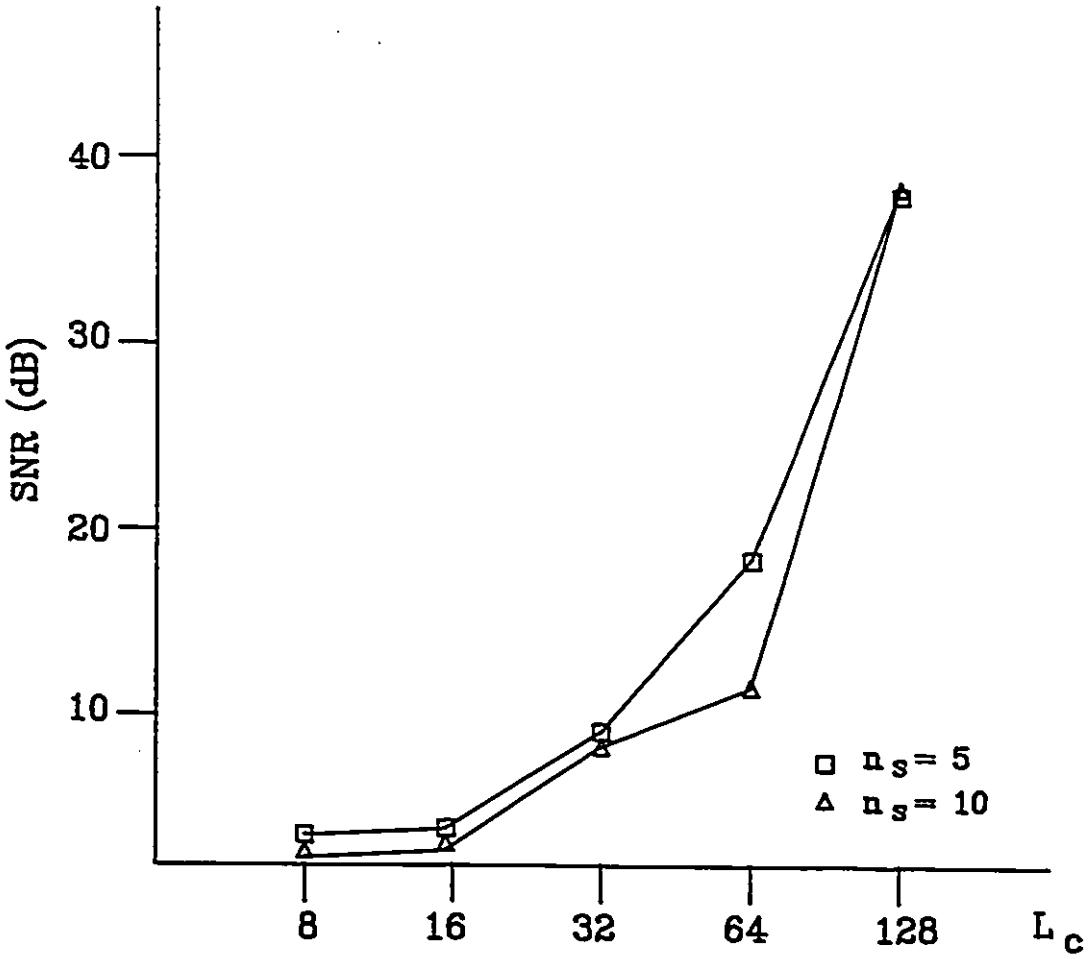


Figure 6.10 Differential Run–Length Encoding

Figure 6.11. However, an increase in the transmission rate is expected since additional bits are required to carry the synchronization information. The simulation results show that the performance is as satisfactory as those provided by Huffman codes with a synchronization word.

The performance of the Huffman coding CASDM is compared to that of the Constant Factor Delta Modulator, CFDM, [42] at 24 Kb/s, using the same speech model. The results are shown in Figure 6.12. As seen in the figure, the only short-coming of the CASDM is that it provides a narrower dynamic range, of about 40 dB, as compared to those of the CFDM, about 60 dB. However, in speech applications, a dynamic range of 40 dB is considered to be sufficient [41].

6.4 VECTOR QUANTIZATION

In order to verify the performance of the vector quantizer, as described in chapter 5, simulated speech-like signals (Gaussian and Laplacian) are applied. Results in Figure 6.13 show that the performance is comparable to that of conventional vector quantizers [41].

