Context Dependent Prediction and Category Encoding for DPCM Image Compression

Paul R. Beaudet
Westinghouse Electric Corporation
Advanced Technology Division
Baltimore, MD

ABSTRACT

Efficient compression of image data requires the understanding of the noise characteristics of sensors as well as the redundancy expected in imagery. Herein, the techniques of Differential Pulse Code Modulation[1](DPCM) are reviewed and modified for information-preserving data compression.

The modifications include:
- o Mapping from intensity to an equal variance space
- o Context dependent one and two dimensional predictors
- o Rationale for nonlinear DPCM encoding based upon an image quality model
- o Context dependent variable length encoding of 2x2 data blocks
- o Feedback control for constant output rate systems

Examples are presented at compression rates between 1.3 and 2.8 bits per pixel. The need for larger block sizes, 2D context dependent predictors, and the hope for sub-bits-per-pixel compression which maintains spacial resolution (information preserving) are discussed also.

Introduction

This paper discusses an image data compression technique which has shown great promise in studies performed at Westinghouse. The technique incorporates "context dependent" and "category encoding" methods. The major steps of the process comprise:

1. equal variance mapping of the input data
2. context dependent prediction of the current pixel value
3. nonlinear differential pulse code modulation (DPCM) of the pixel differences
4. variable length encoding of blocks of the DPCM deltas
5. error correction encoding to protect against transmission and storage bit errors.

Ref. 1. Azriel Rosenfeld and Avinash C. Kak, Digital Picture Processing, Chapter 5, Academic Press, Inc., 1982

A block diagram of the basic concept is given in Figure 1. The input data is passed through a nonlinear function selected to make the noise variance of the signal independent of the absolute signal level. The exact shape of this function is tailored to compensate for signal dependent noises; at low signal levels the predominant noise is system noise, while at higher levels shot noise or scene noise predominate. The equal variance mapping technique renders the DPCM process equally effective at all signal levels.

The context dependent predictor (CDP) applies a set of coefficients to neighboring pixels (casual process) to predict the next pixel intensity. The choice of coefficients depends upon the pattern classification assigned to the neighborhood. The CDP can be either one or two dimensional. The predicted pixel is subtracted from the actual pixel data to obtain a "delta".

DPCM is the third step in the data compression process. A graded, symmetrical table of delta is used. The DPCM code keeps the noise of the delta code selection an approximately constant percentage of the actual signal gradient, and produces an output data set whose probability density function is sharply peaked. This latter property makes the output deltas particularly suited to variable-length encoding techniques such as Huffman encoding.

The next step is the variable-length encoding of 2x2 data blocks. The data is encoded in blocks of two by two pixels for encoding efficiency. Variable-length encoding assigns the output Huffman code to the 2x2 blocks of deltas in such fashion that the shortest groups represent the most frequently used delta blocks, and the longest groups represent the most infrequently used delta blocks. This minimizes the number of bits in the output data stream.

Because the data compression process removes almost all redundancy from the signal, an error in signal transmission can cause a large loss of data. This makes it necessary to follow the data compression process with an error-correcting encoding process. This process reintroduces a small amount of signal redundancy with high efficiency, so that an encoding overhead of only one or two percent suffices to reduce bit loss to an acceptably low level.

Equal Variance Mapping

The equal variance mapping function is derived from a noise model.

The noise model used assumed three independent noise sources:

1. electronic noise
2. shot noise
3. scene noise

Scene noise can be viewed as a fluctuation in scene reflectivity which is not considered as containing any significant information. The purpose of the equal-variance mapping function is to map the input intensity I into a new variable X such that the noise, σ_x, in this new variable is independent of the absolute level X. Figure 2 illustrates this property of the function.

