12

EDGEWOOD
RESEARCH,
DEVELOPMENT &
ENGINEERING
CENTER

ERDEC-CR-059

IMAGE DETECTION USING EDGE EXTRACTION ALGORITHMS

Chein-I Chang

UNIVERSITY OF MARYLAND
Baltimore, MD 21228-5398

August 1993

U.S. ARMY
CHEMICAL
AND BIOLOGICAL
DEFENSE AGENCY

Aberdeen Proving Ground, Maryland 21010-5423

Disclaimer

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorizing documents.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	1993 August	Final, 91 Sep - 92 Aug

4. TITLE AND SUBTITLE

Image Detection Using Edge Extraction Algorithms

5. FUNDING NUMBERS

C-DAAA15-91-K-0002

6. AUTHOR(S)

Chang, Chein-I

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Maryland, Department of Electrical Engineering,
Baltimore, MD 21228-5398

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDEC-CR-059

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DIR, ERDEC,* ATTN: SCBRD-RTE, APG, MD 21010-5423

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

COR: Felix L. Reyes, SCBRD-RTE, (410) 671-2140

(Continued on page ii)

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In this report, we investigate alternative time-domain image detection techniques that do not require statistical models. Two distinct methods are presented; multistage pulse code modulation (MPCM) and relative entropy-based image thresholding technique. The idea of MPCM is to expand the commonly used Pulse Code Modulation (PCM) to multistage quantization to decompose an image into multiple stages. Each stage represents an image description with different details. By virtue of this decomposition, MPCM can be regarded as a progressive edge detection technique and a top-down gray-level triangle method, respectively. For vapor cloud detection, we are often interested in the detection of clouds. Without reconstructing the original image, progressive edge detection extracts edges of objects progressively from the most significant contours to the least important boundaries. The relative entropy-based approach adopts a different philosophy. It utilizes a co-occurrence matrix to efficiently segment objects from the background. It chooses an information theoretic criterion relative entropy as a thresholding technique which turns out to be a better approach than existing entropic thresholding methods in most test images.

14. SUBJECT TERMS

Detection Thermal image Vapor cloud Chemical imaging coding

15. NUMBER OF PAGES

74

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

11. SUPPLEMENTARY NOTES (Continued)

*When this study was conducted, ERDEC was known as the U.S. Army Chemical Research, Development and Engineering Center, and the Contracting Officer's Representative was assigned to the Detection Directorate.

PREFACE

The work described in this report was authorized under Contract No. DAAA15-91-K-0002. This work was started in September 1991 and completed in August 1992.

The use of trade names or manufacturers' names in this report does not constitute an official endorsement of any commercial products. This report may not be cited for purposes of advertisement.

This report has been approved for release to the public. Registered users should request additional copies from the Defense Technical Information Center; unregistered users should direct such requests to the National Technical Information Service.

Blank

CONTENTS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The use of passive thermal sensors for detecting and tracking chemical vapor offers the advantage of covertness, great mapping capability, and a smaller, simpler system design than active detection using lidar. Unfortunately, these advantages are often outweighed by the difficulty in observing low-concentration clouds against natural backgrounds. Obvious techniques for improving target contrast such as frame-to-frame subtraction or thresholding are useful if the background spatial fluctuations are (1) highly correlated in time between successive frames or (2) considerably less than the difference between the target and mean background radiance.

The Low Contrast Detection Algorithm (LCDA) currently being used by CRDEC derives a sequential likelihood ratio detector based on a model developed by Warren in [1,2] for vapor cloud. Since it was shown by Brilliger that Fourier transforming a wide-stationary time series produced a set of nearly uncorrelated Gaussian random variables, the clutter could be statistically approximated by a normal distribution and whitened by mapping the data into spatial Fourier domain. This approach offers improved detection and location performance for imaging sensors under a much broader class of background conditions than the ad hoc methods currently in use. Based on the well-established likelihood rational test, Warren was able to improve detection algorithms by making optimal use of both the spatial and temporal statistics of the background to enhance target contrast prior to the detection decision.

Despite the success of LCDA in improving low contrast of targets, it requires a parametric model to statistically describe vapor clouds. It is often the case, however, that a statistical model may not adequately fit the behavior of a vapor cloud since its characteristics vary with time. Consequently, such a model-based approach may not yield the desired performance. In this report, we present two new time-domain methods, Multistage Pulse Code Modulation (MPCM) and relative entropy-based image thresholding both of which do not require statistical models of vapor clouds like Warren's approach.

MPCM was developed for image coding at the University of Maryland Baltimore County (UMBC) in early 1991. The idea of MPCM is to expand the commonly used Pulse Code Modulation (PCM) to multistage quantization which uses PCM to decompose an image into image components in multiple stages. Basically, MPCM is a progressive data compression technique which use predictive coding to decompose an image stage-by-stage, then recovers the original image progressively by adding all image components in different stages. Since the image components generated by the decomposition process represent different details of the signal, MPCM can be regarded as a multiresolution technique. The original image can be approximated by first processing the image component in the first stage, then progressively moving to higher stages until a desired image resolution is achieved. The resolution of the signal is refined by continuously processing more image components in higher stages. A complete treatment of the theory of MPCM is detailed in Chapter 2.

In Chapter 3, it is found that during the course of image reconstruction, edges of objects can be extracted from the background progressively. The target detection is then accomplished by detecting the contours and boundaries of potential objects. Each stage represents image description with different details so that the image component in the first stage retains the most significant details of the image and the least important information of the image is stored in the last stage.

By virtue of this decomposition, we can apply different image preprocessing techniques such as image enhancement to contrast the objects of interest appearing in a certain stage or filter image components to remove unnecessary details such as noise. After preprocessing, the original image can be reconstructed by summing all preprocessed image components. As a result, the reconstructed image generally enhances desired objects with less noise. Due to this capability, MPCM can be also viewed as a progressive edge detection technique. For vapor cloud detection, we are often interested in detection of clouds. In this case, progressive edge detection provides a cost-effective technique. Without reconstructing the original image, progressive edge detection extracts edges of objects progressively from the most significant contours to the least important boundaries. One of chief advantages is that the process is progressive and can be terminated at any stage as long as the edges of the desired objects are detected and extracted. Another advantage of MPCM is easy hardware implementation since its coding structure is relatively simple for circuit design. This is very crucial in real-time processing.

It is also interesting to note that MPCM can be viewed as a top-down gray-level triangle method opposed to commonly used bottom-up pyramid methods. By introducing a top-down gray-level triangle structure, MPCM performs both decomposition and reconstruction of images from top to bottom according to a pre-assigned top-down gray-level triangle. In contrast, a bottom-up pyramid method decomposes an image from bottom to top by a pre-selected blurring window; however, it reconstructs the image from top to bottom by interpolation. In Chapter 4, a detailed comparison is investigated between MPCM and a popular pyramid method developed by Burt and Adelson.

The relative entropy-based image thresholding method was developed in late 1991 and uses an information theoretic concept, relative entropy as a thresholding criterion to efficiently segment potential objects from the background. Applying Shannon's entropy concept to image thresholding is not new. Several studies have been reported over the past years. All existing entropic methods use a co-occurrence matrix to define second-order Shannon's entropy. A co-occurrence matrix is a transition matrix generated by changes in pixel intensities. For any two arbitrary gray levels i and j (i, j are not necessarily distinct), the co-occurrence matrix describes all intensity transitions from gray level i to gray level j. Two definitions are of interest, local entropy and joint entropy. Suppose that t is the desired threshold. t then classifies all pixels in an image into either background or objects where the background contains pixels with gray levels below or equal to t and the objects have gray levels above t. The two-region classification further divides the co-occurrence matrix into four quadrants which correspond respectively to transitions from background to background (BB), background to object (BO), object to b ckground (OB) and object to object (OO). The local entropy is defined only on two quadrants, BB and OO, whereas the joint entropy is defined only on the other two quadrants, BO and OB. Based on these two definitions, two algorithms were developed, each of which maximizes local entropy and joint entropy respectively.

The relative entropy-based method given in Chapter 5 is a new approach. Rather than maximizing entropies of background or object separately as mentioned above, we introduce the concept of relative entropy (or cross entropy, Kullback-Leiber's discrimination information and directed divergence) to measure discrepancy between two probability distributions. Since an image can be completely described by its probability distribution, the relative entropy can be interpreted as a measure of the distance between two images. To implement the relative entropy approach, one calculates the gray-level transition probability distributions of the co-occurrence matrices for an image to be segmented and a thresholded bilevel image, then determine which threshold minimizes

the discrepancy between these two transition probability distributions, i.e., their relative entropy. The threshold which has the smallest relative entropy will be used to segment the original image. As a result, the thresholded bilevel image will be the best approximation of the original image. Because transitions of OB and BO generally represent edge changes in boundaries and transitions of BB and OO indicate local changes in regions, a thresholded bilevel image resulting from relative entropy thresholding must have the best transition match to that of the co-occurrence matrix of the original image.