In the application of vector quantization, we expected to attain a lower transmission rate compared to the run-length code. However, a lower SNR is also expected since

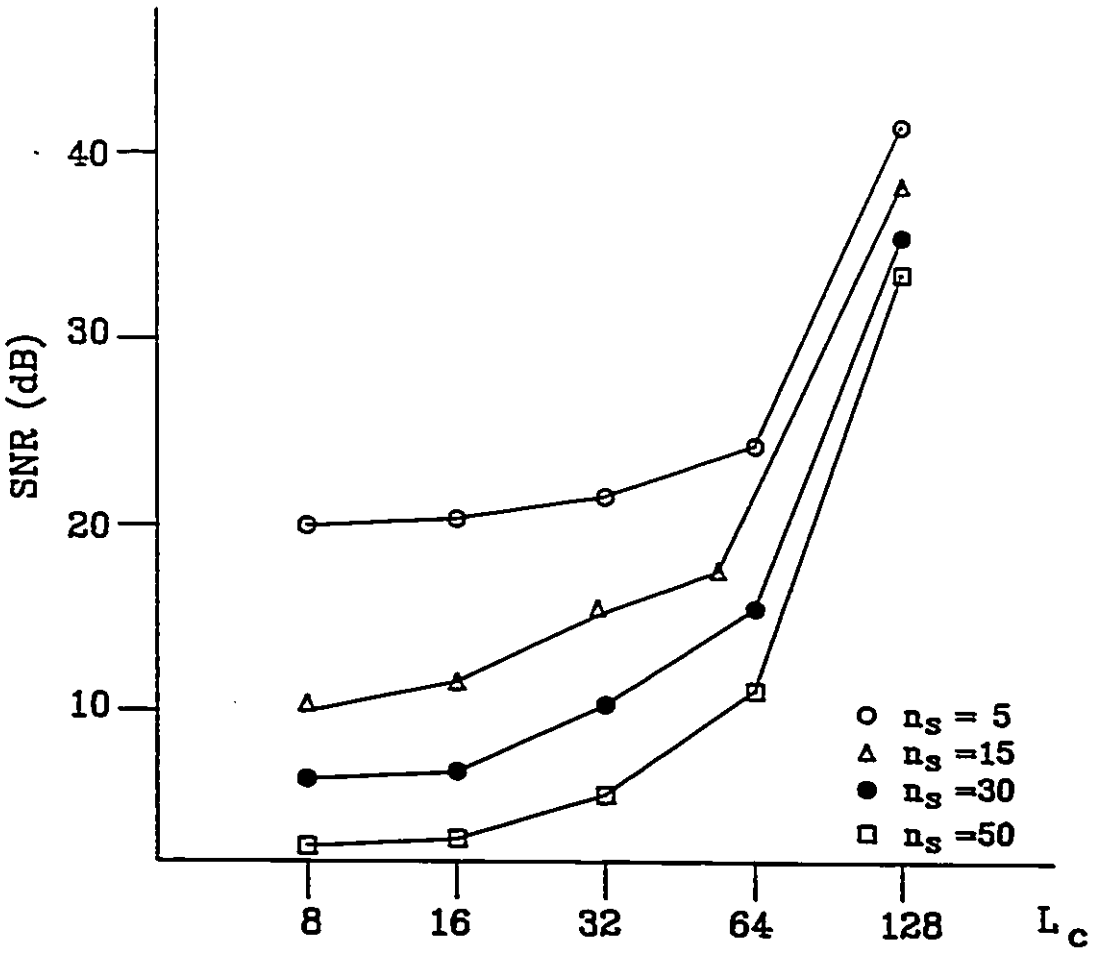


Figure 6.11 Differential Run-Length Encoding
With Synchronization Words