The noise model expresses the input intensity variance,

$$\sigma_I^2 = \underbrace{\sigma_o^2}_{\text{electronic noise}} + \underbrace{\Gamma I}_{\text{shot noise}} + \underbrace{(ßI)^2}_{\text{scene noise}}$$

In this model, $N = I/\Gamma$, measures the input signal in electrons. In terms of N,

$$\sigma_N^2 = \left(\frac{\sigma_o}{\Gamma}\right)^2 + N + (ßN)^2$$

σ_o/Γ is the number of electrons of electronic noise. Some sensor electrometers have reduced this noise component down to seven electrons. More realistically, electronic noise of hundreds of electrons might be expected. $\sqrt{N}$ is the shot noise due to the random nature of photon emission. A fraction, ß, of the scene signal is considered texture or scene noise. Typically, $ß \simeq .01$ is used. The equal variance mapping function satisfies the differential equation

$$\sigma_x = \sigma_I \frac{dx}{dI}$$

which has solution

$$X(I) = \Phi \frac{\sigma_o}{ß} \log \left[\frac{\left(\sqrt{\sigma_o^2 + \Gamma I + ß^2 I^2} + ßI + \frac{\Gamma}{2ß} \right)}{\left(\sigma_o + \frac{\Gamma}{2ß} \right)} \right]$$

Φ is selected so that $X(I_{MAX}) = X_{MAX}$. We often select $X_{MAX} = 4095$ so as to optimize the dynamic range of a 12 bit processor.

Context Dependent Predictor

One dimensional predictors are used mostly because two dimensional ones require sensor equalization which makes all of the detectors respond similarly to input intensity; many systems cannot afford the luxury of data equalization. A context-free linear predictor is of the form

$$X_{i+1} = X_i + \in (X_i - X_{i-1})$$

where $\in$ is a constant ($\in = .75$ is sometimes used). A context-dependent predictor has the same "form" of this equation, but $\in(x_i - X_{i-1})$ is interpreted as "a function of" $(X_i - X_{i-1})$. This function is selected to maximize the prediction accuracy and can

be determined using scene statistics. The results suggest that an equivalent multiplier, ϵ, should be negative for very small differences, $\epsilon \sim -1$ for intermediate but significant gradients and $\epsilon \sim .7$ for very large gradients. Intuitive rationale supports these findings; in uniform regions, small differences should be smoothed (negative ϵ), and $\epsilon = 1$ near large edge discontinuities would overshoot the edge.

Two dimensional predictors are currently being inserted into the context dependent DPCM image compression software. Also, the encoding block is being increased from 2x2 to 4x4 pixel blocks. The two dimensional predictor uses data from a causal neighborhood as shown in Figure 3. This data set is used to characterize the neighborhood in terms of a context class index, k. The set of prediction coefficients used is dependent upon k but is otherwise linear:

$$\tilde{X} = \sum_{i,j \in C_4} \alpha^{(k)}_{ij} \; X_{ij}$$

The DPCM Processor

The block diagram of the DPCM processor is given in Figure 4. The input to the process is the output of the equal-variance mapper. The predictor functions in this mapped space, attempting to predict the next pixel value as accurately as possible. The better the predictor, the narrower will be the density function of the resulting deltas. The processor includes gain in the forward and reverse paths so that a *dynamic range adjustment* (DRA) can be made by the feedback parameter Q. This controls the output bit rate and ensures optimum delta encoding of the data in the transmission mode. The delta values are selected from a table of possible deltas which has been derived by consideration of data noise and rate distortion. This consideration leads to a hyperbolic sine relationship between the actual delta difference and the delta code. A typical delta table is given in Table 1.

Variable-Length Encoding with Blocking

The distribution of δ-code values is peaked near zero. If P_i is the probability of the $i^{th}\delta$-code (relative frequency), then a variable length code (Huffman code) having about $b_i \approx -\log_2 P_i$ bits for the i^{th} δ-code state minimizes the number of bits required to encode the image values. Without encoding, the bits per pixel might be 5 since δ-codes range over $-15 \leq +15$; (5 bits per pixel (bpp)). Because P_o is typically .7 or so, only 0.5 bits are allocated to that state. As a consequence, an encoding inefficiency is incurred whenever a single state occurs with such high probability. To avoid this encoding inefficiency, 2X2 blocks are Huffman encoded with 2 bits $\simeq -\log_2(.7)^4$ assigned to blocks consisting of all zero δ-code states.