In this report, we present the work which had been done during the contract period of 9/1991-8/1992. The results are categorized into five chapters. Chapter 2 deals with the theory of MPCM. Chapters 3 and 4 are two applications of MPCM by viewing MPCM as a progressive edge detection and a top-down gray-level triangle method respectively. Chapter 5 is devoted to the relative entropy-based approach. Chapter 6 is made of papers which were presented in various conferences under the support of this contract.

In Chapter 2, we describe the theory of MPCM in detail, particularly, its encoding and decoding algorithms, and apply MPCM to chemical vapor cloud detection. In Section 2.1, a brief description of motivation of MPCM is given followed by Section 2.2 on the structure of 1-D MPCM, Section 2.3 on 1-D MPCM encoding and decoding algorithms with extension to 2-D MPCM and Section 2.4 on experiments. In Chapter 3, MPCM is suggested to be used as a progressive edge detection technique in Section 3.1. In order to demonstrate effectiveness and efficiency of MPCM, MPCM is implemented progressively to achieve image decomposition, edge detection and reconstruction. Several experiments are presented in Section 3.2, particularly, chemical vapor cloud detection where the experiment was conducted based on the data provided by CRDEC. Viewing MPCM as a *top-down gray-level triangle method*, an alternative view of MPCM is given in Chapter 4. Using this interpretation, MPCM is compared to commonly used pyramid coding methods. Section 4.1 introduces the concept of *top-down gray-level triangle* and Section 4.2 describes Burt-Adelson's bottom-up pyramid method. A detailed comparison is treated in Section 4.3 with experimental results given in Section 4.4. In Chapter 5, the relative entropy-based approach is described. Section 5.1 gives the design rationale of relative entropy followed by Section 5.2 on a review of previous work on entropic thresholding methods. In Section 5.3, the proposed relative entropy-based approach is described in detail. Section 5.4 investigates different experiments to analyze the relative entropy-based method and compare the performance to that using local entropy-based and joint entropy-based methods. Finally, Chapter 6 includes a brief conclusion and copies of papers which were presented in conferences throughout this contract support.

References

1. R. Warren, *Algorithm for Image Enhancement and Detection of Chemical Vapor Using a Thermal Imager,* Final Report, US Army Chemical Research, Development and Engineering Center, Aberdeen Proving Ground, MD 1987.

2. L. Carr, D. McPherrin and R. Wrarren, *Low Contrast Detection Algorithm,* SRI Technical Report, US Army Chemical Research, Development and Engineering Center, Aberdeen Proving Ground, MD, January 1990.

CHAPTER 2

MULTISTAGE PULSE CODE MODULATION (MPCM)

2.1 Motivation of MPCM

Multiresolution analysis has received considerable attention recently in image processing, particularly, computer vision [1]. The purpose of multiresolution analysis is to decompose an image of interest into a set of appropriately localized image components where each image component represents different image pattern information. Since the details of an image generally require characteristics of the physical structures of local scenes with different resolutions, analysis from coarse resolution to fine resolution provides a tool to localize or zoom in on significant parts of the signal in time as well as in frequency, thus it is very useful for pattern recognition. Therefore, there has been great deal of interest in developing multiresolution techniques for image representation.

PCM and DPCM are well-known coding schemes in converting analog signals to digital signals without and with predictors. Unfortunately, both PCM and DPCM are not multiresolution techniques. The motivation behind MPCM is inspired by a desire of making PCM and DPCM progressive and possess a multiresolution property. The idea is to expand single-stage PCM and DPCM into multistage PCM and DPCM in such a way that an input signal is quantized one level at a time rather than all levels at a time as is the case do in PCM and DPCM. As a result of multistage quantization, each stage represents a different signal resolution.

In essence, MPCM has a similar structure to that of multistage quantization [2]. However, the MPCM presented in this report is unique in three aspects. (1) While at each stage, MQ needs a new code book for the quantizer used in that stage, MPCM uses fixed quantization output levels. Therefore, using the Lloyd/LBG algorithm [2] to generate optimum sets of output levels for MQ is not necessary for MPCM. (2) The quantizers required in MQ are generally vector quantizers. It is however, not the case in MPCM where only scalar quantizers are used. (3) MPCM is relatively simple compared to MQ in terms of computational complexity and coding structure.

Although MPCM also possesses capability of progressive transmission, MPCM distinguishes itself from pyramid methods [3] in several important features. (1) Pyramid methods generally use window functions as filters such as Gaussian window functions to decimate images. MPC uses predictors to extract images. (2) Pyramid methods generate an image pyramid in which each level represents a coarse version of the original image with reduced-size. In this case, in order to recover the original image, a procedure of upsampling to expand reduced-size images to the same size of the original image is necessary. However, the image components generated by MPCM at each stage have the same size as the original image. (3) In an image pyramid produced by pyramid coding methods, the number of bits required to encode images at any layer is the same. This is not true for MPCM where the image components generated by MPCM require different bit rates which start with very low bit rates, e.g., one bit and gradually increase from a lower stage to a higher stage. More details will be given in Chapter 4.

2.2 Structure of MPCM

The idea of MPCM is to use a quantizer of two quantization levels and three threshold intervals coupled with a predictor to decorrelate input signals. More specifically, at each stage say stage k, the input to the k-th stage is the difference between $e_{k-1}(n)$ and $\hat{x}_k(n)$ (i.e., the kth prediction

error) where $e_{k-1}(n)$ is the $(k-1)$-th error signal resulting from the preceding stage and $\hat{x}_k(n)$ predicted at the k-th stage from the previous coded signal $x_k(n-1)$. We then use a two-level quantizer with three threshold intervals to quantize the k-th prediction error $(e_{k-1}(n) - \hat{x}_k(n))$. The designed quantizer is a soft-decision limiter with two output levels set to be the upper limit and lower limit respectively. As an input to the quantizer has value between these two limits, the quantizer simply passes it without changing its value. If the input exceeds the upper limit or falls below the lower limit, the input value will be tailored and replaced by either upper or lower limits.

Let M be the number of stages and $\{\Delta_k\}_{k=1}^{M}$ be a set of M quantization levels (or increments) representing stage levels, each of which is associated with a stage, e.g., Δ_k is the stage level of stage k. Basically, MPCM is a multistage version of PCM. Instead of quantizing inputs in a single stage as does PCM, MPCM quantizes inputs one level at a time where each quantization level represents one stage. As a result, M-level PCM can be expanded into M-stage MPCM and both also require the same number of bits, $(\log_2 M)$ for coding. It should be noted that an M-bit quantizer can generate 2^M output levels.

A general quantizer structure of an MPCM system described below is a multistage quantizer which divides the encoding task into successive stages, where the first stage performs a crude quantization using a small codebook. Then a second stage quantizer performs a quantization on the error data resulting from the difference between the original data and quantized data from the first stage quantization. A third stage quantizer further quantizes the second stage error data to refine error accuracy. Continuing this procedure results in a sequence of cascaded quantizers.

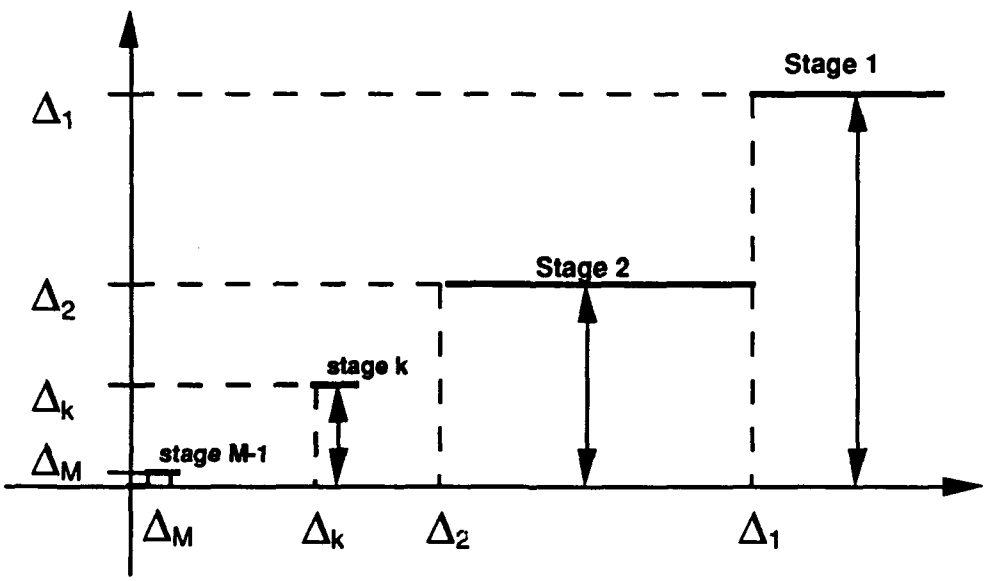


Figure 1: Quantizer Structure of M-stage MPCM

The quantizers used in the multistage quantizer for MPCM are depicted in Figure 1 where $\{\Delta_k\}$ are quantization levels used as stage levels and an image is quantized by one level at a time per stage. More precisely, the multistage quantizer used in MPCM is a sequence of cascaded quantizers Q_k and each Q_k is defined by two quantization levels $\{0, \Delta_k\}$ and three threshold-decision intervals $\{(-\infty, 0), [0, \Delta_k], (\Delta_k, \infty)\}$. If the input to Q_k falls off the range $[0, \Delta_k]$, it will replaced by either

Δ_k or 0 depending upon whether the input is positive or negative. Otherwise, a soft decision is made on the basis of the prediction from a predictor. While MPCM decomposes an image, it also generates a specially designed code, a priority code shown in Figure 2.