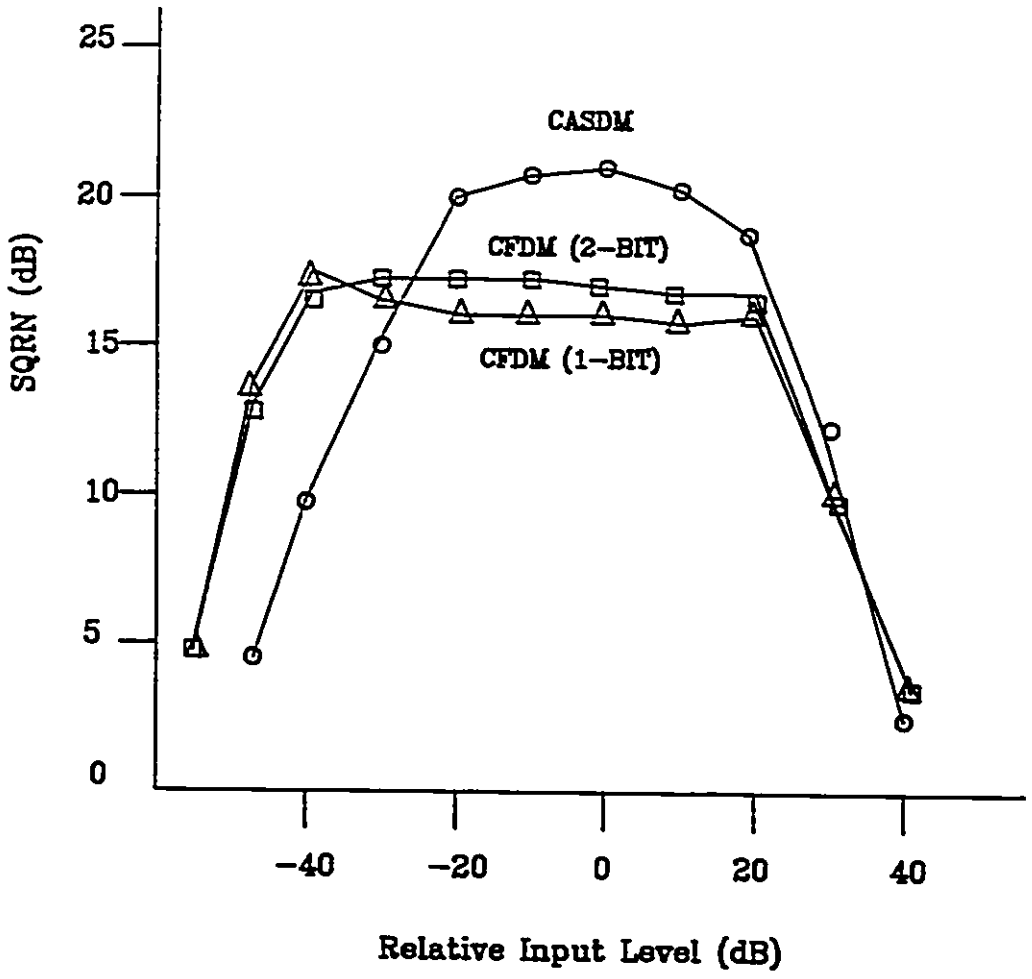
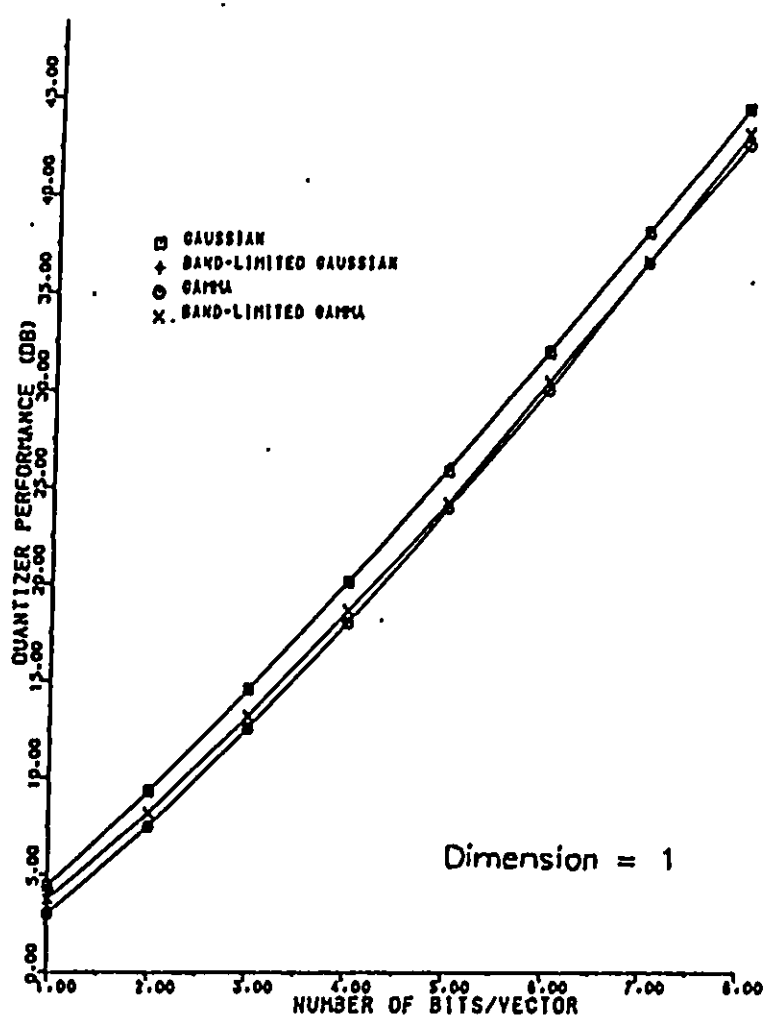
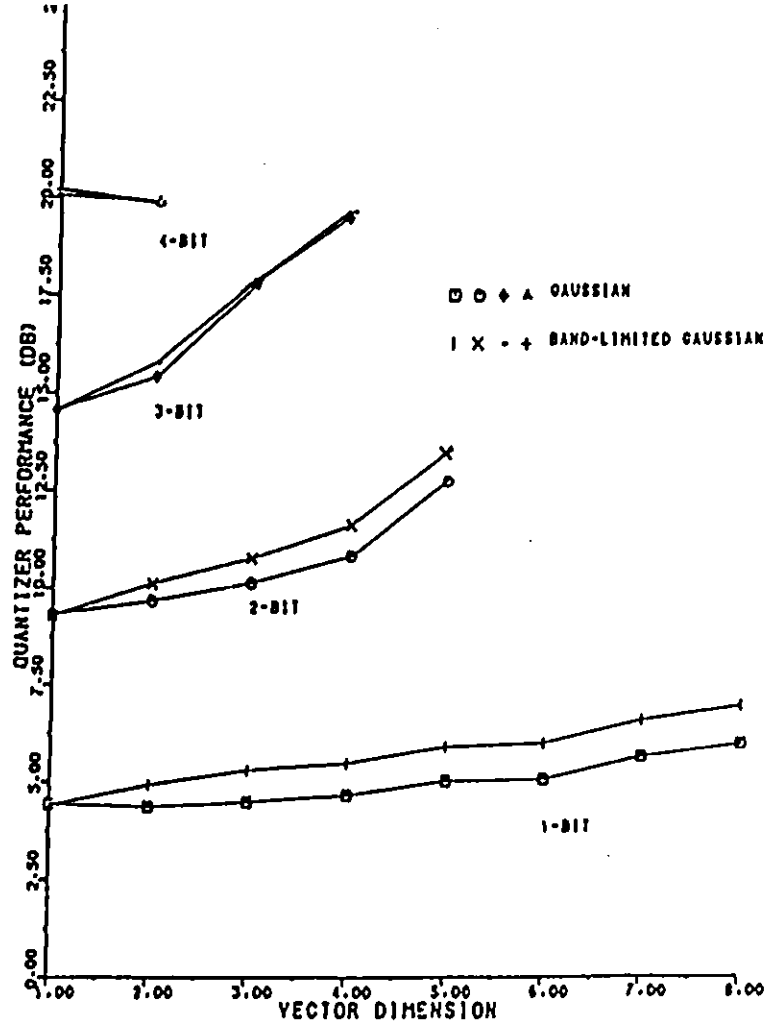