To provide a structure which can be continuously processed, the 2x2 blocks are arranged as shown in Figure 5. To further improve the encoding, contextual information from neighboring blocks is used to create an adaptive Huffman encoding scheme. The sign bits from the delta codes of the pixel on either side of the block, taken together with two bits from the adjacent top pixels are used to define 16 context states. Statistics have been accumulated for all of these states and a different Huffman encoding table is used for each context.

Category Encoding Method

It is impractical to assign variable length codes to each of the possible 2x2 block states. Since 5 bit DPCM codes are used, this would require storage for $2^{4x5} = 2^{20}$ code words for each of 16 contextual situations. To avoid such a massive table, category (cluster) encoding is used. Many of the 2^{20} states which are clustered into separate categories include only one of the block states while other categories include many of these states. For those categories containing many states, an additional resolution code must be supplied in order to identify the specific state of the cluster.

In the schema shown in the Figure 6, a 3-bit pixel category (sub-category) is assigned. The eight states correspond to δ-code values +0, $\pm$1, $\pm$2, $\pm$(3, 4, 5, 6 or 7) and $|\delta\text{-code}| \geq 8$. The 3-bit pixel categories are used to define a 5-bit α, ß pair category (1x2 category) according to the strategic table shown. Note that if both α and ß pixels have δ-code values between $\pm$2, the 1x2 category code completely determines the state of both pixels; no resolution code need be appended to the category. When the two upper and lower α,ß pair 1x2 category codes are brought together, each pair contributes five bits to a variable length 2x2 category code table. These 10 bits plus the four context bits constitute an address for a single 16K ROM containing the complete adaptive Huffman encoding tables. This is a thousand times less storage than the direct approach!

Error-Correction Overhead

Error correction encoding processes operate by the insertion of error correction bits into blocks of data before subjecting the data to a noisy process such as storage or transmission. After the data is transmitted (or read from storage) a mathematical operation on the combined block of data plus error correction bits reveals the presence and location of any bits in error, up to a limit determined by the number of correction bits and the size of the data block. The quality of the code is measured by the maximum number of bits in error, L, that can without fail be corrected by the code within a fixed block. The overhead ratio M/K of this process is the number of error correction bits M divided by the number of bits K in the original data block. This ratio is a function of the ratio L/K. For a fixed value of this latter ratio, the bit error rate improves with the block size K in a complex manner. An analysis of this problem was made assuming an output bit error rate of 10^{-15} with an input bit error probability of 10^{-5} and an overhead ratio of just over 1 percent (1.14%) was required. Under these assumptions, calculations showed that, with a block size of 5000, any 5-bit error could be corrected. This is far better performance than needed for any perceived application.

Compression Performance

The performance of this data compression technique is dependent on the statistics of the input images. Our experience to date has been with scenes taken in the visible spectrum. With this type of input, compression from 2 to 2.5 bits per pixel can be achieved with a noise increase of no more than 3 dB. Images compressed to 1.3 bpp show no loss of information content. Figure 7(b-d) is the compressed data of a scene at (1.4, 1.7, and 2.1 bpp). There is little perceptual difference between this compressed image and the original, even at 1.3 bpp. Figure 8 (b-d) shows another image compressed at 1.5, 2.0, and 2.4 bpp. The two images were courtesy of the National Institute of Health, Bethesda, Maryland.