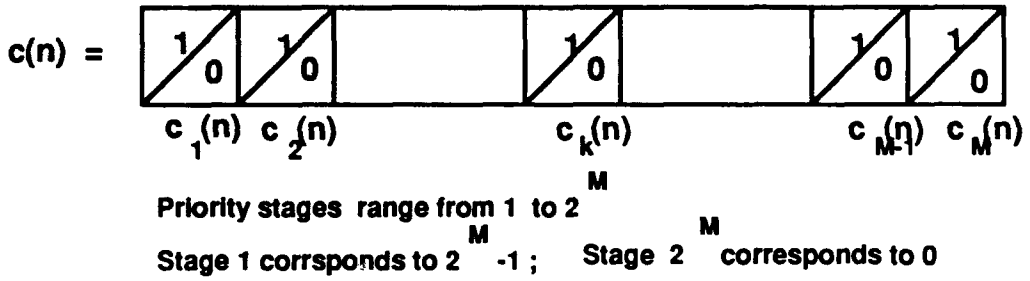


Figure 2: Priority Code for MPCM

The task of a priority code is to prioritize stages according to the importance of data information contained in stages. Assume that an MPCM system decomposes an image into M stages. The priority code assigns to a stage an integral number ranging from 0 to $M-1$. The higher number, the higher priority. The first stage representing the most significant information will be assigned by the largest integer $M-1$. This implies that the image component in stage 1 has the highest priority to be processed. On the Contrary, the last stage, stage M which contains the least important information will be assigned a 0. Therefore, it has the lowest priority. It should be noted that each priority determines a stage level, and vice versa. For example, a stage prioritized by k is stage $M-k$ with stage level Δ_{M-k}. Based on the above assignment, the higher the number assigned to the stage, the higher priority the stage. Since the priority code is the only information needed for encoding and decoding processes, the number of bits to represent the priority code is that required for an M-stage MPCM system.

In order to recover the original image, we use as a basis of reconstruction the coded images generated by the priority code. Namely, MPCM first decodes pixels whose stages have highest priority to form a coded image which represents the coarsest edge map of the image. If the image component predicted at this stage is added to the coded image, we then produce the coarsest version of the image. If the image quality is not satisfied, we continue to extract pixels of stages with the second highest priority so that a finer coded image can be generated from which a better reconstruction can be produced. This process is continued until the lowest priority is processed; the least significant edges are extracted and a complete reconstruction is then accomplished. During the course of producing coded images from higher priorities to lower priorities, image components at different stages can be added to reconstruct images to produce various resolutions of the image. As a result, making use of the priority code to process images yields lower-level to high-level computer vision analysis.

2.3 1-D MPCM

In order to illustrate the idea of MPCM, in what follows, we describe one-dimensional MPCM (1-D MPCM) in detail and give a brief description of 2-D MPCM in the following section.

Let M be the number of stages and $\{\Delta_k\}_{k=1}^{M}$ be a set of M strictly decreasing increments representing M stages, each of which is associated with one stage level, e.g., Δ_k at stage k. This gives rise to a $(\log_2 M)$-bit MPCM system with M stages and 2^M quantization levels. For instance,

if $M = 8$, a priority code c will be designed using $\log_2 8 = 3$ bits to represent 8 stages, e.g., 000 corresponding to stage 8, 001 to stage 7, $\cdots$, 111 to stage 1 such that the complement of the bit representation of the code c is the stage it represents. More precisely, the sum of the value of c and the stage specified by the c is fixed at 8 which is the total number of stages.

2.3.1 1-D M-Stage MPCM Encoding Algorithm (1-D MPCMEA)

The encoder of M-stage MPCM given below generates a priority code $c(n)$ and a set of M signal components $\{x_j(n)\}_{j=1}^{M}$. The general structure of the encoder of 3-stage MPCM is described in Figure 3.

1-D M-STAGE MPCMEA

A flow chart of 1-D M-stage MPCMEA is given in Figure 4.

1. Initialization:

   Let $x(0) = 0$ be the initial input signal and $c(n) = 0$. We initialize $x_k(0)$ by letting $x_j(0) = 0$ for all $j = 1, \cdots, M$. Let $x(n)$ be an input signal for $n = 1, 2, \cdots$.

   Set $e_0(n) = x(n)$.

   Set $k = 1$ and $c(n) = 0$.

2. If $k = M$, set $x_M(n) = 0$ and $c(n) = 0$ and **goto** step (4).

3. If $e_{k-1}(n) - x_k(n-1) \geq \Delta_k$, **then**

   (a) $x_k(n) = \Delta_k$, $x_j(n) = 0$ for all $k < j \leq M$ and $c(n) = M - k$

   (b) $e_M(n) = e_{k-1}(n) - x_k(n)$.

   Else,

   if $e_{k-1}(n) - x_k(n-1) < 0$, **then**

   (a) $x_k(n) = 0$, $x_j(n) = \Delta_j$
   for all $k < j \leq M$ and $c(n) = M - k$

   (b) $e_M(n) = e_{k-1}(n) - \sum_{j=k+1}^{M} x_k(n)$
   $= e_{k-1}(n) - \sum_{j=k+1}^{M} \Delta_j(n)$.

   else,

   (a) $x_k(n) = x_k(n-1)$ and $c(n) = 0$.

   (b) $e_k(n) = e_{k-1}(n) - x_k(n)$.

   (c) Set $k = k + 1$ and **goto** step (2).

4. Output all $x_j(n)$ for all $1 \leq j \leq M$ and $e_M(n)$ if necessary.

   Input the next signal $x(n+1)$. Set $n = n + 1$.

As a result, $x(n)$ can be decomposed into M signal components $\{x_j(n)\}_{j=1}^{M}$ and the reconstruction, $x'(n)$ given by the sum of all $x_j(n)$s obtained in M stages,

$$x'(n) = \sum_{j=1}^{M} x_j(n).$$

Three comments on MPCM are worth mentioning and given as follows.

- In step (3), according to the way the priority code is designed, the stage number represented by the value of $c(n)$, k is $M-k$ which is the complement of the bit representation of k. The sum of k and $M-k$ is always fixed at M. A block diagram of step (3) is given in Figure 5.
- Since the signal component in the last stage, i.e., stage M contains the least important information, the priority code representing stage M is value 0 which indicates stage M has the lowest priority for signal recovery. In this case, we intend to do nothing to achieve bit saving as depicted in step (5).
- Although MPCM is basically designed for lossy data compression, it can be made error-free by keeping tracking error correction terms $e_k(n)$. The extra number of bits required for such lossless compression is bounded by $\lceil \log_2 \Delta_1 \rceil$, the smallest integer equal to or larger than $\log_2 \Delta_1$.

2.3.2 1-D M-Stage MPCM Decoding Algorithm (1-D MPCMDA)

The following 1-D MPCM decoding algorithm is relatively simple because we only need the priority code $c(n)$ to reconstruct $x(n)$. As mentioned previously, a higher value of $c(n)$ corresponds to a higher priority. The general structure of the decoder of 3-stage MPCM is depicted in Figure 6.

1-D M-STAGE MPCMDA

Since $c(n)$ can take on integral values ranging from 0 to $M-1$, we assume $c(n) = k$ where $k \in \{0, 1, \cdots, M-1\}$ and $x'(n) = 0$ is the reconstruction of $x(n)$. A flow chart of 1-D MPCMDA is given in Figure 7.

1. Stage 1: $k = M-1$.

   If $x_1(n-1) = \Delta_1$, **then**

   (a) $x_1(n) = 0$ and $x_j(n) = \Delta_j$
   for all $1 < j \leq M$

   (b) $x'(n) = \sum_{j=1}^{M} x_j(n)$

   (c) **goto** step (3).