Figure 6.12 SQNR COMPARSIONS OF CFDM AND CASDM FOR SIMULATED SPEECH SIGNAL AT 24 KB/S



QUANTIZER PERFORMANCE VS QUANTIZER RATE



QUANTIZER PERFORMANCE VS VECTOR DIMENSION

Figure 6.13 Performance of Vector Quantizer

additional quantization is introduced by the vector quantizer.

In order to compare the performances of the vector quantized CASDM and that of the run-length coding, output rates of the vector quantizer are fixed by selecting appropriate parameters, i.e. the total number of waveform patterns and the dimensions, k. The bit output rate of the VQ (vector quantizer) is given as :

$$r = \frac{log_2 \text{ (total \# of patterns)}}{\text{Dimension of the VQ}} \quad \text{bits/sample} \qquad (6.4)$$

The transmission rate is then :

$$B = k \times \text{Nyquist frequency} \times r \quad \text{bits/sec.} \qquad (6.5)$$

where k is the over-sampling factor defined in (6.2)

The splitting algorithm, as described in section 5.2, is used to generate the waveform patterns.

Figure 6.14 shows the SNR of the vector quantized CASDM versus the transmission rate. A higher SNR ratio is obtained compared to that obtained by using Huffman coding (Figure 6.7), at a cost of higher complexity.

The results show that higher compression ratio can be obtained by the use of vector quantization over run-length encoding. However, we should focus our interest on the

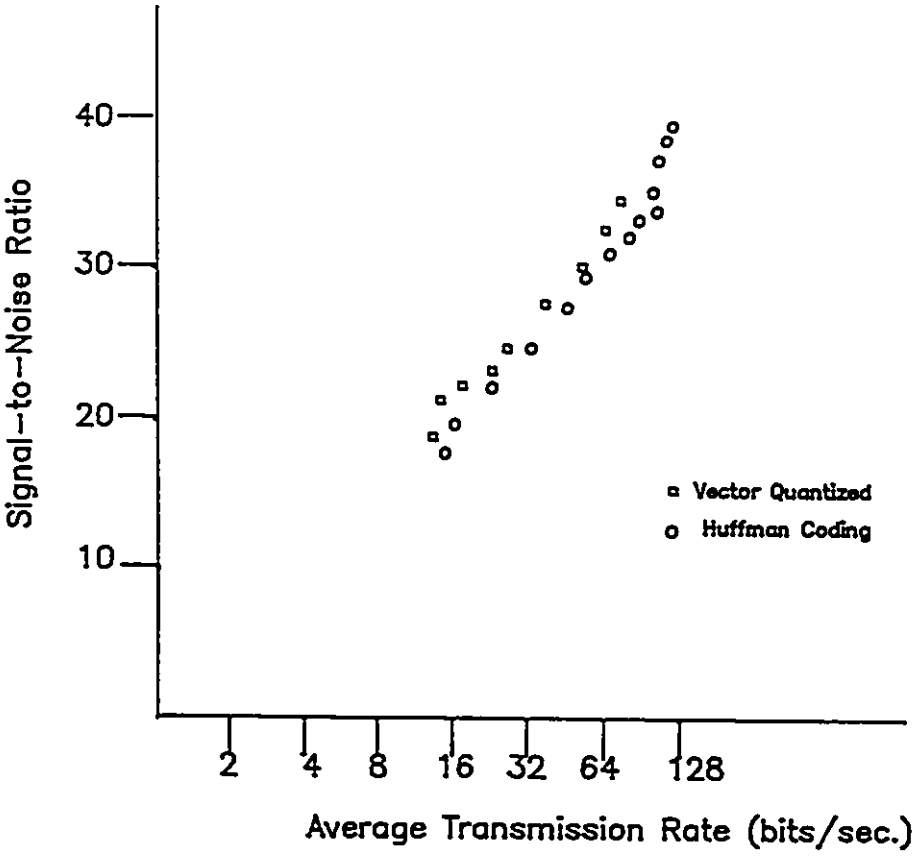


Figure 6.14 SNR vs Average Transmission Rate for Vector Quantized CASDM

transmission rates below 24 Kb/s, since the complexity of the VQ at higher bit rate is so high that it is not practical for implementation.

CHAPTER VII

7.0 CONCLUSION

Although asynchronous delta modulation systems are more robust compared to delta modulation and PCM, their potentials have not been fully exploited. This is mainly due to the problem of the transmission of asynchronous samples over synchronous channels. However, if we run-length encode the asynchronous samples as well as the sampling intervals, both synchronous transmission and transmission bandwidth reduction can be accomplished. The time quantization error introduced by this process is found to be negligibly small [less than (1/8)'th, as found in (3.27), wiht N=4] compared to the amplitude quantization distortion.