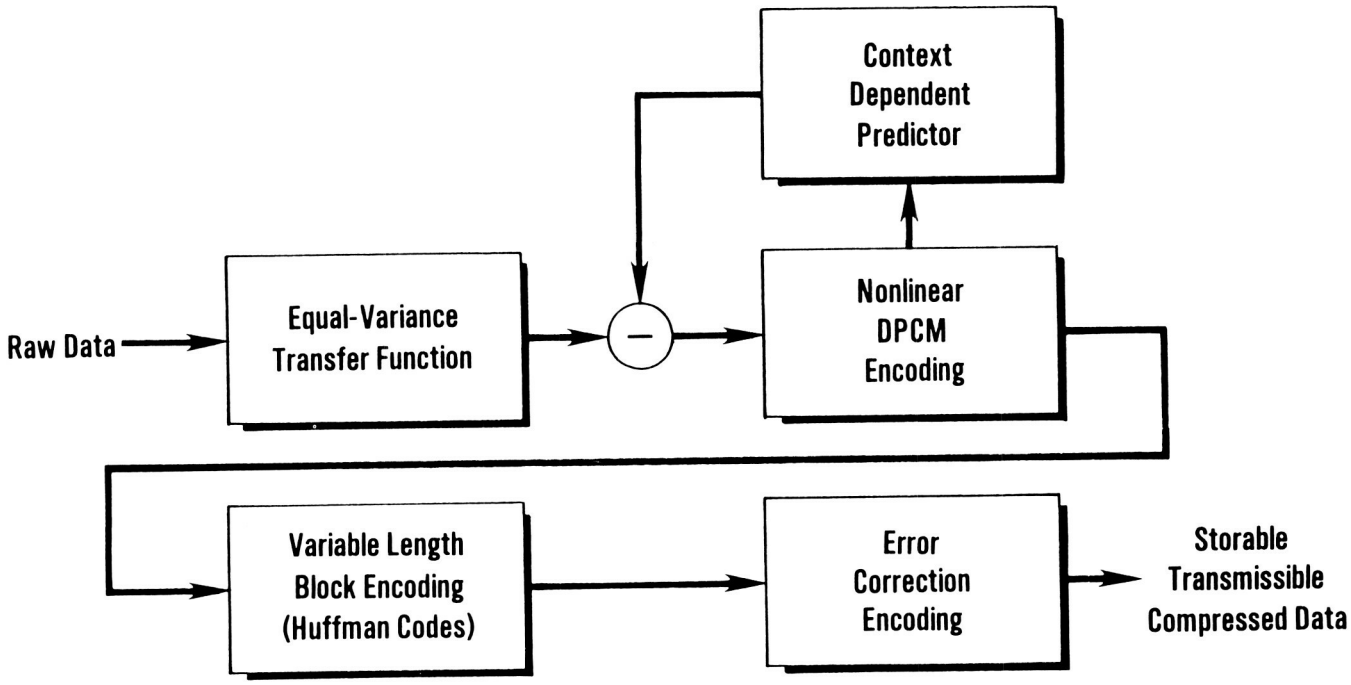


Figure 1. Data Compression Block Diagram