   Else, (in this case, $x_1(n-1) = 0$)

   (a) $x_1(n) = \Delta_1$ and $x_j(n) = 0$ for all $1 < j \leq M$

   (b) $x'(n) = x_1(n) = \Delta_1$

(c) **goto** step (3).

2. If $k = 0$, **goto** step (3). (i.e., last stage: stage M)

   Else,

   if $x_{(M-k)}(n-1) = \Delta_k$, **then**

   (a) $x_{(M-k)}(n) = 0$ and $x_j(n) = \Delta_j$ for all $M - k < j \leq M$

   (b) $x'(n) = \sum_{j=1}^{M-k-1} x_j(n-1) + \sum_{j=M-k}^{M} x_j(n) = \sum_{j=1}^{M-k-1} x_j(n-1) + \sum_{j=M-k+1}^{M} \Delta_j$.

   else,

   (a) $x_{(M-k)}(n) = \Delta_k$ and $x_j(n) = 0$ for all $M - k < j \leq M$

   (b) $x'(n) = \sum_{j=1}^{M-k-1} x_j(n-1) + x_{M-k}(n) = \sum_{j=1}^{M-k-1} x_j(n-1) + \Delta_k$.

3. Output $x'(n)$ and $x_j(n)$ for all $1 \leq j \leq M$.

It should be pointed out that if the value of $c(n)$ is k, it indicates the signal component in stage $M - k$ needs to be updated and the increment to be used is Δ_k.

2.3.3 2-D MPCM

The 2-D MPCM presented here is a straightforward extension of 1-D MPCM by encoding images line by line. It is also possible to use a zig-zag, horizontal or vertical scanning techniques to extend 1-D MPCM [4].

2.4 Experiments and Error Analysis

2.4.1 Examples of 1-D MPCM

In this section we apply 1-D MPCM to two examples. The first example is a 1-D waveform which is a scan line taken from a video signal. Unlike the first example which deals with a continuous signal, the second example uses discrete values to illustrate how 1-D MPCM works.

EXAMPLE 1

In this example we apply 3-bit MPCM to a 1-D waveform $x(n)$. Figure 8 shows the decomposition and reconstruction of 3-bit MPCM. As shown in Figure 8, MPCM has 7 stages corresponding to 7 stage levels $\{\Delta_k = 2^{8-k}\}_{k=1}^{7} = \{128, 64, 32, 16, 8, 4, 2\}$ which represent 7 coarse-to-fine signal resolutions where the last stage, stage 8 is of little interest and not shown in Figure 8 because the stage level is 0. As demonstrated, the first column shows the residual signals at different stages; the second column represents decomposed signals and the third column reproduces signals for signal recovery. The signal can be reconstructed by adding decomposed signals stage by stage.

EXAMPLE 2

In order to illustrate the ideas of 1-D MPCM, we apply 3-bit 1-D MPCM to two sets of numeric values $\{x(n)\}_{n=1}^{10}$. The first set data generates tables 1-2 and the second data produces tables 3-4. Tables 1 and 3 tabulate signal components at all stages (8 stages for 3-bit MPCM) and priority codes generated by encoders of MPCM. Tables 2 and 4 reconstructed by their corresponding decoders.

As noted in these tables, the values of $c(n)$ in most of cases are small which means that most stages are in low priority. This implies that the data is highly correlated.

2.4.2 Examples of 2-D MPCM

In this section, 2-D MPCM is applied to a digitized girl image with 256 gray levels. The results are compared to that produced by PCM and DPCM respectively.

EXAMPLE 3

The example provided here is to demonstrate relative performance between PCM, DPCM and MPCM. Experiments were conducted using different bit rates. Figure 9b and 9c are results of PCM and DPCM using 1, 2, and 3 bits. Figure 9a results from MPCM using 1, 2, and 3 bits with the maximum stage level $\Delta_1 = 128$ where 1-bit MPCM has two stage levels 128 and 0, 2-bit MPCM has four stage levels labelled by $\{\Delta_k = 2^{8-k}\}_{k=1}^{3} = \{128, 64, 32\}$ and 0; 3-bit MPCM has 8 stage levels represented by $\{\Delta_k = 2^{8-k}\}_{k=1}^{7} = \{128, 64, 32, 16, 8, 4, 2\}$ and 0. In Figures 9a-9c, the 3/4 of each of the entire images is produced by using 3 bits, the 3/16 image at the right top corner is done with 2 bits and the 1/16 image at the top right uses only one bit. It is worth noting that 1-bit MPCM is embedded in 2-bit MPCM in the sense that the signal components resulting from 1-bit MPCM can be generated by 2-bit MPCM at stage levels 0 and 128. 2-bit MPCM is embedded in 3-bit MPCM since the signal components produced by 2-bit MPCM can be also obtained by 3-bit MPCM at stage levels 128, 64, 32 and 16. As a result, MPCM possess multiresolution property as demonstrated in Figure 9 which shows that the image quality is gradually improved by increasing the number of stages. This salient difference distinguishes MPCM from PCM and DPCM where 2-bit DPCM can not be embedded in 3-bit DPCM nor 2-bit PCM in 3-bit PCM. A similar idea for such embeded quantization was treated in [5], but the approach is based on threshold alignment which is completely different from MPCM which uses prescribed thresholds (or increments).

2.4.3 Error Analysis

An error analysis of PCM, DPCM and MPCM with different bit rates is given by Figure 10 which is plotted based on entropy rate versus the mean-squared error measure (MSE), normalized MSE (NMSE) and signal-to-noise ratio (SNR) defined as follows.

$$\begin{aligned}
\text{MSE} &= \frac{1}{LN} \sum_{i=1,j=1}^{L,N} [(x(i,j) - \sum_{k=1}^{M} x_k(i,j))^2] \\
\text{NMSE} &= [\frac{\sum_{i=1,j=1}^{L,N} (x(i,j) - \sum_{k=1}^{M} x_k(i,j))^2}{\sum_{i=1,j=1}^{L,N} (x(i,j))^2}] \\
\text{SNR} &= -10 \log_{10} [\frac{\frac{1}{LN} \sum_{i=1,j=1}^{L,N} (x(i,j))^2}{\text{MSE}}] \quad (dB)
\end{aligned}$$

where $L = 200$, $N = 320$.

Table 5 tabulates MSE, NMSE and SNR produced by PCM, DPCM and MPCM used in the above experiments. Since for an image with 256 gray levels the best choice of the largest stage level for PCM and MPCM is 128, there are only three points in Figure 10 each of which corresponds to 1 bit, 2 bits and 3 bits respectively. Figure 11 shows error images of PCM, DPCM and MPCM with

different number of bits. Obviously, there is no comparison for PCM based on the same number of bits. As expected, DPCM achieves the higher SNR than does MPCM. However, this advantage is offset by incapability of progression and multiresolution.

References

1. A. Rosenfeld, ed., *Multiresolution Image Processing and Analysis*, Springer-Verlag, 1984.
2. R.M. Gray and A. Gersho, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.
3. P.J. Burt and E.H. Adelson, "Laplacian pyramid as a compact image code," *IEEE Trans. on Comm.*, vol. COM-31, Apr. 1983, pp. 532-540.
4. K.R. Rao and R. Yip, *Discrete Cosine Transform*, Academic Press, 1990.
5. K.H. Tzou, "Embedded Max quantizer," *ICASSP'86*, 1986, pp. 505-508.

Text continues on page 25.