Run-length encoding, run-length encoding with sychronization, and differential run-length encoding have been compared on the basis of simulation results for simulated speech signals and also to several conventional speech modulation systems. An improvement of up to 5 dB in SNR is obtained by the use of Huffman coding on the run-length over CFDM at 24 Kb/s.

Another method of transmitting the non-uniformly spaced samples on synchronous channels is by block coding the asynchronous samples and the sampling intervals with vector quantization. Higher data compression is achieved compared

to run-length encoding. A compression ratio of over 0.25 (Figure 6.14) is obtained in transmission rates, 16 to 24 Kb/s, over the Huffman coding.

The run-lengths generated by the CASDM are encoded, either by Huffman codes or vector quantization, and are transmitted synchronously by the use of a buffer. Thus the transmission bandwidth required is that corresponding to the average rate, rather than the maximum, of the information contained in the signal. The main problems concerning the buffer are the overflow distortion and the system delay. Especially in two-way communication systems, the delay introduced by the buffer is a limiting factor on the choice of the buffer size. Although this research is concerned with non-uniformly sampling systems, the analysis of the buffer is made on a more general basis. In-depth studies on the buffer requirements are required for speech signals.

In this research, only computer simulated speech was used for the evaluation of the proposed system. Experimental work needs to be carried out with real speech signal to verify the system performance.

In the course of this research work, channel coding problems have not been considered. A remaining line of investigation is the employment of unified signal and channel coding techniques.

This study is also confined to single channel communication systems. However, the development principles can be used in a multiplexed system, such as the T1 networks, where it is possible to use either one buffer for each channel or a common buffer.

The vector quantized or run-length encoded CASDM yields a viable and attractive alternative for speech coding in the range of 12 to 24 Kb/s. Further improvement is feasible if one can provide an effective encoding scheme for the non-uniform sample intervals.

REFERENCES

[1] R. Zelinski, P. Noll, "Adaptive Transform Coding for Speech Signals," IEEE Trans. on ASSP, pp.299-309, Aug. 1977.

[2] T.A. Hawkes, P.A. Simonpieri,"Signal Coding using Asynchronous Delta Modulation," IEEE Trans. on Comm., Mar. 1974.

[3] R. Steele, Delta Modulation Systems, Halsted Press, New York, 1975.

[4] R. E. Crochiere, J. L. Flanagan,"Current Perspectives in Digital Speech", IEEE, International Conference on Communication 1982, pp. E6.5-6.13.

[5] H. Taub, D. L. Schilling, Principles of Communication Systems, McGraw Hill, Inc., 1971.

[6] R. Steele, Delta Modulation Systems, Halsted Press, New York, 1975.

[7] B. Sankur,"Signal Coding Properties of Asynchronous Delta Modulation," IEEE ICASSP., pp.1704-1708, 1982.

[8] E.M. Deloraine, S. Miero, B. Derjavitch, French Patent 923 140, 1946.

[9] H. Inose, et al.,"Asynchronous Delta Modulation Systems," Electronics and Communication in Japan, pp.34-42, March 1966.

[10] P.D. Sharma,"Characteristics of Asynchronous Delta Modulation and Binary Slope Quantized PCM Systems," IEEE Trans. on Comm., pp.32-37 Jan. 1968.

[11] N.S. Jayant,"Adaptive delta modulation with one-bit memory," Bell System Tech. Journal, pp.321-342, Mar. 1970.

[12] L. Weiss, I. Paz, D.L. Schilling,"Overshoot Suppression in Adaptive Delta Modulator Links for Video Transmission," Proc. of the N.T.C. 73, Atlanta, Ga., vol.1, pp. 6D1 to 6D6, Nov. 1974.

[13] M. Oliver,"An asynchronous delta modulator with overshoot suppression for video signals," IEEE Trans. on Comm. pp.234-247, Mar. 1983.

[14] N.S. Jayant, S.W. Christensen,"Adaptive Aperture Coding for Speech Waveforms-1," Bell System Tech. Journal, vol.58, No.7, Sept. 1979.

[15] W.B. Davenport Jr.,"An experimental study of speech wave probability distributions," Journal of Acoustical Society of America, vol.24, pp.390-399, July 1952.

[16] J.A. Scuilli,"A Statistical Characterization of Digital Voice Signals for Compression System Applications," Tech. Report COMSAT Lab.,1964.