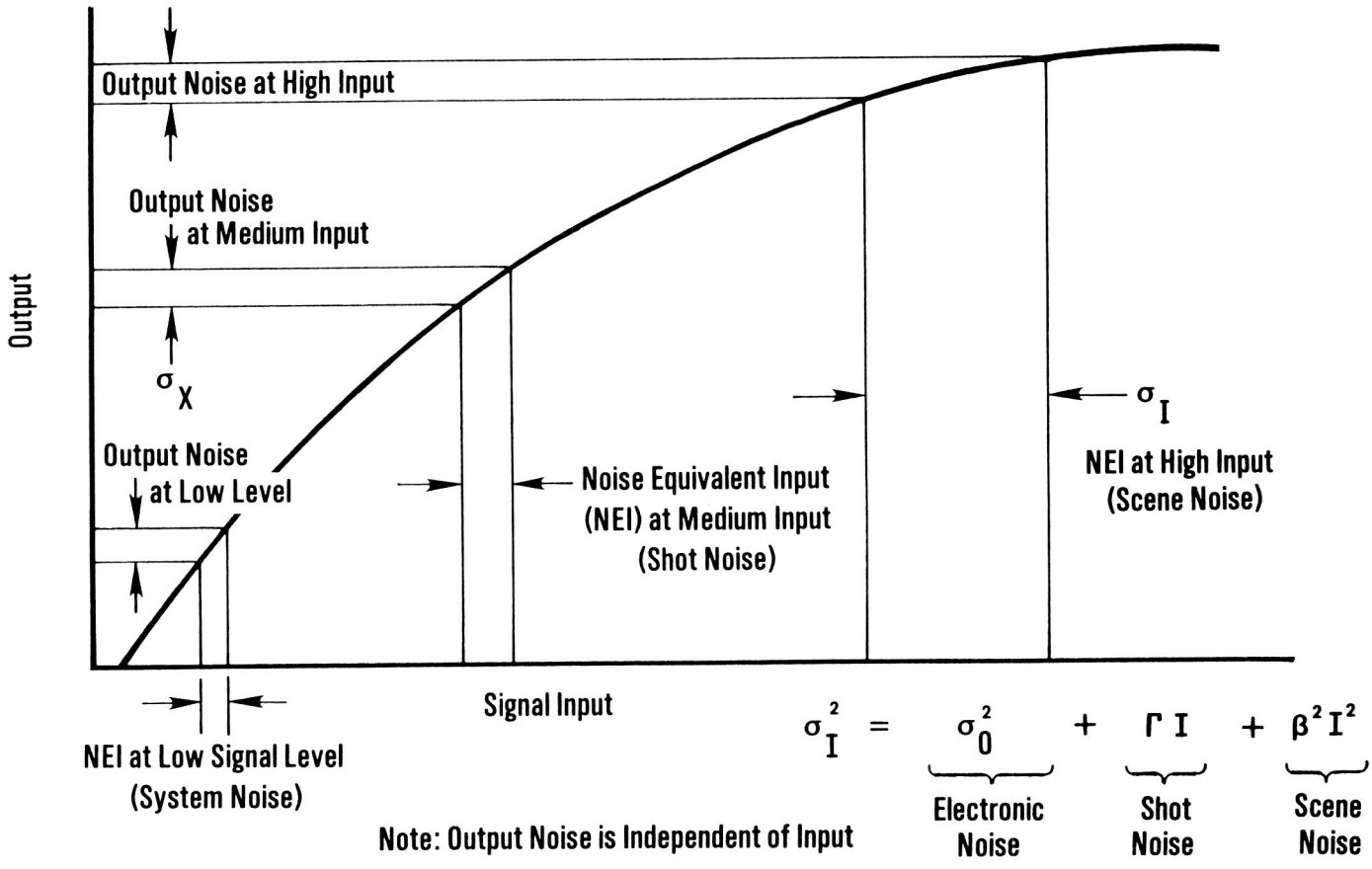
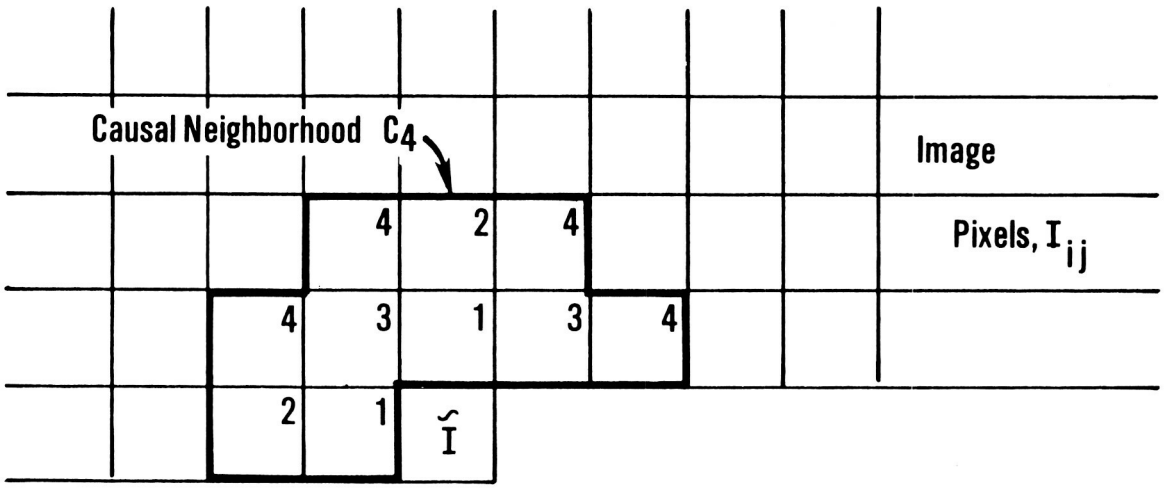