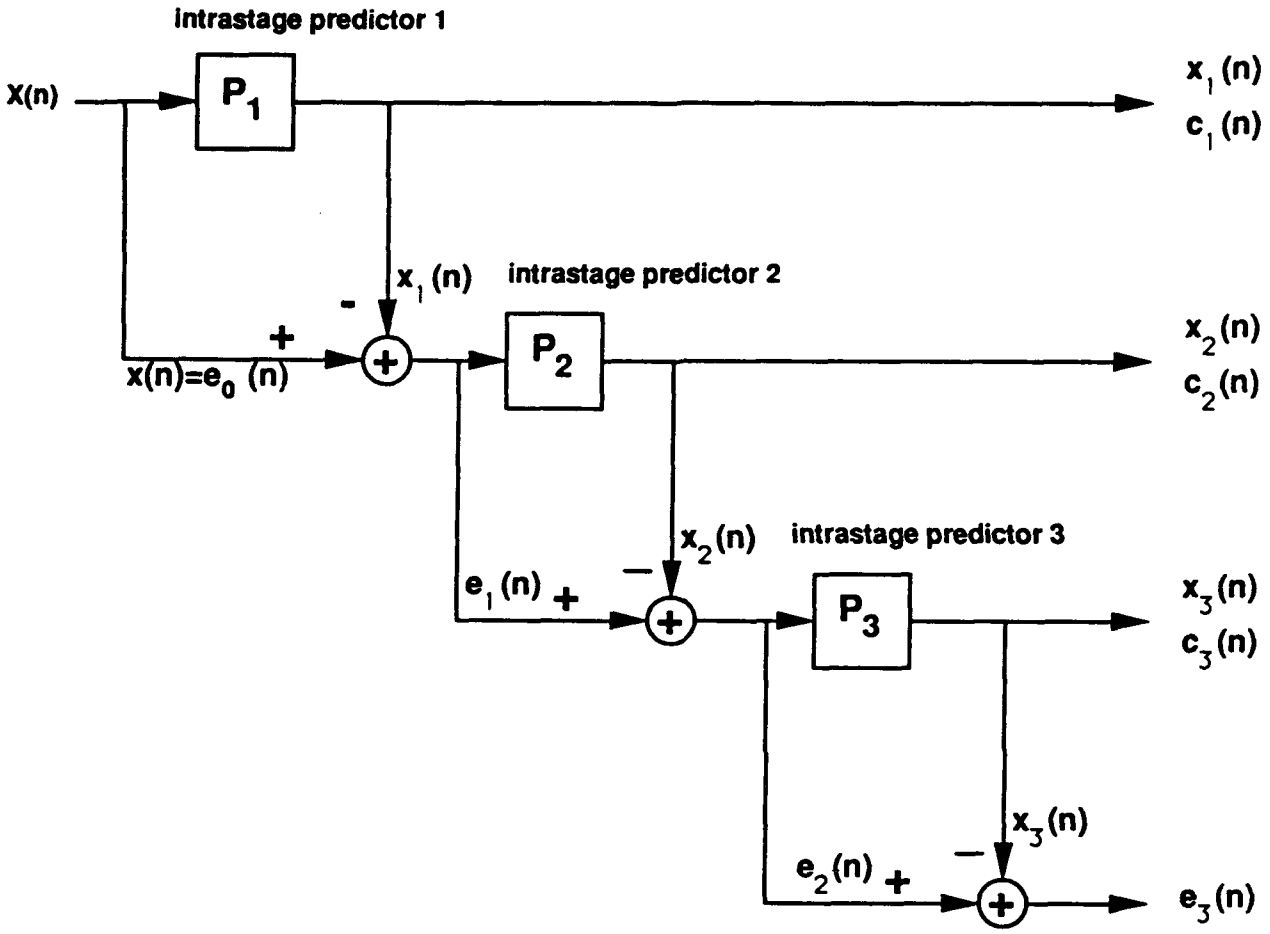


Figure 3: Encoder of 3-Stage MPCM

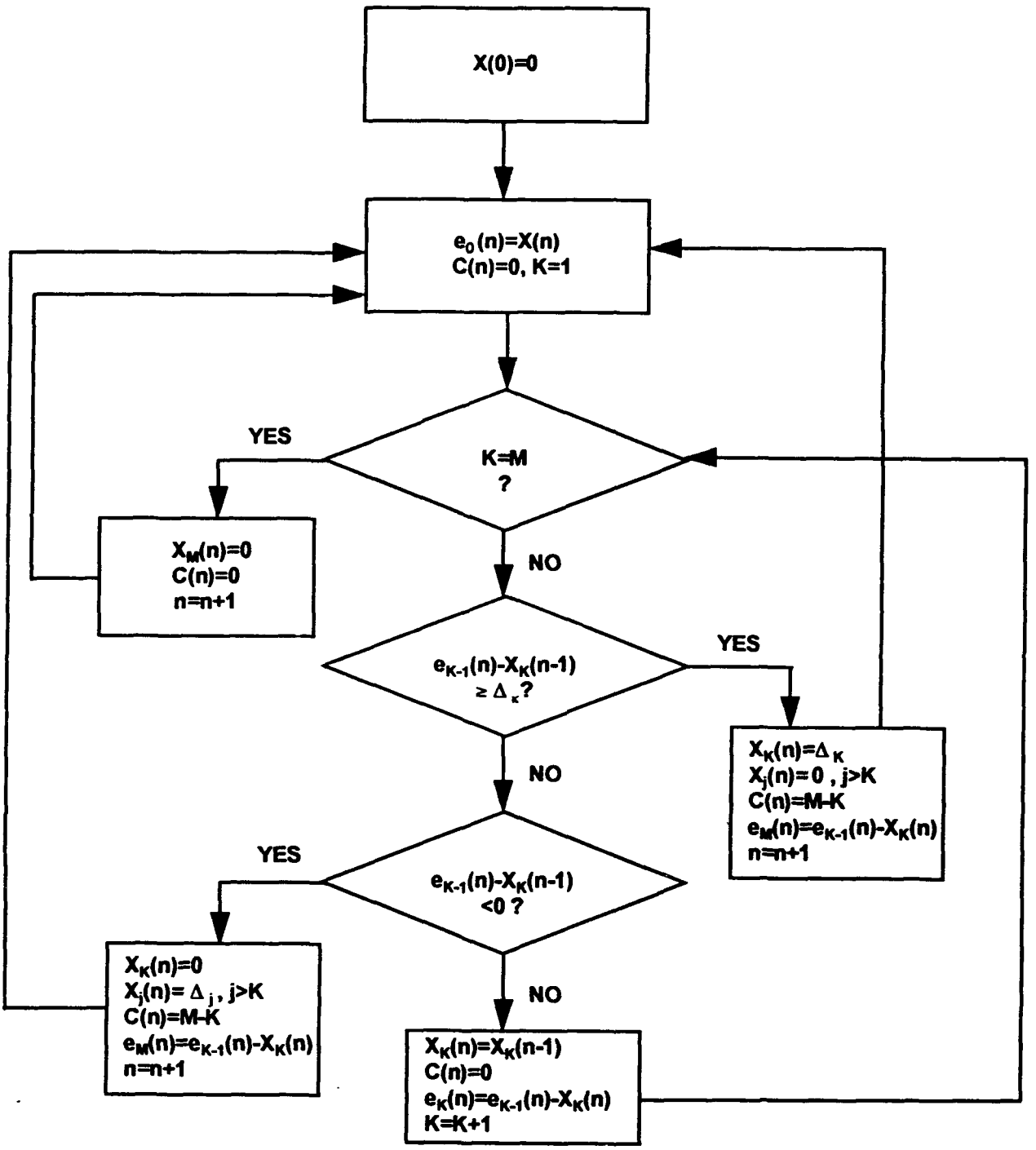


Figure 4: 1-D MPCM Encoding Algorithm

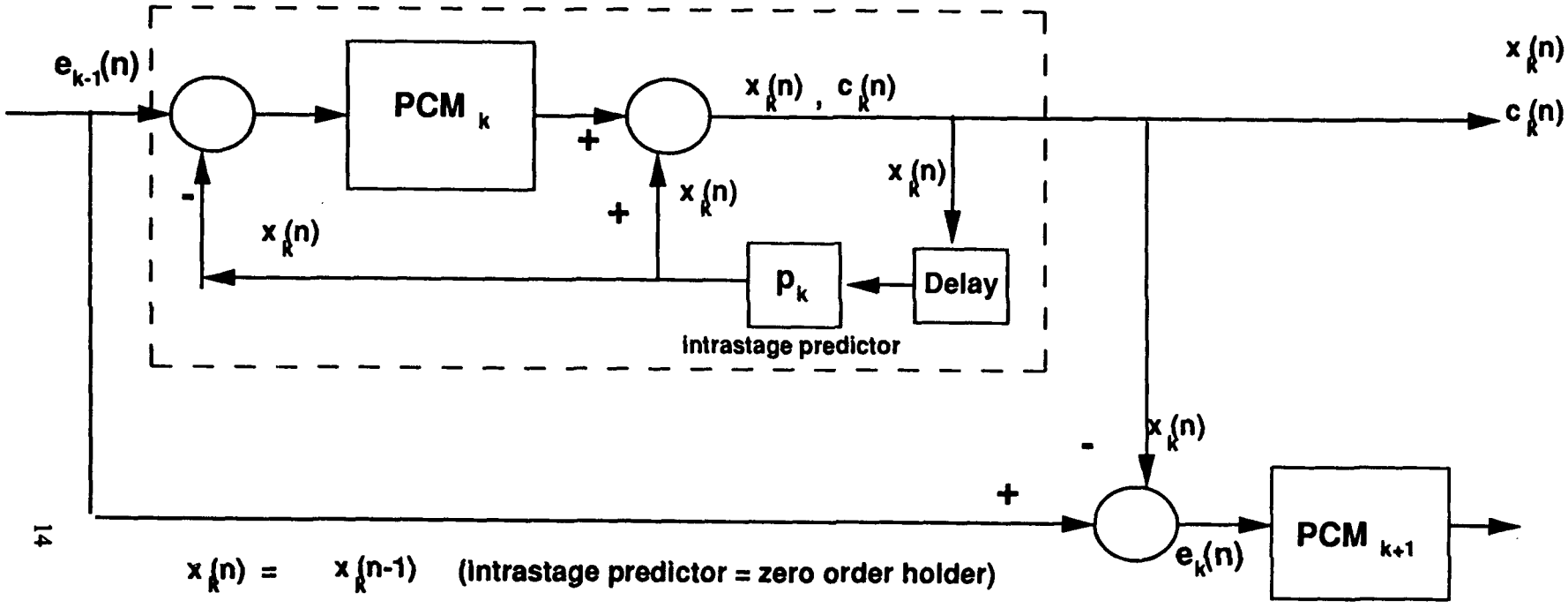


Figure 5: k-th Stage 1-D MPCM

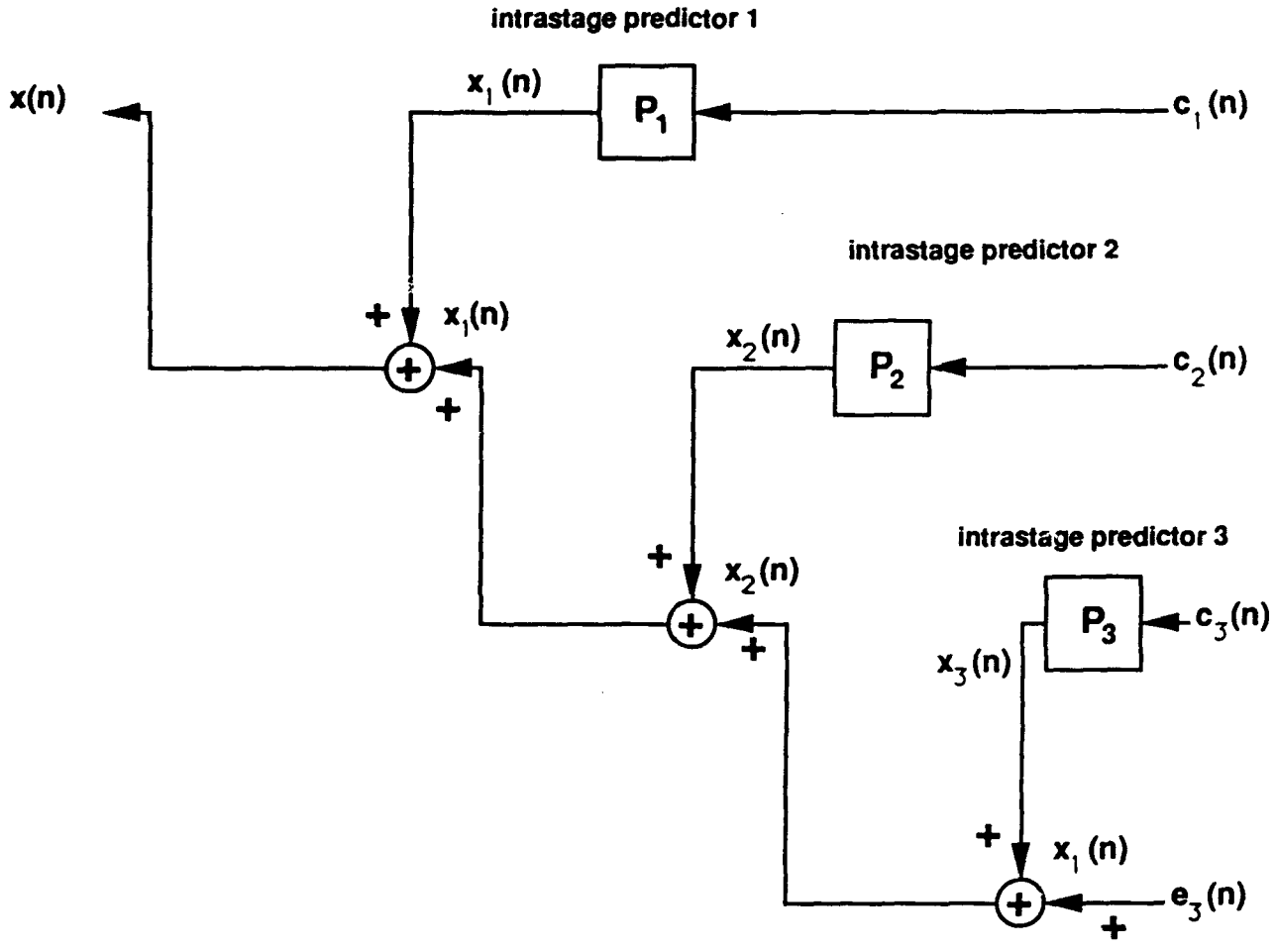


Figure 6: Decoder of 3-Stage MPCM

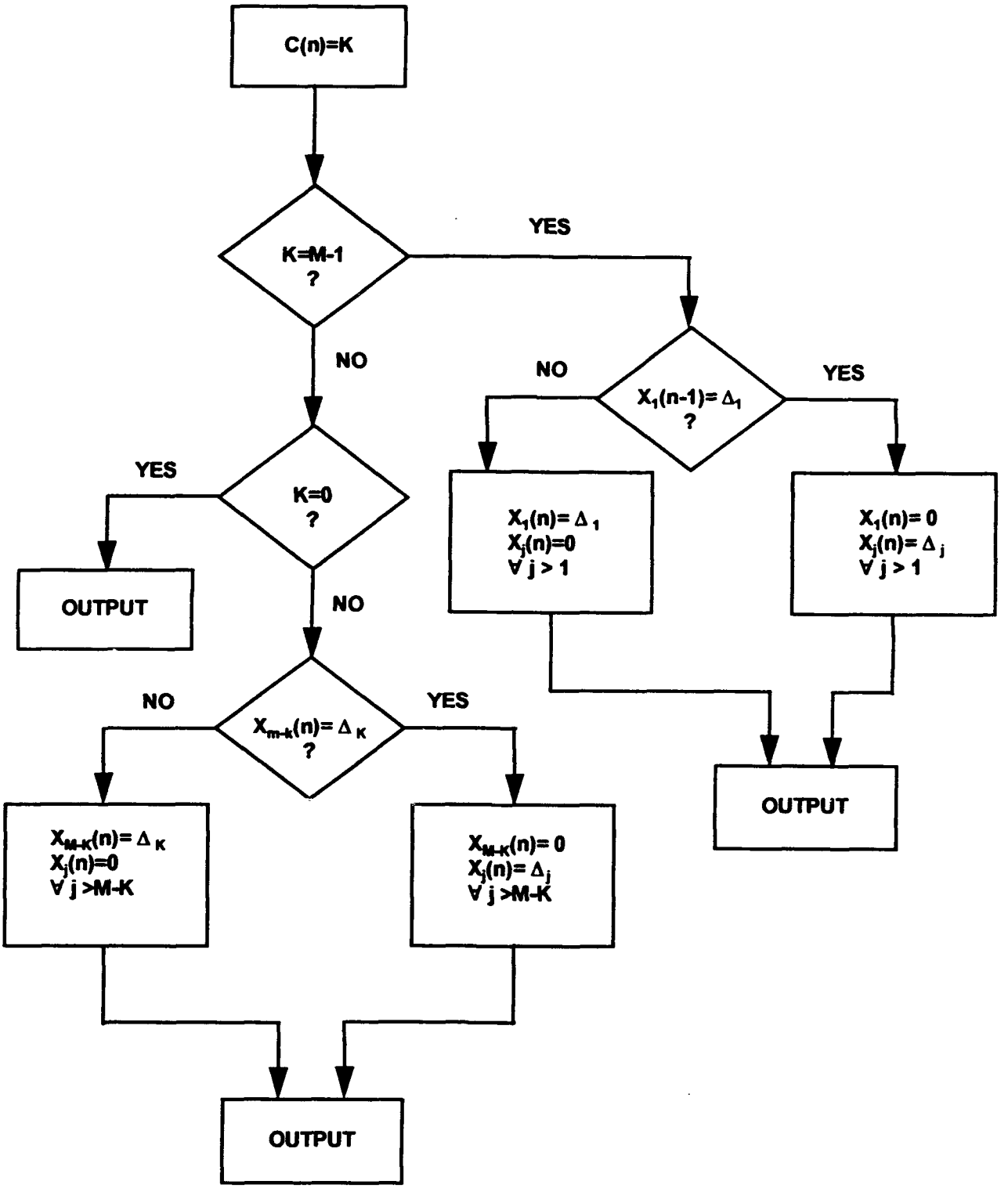


Figure 7: 1-D MPCM Decoding Algorithm

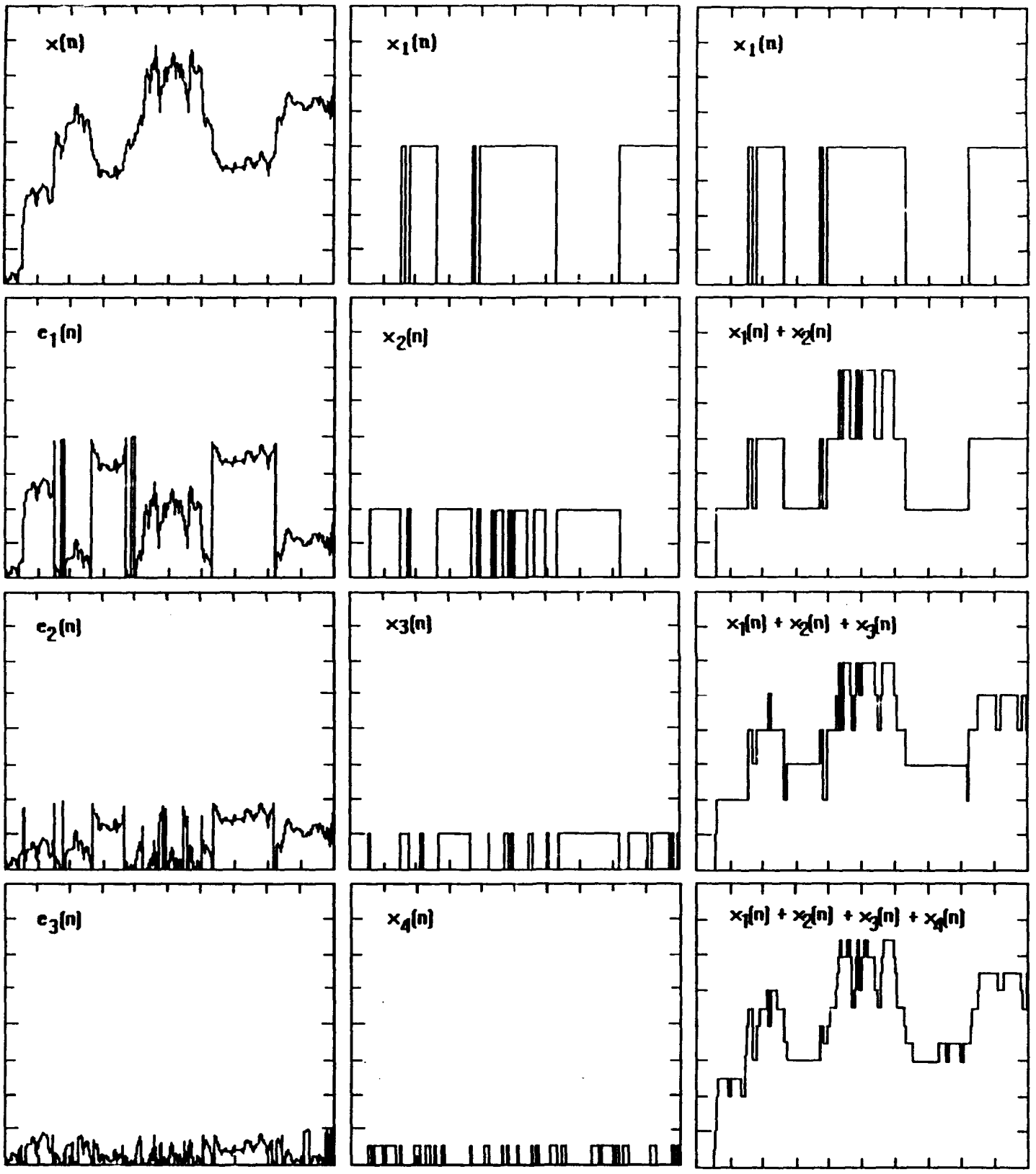