[17] M.E. Binal,"On the Derivation of Certain Statistical Properties of Speech," IEEE Trans. on Comm., vol 26, pp.203-205, 1978.

[18] S. O. Rice, "Time-Series Analysis," John Wiley & Sons, Inc.,New York, 1963.

[19] G. R. Schwarz, "Buffer Design for Data Compression Systems," IEEE Trans. Comm. Tech., Com-16, pp. 606-615, AUG. 1968.

[20] D. N. Sherman,"Storage and delay estimates for asynchronous multiplexing of data in speech," IEEE Trans. on Comm. Tech., pp.551-555, August 1971.

[21] R. Ash, Information Theory, Interscience Publication, John Wiley & Sons, 1967.

[22] T. S., Huang,"An Upper Bound on the Entropy of Run Length Coding", IEEE Trans. on Info. Theory, pp.675-676, Sept. 1974.

[23] S. D. Bradley, "Optimizing a Scheme for Run Length Coding", Proceedings of IEEE, pp.108-109, Jan. 1969.

[24] J. I. Molinder, "Optimal Coding with a Single Standard Run Length, IEEE Trans. on Information and Theory, IT-20, pp.336-343, May 1974.

[25] G. R. Schwarz, "Buffer Design for Data Compression Systems", IEEE Trans. on Comm. Tech., COM-16, pp.606-615, Aug. 1968.

[26] P. H. Dosik, M. Schwartz,"An Optimized Buffer Controlled Data Compression Systems", IEEE Trans. on Comm., COM-22, pp.1506-1515. Oct. 1974.

[27] J. E. Medlin,"The Prevention of Transmission Buffer Overflow in Telemetery Data Compressors", IEEE Trans. on Comm. Tech., COM-16, pp.94-107, Feb. 1968.

[28] A. Buzo, A.H. Gray,Jr., R.M. Gray, J.D. Markel, "Speech coding based upon vector quantization," IEEE Trans. Acoust. Speech and Sig. Process, ASSP-28, pp. 562-574, Oct. 1980.

[29] S. Roucos, R. Schwartz, J. Marhoul, "Vector quantization for low rate coding of speech," Conf. Rec., 1982 Global Comm. Conf., pp. E6.2.1-E6.2.5, Miami, Fl, Nov.29-Dec.2, 1982.

[30] H. Abut, R.M. Gray, G. Rebolledo, "Vector Quantization of speech and speech-like waveforms," IEEE Trans. Acoust. Speech and Sig. Process, ASSP-30, pp. 423-436, June 1982.

[31] V. Cuperman and A. Gersho, "Adaptive differential vector coding of speech ," Conf. Rec., 1982 IEEE Global Comm. Conf., pp. E6.6.1-E6.6.5, Miami, Fl, Nov.29-Dec2, 1982.

[32] Not used.

[33] J. Max, "Quantizing for Minimum Distortion", IRE Trans. on Information Theory, no. 6, pp.169-174, March 1960.

[34] Y. Lind, A. Buzo, R. M. Gray, "An Algorithm for Vector Quantization", IEEE Trans. on COmmunications, vol. COM-28, pp.84-95, Jan 1980.

[35] Not used.

[36] G. h. Ball and D. J. Hall '"A Clustering Technique for Summarizing Multivariate Data",IEEE Trans. on Information Theory, vol.IT-19, pp.73-77, January 1973.

[37] R. M. Gray,"Vector Quantization", IEEE ASSP Magazine, pp.4-29, April 1984.

[38] J. H. Chen, A. Gersho," Gain-Adaptive Vector Quantization with Application to Speech Coding",IEEE Trans. on Comm., Vol. COM-35, Sept. 1987, pp 918-930.

[39] V. Cuperman, A. Gersho,"Vector Predictive Coding of Speech at 16 kbits/s," IEEE Trans. Comm.,vol. COM-33, pp.685-696.

[40] P. Chang, R. M. Gray, J. May,"Fourier Transform Vector Quantization for Speech Coding", IEEE Trans. Comm., vol. COM-35, No. 10, October 1987, pp.1059-1068.

[41] L. R. Rabiner, R. W. Schafer, Digital Processing of Speech Signals, Prentice Hall, 1978.

[42] N.S. Jayant, P. Noll, Digital Coding of Waveforms, Principles and Applications to Speech and Video, Prentice Hall, 1984.