Figure 2. Equal Variance Transfer Function



- Context Index "K" is Determined By Image Gradient
- Context Dependent Predictor

$$I = \sum_{i,j}^{(K)} a \; I_{ij}$$

$$i,j \; \varepsilon \; C_4$$

Figure 3. Context Dependent Bandwidth Compression

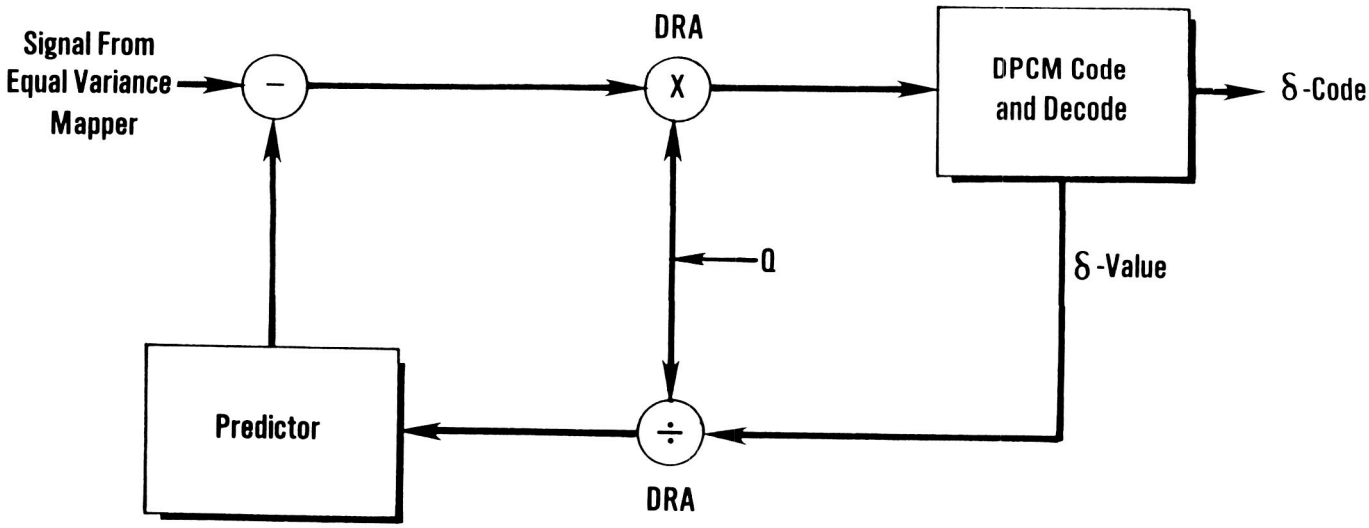


Figure 4. Block Diagram of DPCM Processor

Table 1. 5-Bit DPCM Table

δ-Code State	δ_x -Range	No. of x-States	δ Value
0	-3 – -3	7	0
±1	±(4 – 9)	6	±6
±2	10 – 17	7	13
±3	18 – 28	10	21
±4	29 – 42	14	33
±5	43 – 62	20	50
±6	63 – 91	29	73
±7	92 – 132	41	106
±8	133 – 192	60	154
±9	193 – 278	86	223
±10	279 – 402	124	323
±11	403 – 582	180	467
±12	583 – 843	261	676
±13	844 – 1220	377	978
±14	1221 – 1765	545	1415
±15	1766 – ∞	–	2047