Figure 8: 3-Bit Multistage PCM

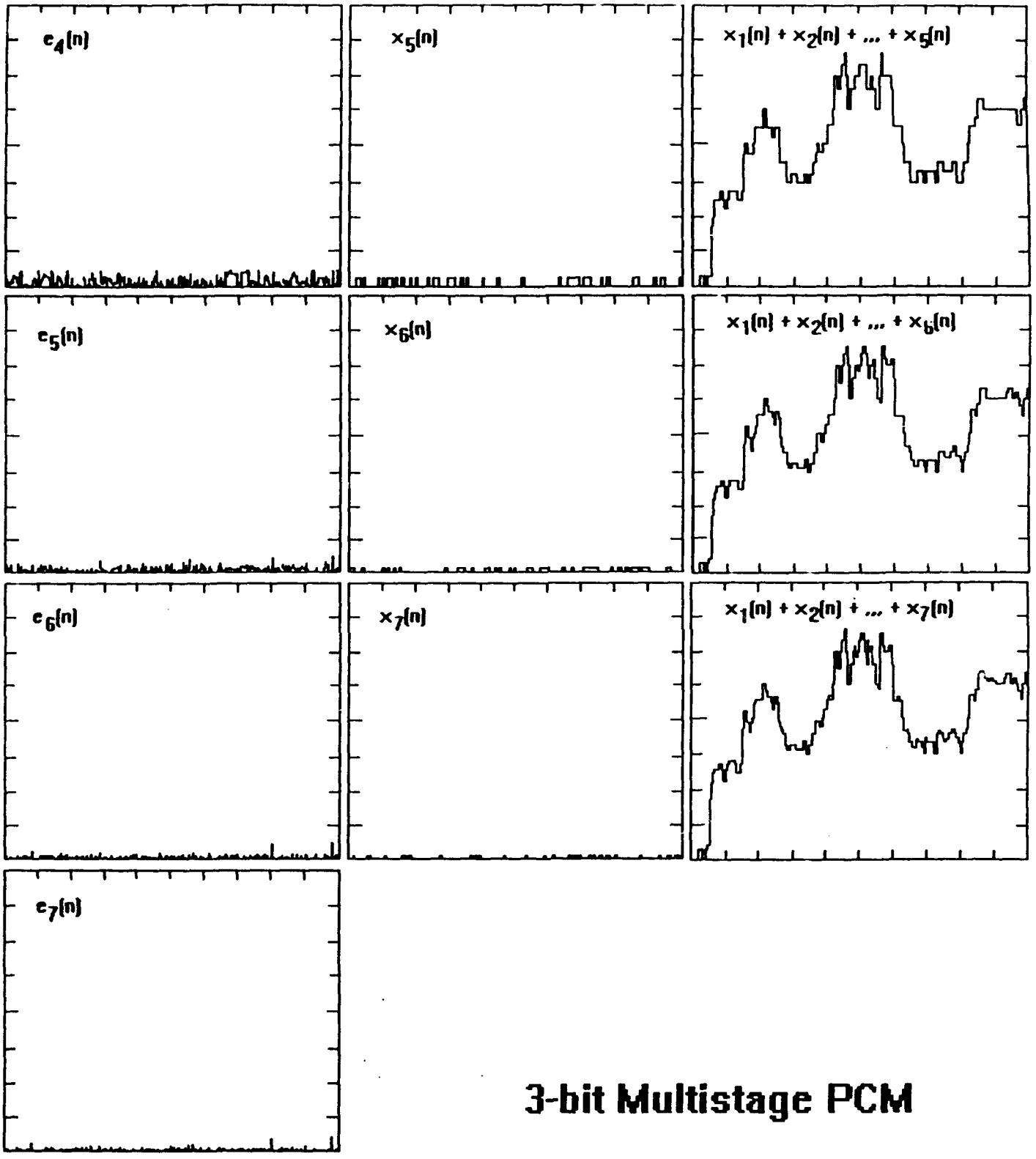


Figure 8: 3-Bit Multistage PCM (Continued)

Table 1. An example for encoding of 3-bit multistage PCM (MPCM).

$x'(n) = x_1(n) + x_2(n) + x_3(n) + , , , + x_8(n)$

$e(n) = x(n) - x'(n)$

MPCM encoding algorithm												
INPUT		PREDICTION		SIGNAL COMPONENTS								OUTPUT
n	x(n)	x'(n)	e(n)	$x_1(n)$ 128	$x_2(n)$ 64	$x_3(n)$ 32	$x_4(n)$ 16	$x_5(n)$ 8	$x_6(n)$ 4	$x_7(n)$ 2	$x_8(n)$ 1	C(n)
0	0	0	0	0	0	0	0	0	0	0	0	0
1	20	16	4	0	0	0	16	0	0	0	0	4
2	22	20	2	0	0	0	16	0	4	0	0	2
3	25	24	1	0	0	0	16	8	0	0	0	3
4	29	28	1	0	0	0	16	8	4	0	0	2
5	35	32	3	0	0	32	0	0	0	0	0	5
6	34	34	0	0	0	32	0	0	0	2	C	1
7	32	32	0	0	0	32	0	0	0	0	0	1
8	30	31	-1	0	0	0	16	8	4	2	1	5
9	30	30	0	0	0	0	16	8	4	2	0	0
10	30	30	0	0	0	0	16	8	4	2	0	0

Table 2. An example for decoding of 3-bit multistage PCM (MPCM).