Blocking Geometry:

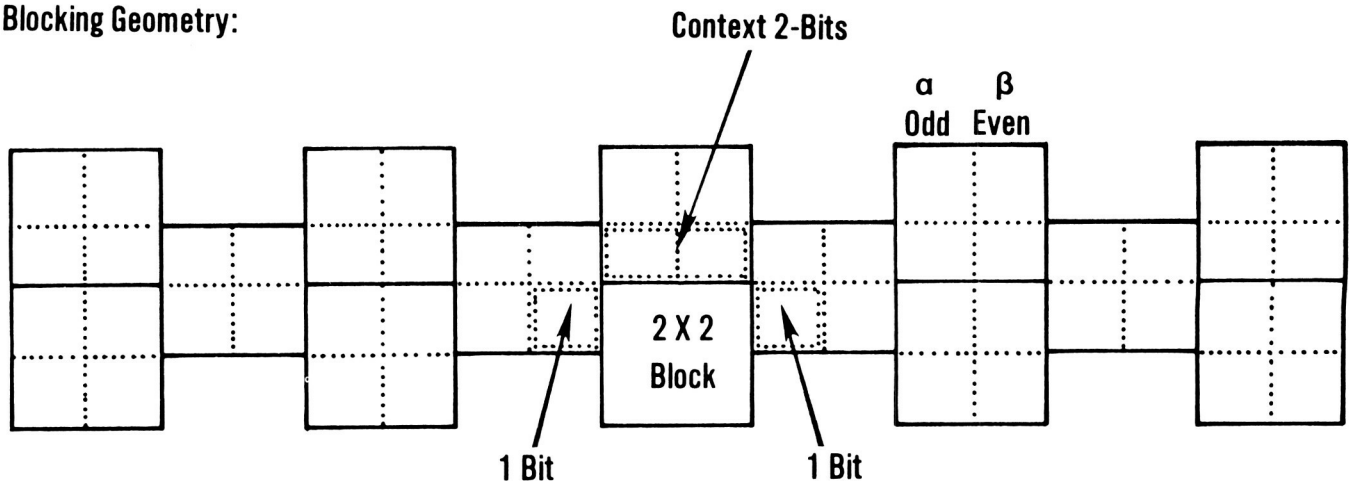


Figure 5. Variable Length Encoder 2 X 2 Blocking Concept

SUB-CATEGORIES

- 8 Sub-Categories Per Pixel
  - Unique Categories

δ-Code	Category Index	Resolution Required
±0	±0	None
±1	±1	None
±2	±2	None

  - Non-Unique Categories

±(3, 4, 5, 6, 7)	±3	1 of 5
(±8 I 9 I 10....±15)	-0	1 of 16

1 X 2 CATEGORY CODES ★

β \ α	-3	-2	-1	0	1	2	3	-0
-3	27	25	25	25	25	25	27	29
-2	25	24	20	21	22	23	26	28
-1	25	19	4	5	6	9	26	28
0	25	18	3	0	7	10	26	28
1	25	17	2	1	8	11	26	28
2	25	16	15	14	13	12	26	28
3	27	26	26	26	26	26	27	29
-0	30	28	28	28	28	28	30	31

α	β

★ MS Two Bits Are Stored for Context

Figure 6. Category and Resolution Concepts

Figure 7a. Plane Original.

Figure 7b. Plane at 2.1 BPP.

Figure 7c. Plane at 1.7 BPP.

Figure 7d. Plane at 1.4 BPP.

Figure 8a. Building Original.

Figure 8b. Building at 2.4 BPP.

Figure 8c. Building at 2.0 BPP.

Figure 8d. Building at 1.5 BPP.