$x'(n) = x_1(n) + x_2(n) + x_3(n) + , , , + x_8(n)$

MPCM decoding algorithm										
INPUT		SIGNAL COMPONENTS								OUTPUT
n	C(n)	$x_1(n)$ 128	$x_2(n)$ 64	$x_3(n)$ 32	$x_4(n)$ 16	$x_5(n)$ 8	$x_6(n)$ 4	$x_7(n)$ 2	$x_8(n)$ 1	x'(n)
0	0	0	0	0	0	0	0	0	0	0
1	4	0	0	0	16	0	0	0	0	16
2	2	0	0	0	16	0	4	0	0	20
3	3	0	0	0	16	8	0	0	0	24
4	2	0	0	0	16	8	4	0	0	28
5	5	0	0	32	0	0	0	0	0	32
6	1	0	0	32	0	0	0	2	0	34
7	1	0	0	32	0	0	0	0	0	32
8	5	0	0	0	16	8	4	2	1	31
9	0	0	0	0	16	8	4	2	0	30
10	0	0	0	0	16	8	4	2	0	30

Table 3. An example for encoding of 3-bit multistage PCM (MPCM).

$x'(n) = x_1(n) + x_2(n) + x_3(n) + , , , + x_8(n)$

$e(n) = x(n) - x'(n)$

MPCM encoding algorithm												
INPUT		PREDICTION		SIGNAL COMPONENTS								OUTPUT
n	x(n)	x'(n)	e(n)	$x_1(n)$ 128	$x_2(n)$ 64	$x_3(n)$ 32	$x_4(n)$ 16	$x_5(n)$ 8	$x_6(n)$ 4	$x_7(n)$ 2	$x_8(n)$ 1	c(n)
0	0	0	0	0	0	0	0	0	0	0	0	0
1	100	64	36	0	64	0	0	0	0	0	0	6
2	102	96	6	0	64	32	0	0	0	0	0	5
3	120	112	8	0	64	32	16	0	0	0	0	4
4	120	120	0	0	64	32	16	8	0	0	0	3
5	120	120	0	0	64	32	16	8	0	0	0	0
6	118	119	-1	0	64	32	16	0	4	2	1	3
7	110	111	-1	0	64	32	0	8	4	2	1	4
8	100	103	-3	0	64	32	0	0	4	2	1	3
9	100	101	-1	0	64	32	0	0	4	0	1	1
10	100	100	0	0	64	32	0	0	4	0	0	0

Table 4. An example for decoding of 3-bit multistage PCM (MPCM).

$x'(n) = x_1(n) + x_2(n) + x_3(n) + , , , + x_8(n)$

MPCM decoding algorithm										
INPUT		SIGNAL COMPONENTS								OUTPUT
n	c(n)	$x_1(n)$ 128	$x_2(n)$ 64	$x_3(n)$ 32	$x_4(n)$ 16	$x_5(n)$ 8	$x_6(n)$ 4	$x_7(n)$ 2	$x_8(n)$ 1	x'(n)
0	0	0	0	0	0	0	0	0	0	0
1	6	0	64	0	0	0	0	0	0	64
2	5	0	64	32	0	0	0	0	0	96
3	4	0	64	32	16	0	0	0	0	112
4	3	0	64	32	16	8	0	0	0	120
5	0	0	64	32	16	8	0	0	0	120
6	3	0	64	32	16	0	4	2	1	119
7	4	0	64	32	0	8	4	2	1	111
8	3	0	64	32	0	0	4	2	1	103
9	1	0	64	32	0	0	4	0	1	101
10	0	0	64	32	0	0	4	0	0	100

Table 5. Error Analysis for PCM, Multistage PCM (MPCM) and DPCM

schemes	bpp	levels	entropy rate	MSE	NMSE	SNR (dB)
8-bit PCM	8	128, 64, ,,, ,2,1	7.100	< 1.00	< 1/10280	> 40
7-bit PCM	7	128, 64, ,,, , 2	6.124	0.48	0.000047	43.28
6-bit PCM	6	128, 64, ,,, , 4	5.135	3.49	0.000340	34.69
5-bit PCM	5	128, 64, ,,, , 8	4.158	17.83	0.001734	27.61
4-bit PCM	4	128, 64, 32, 16	3.199	78.83	0.007668	21.15
3-bit PCM	3	128, 64, 32	2.269	287.17	0.027935	15.54
2-bit PCM	2	128, 64	1.675	1575.89	0.153297	8.15
1-bit PCM	1	128	0.784	3802.24	0.369868	4.32
3-bit DPCM	3	64, 32, 16	1.431	249.17	0.024238	15.06
3-bit DPCM	3	32, 16, 8	1.587	57.09	0.005554	22.55
3-bit DPCM	3	16, 8, 4	2.060	14.10	0.001371	28.63
3-bit DPCM	3	8, 4, 2	2.764	7.03	0.000684	31.65
3-bit DPCM	3	4, 2, 1	2.935	14.90	0.001449	28.39
3-bit MPCM	3	128, 64, ,,, ,2	2.778	23.51	0.002287	26.41
2-bit MPCM	2	128, 64, 32	0.908	238.12	0.023152	16.35
1-bit MPCM	1	128	0.142	3675.80	0.357568	4.47

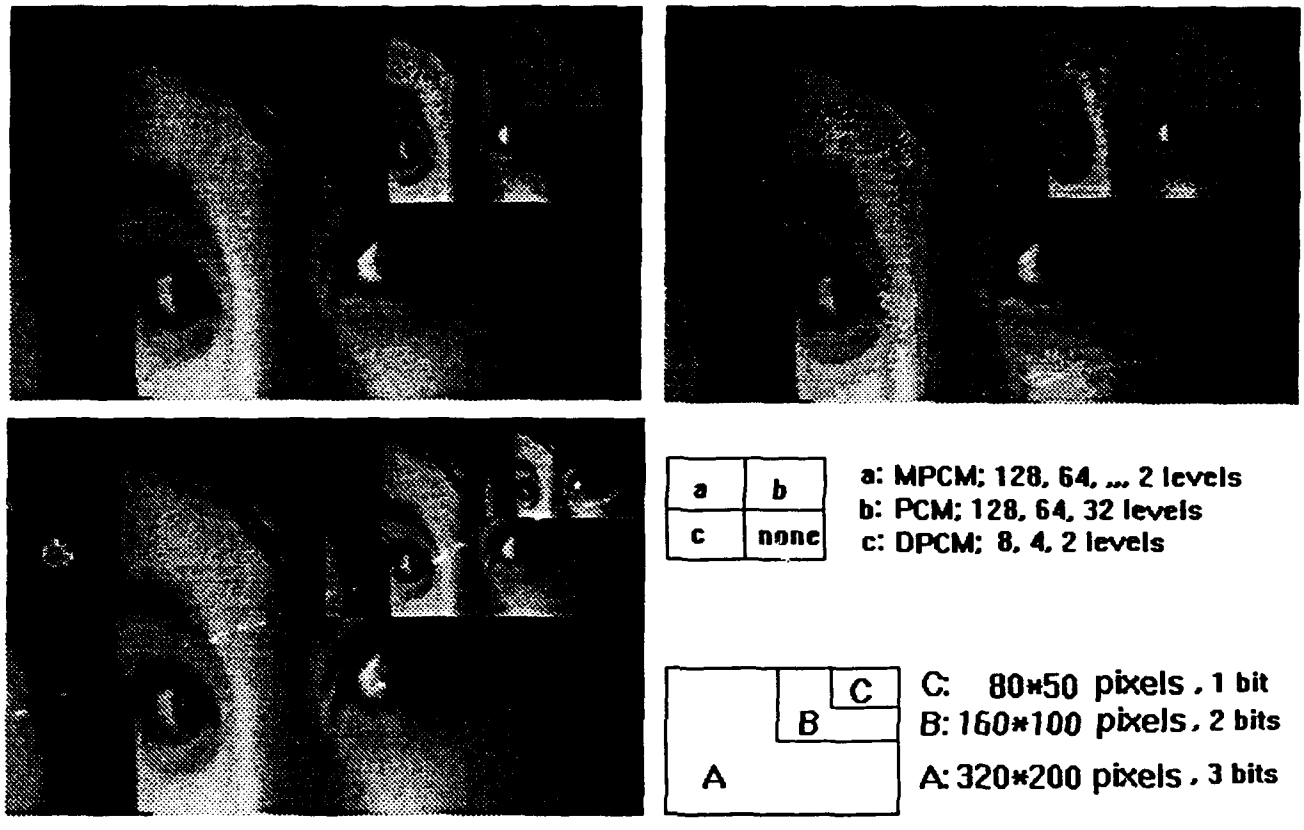
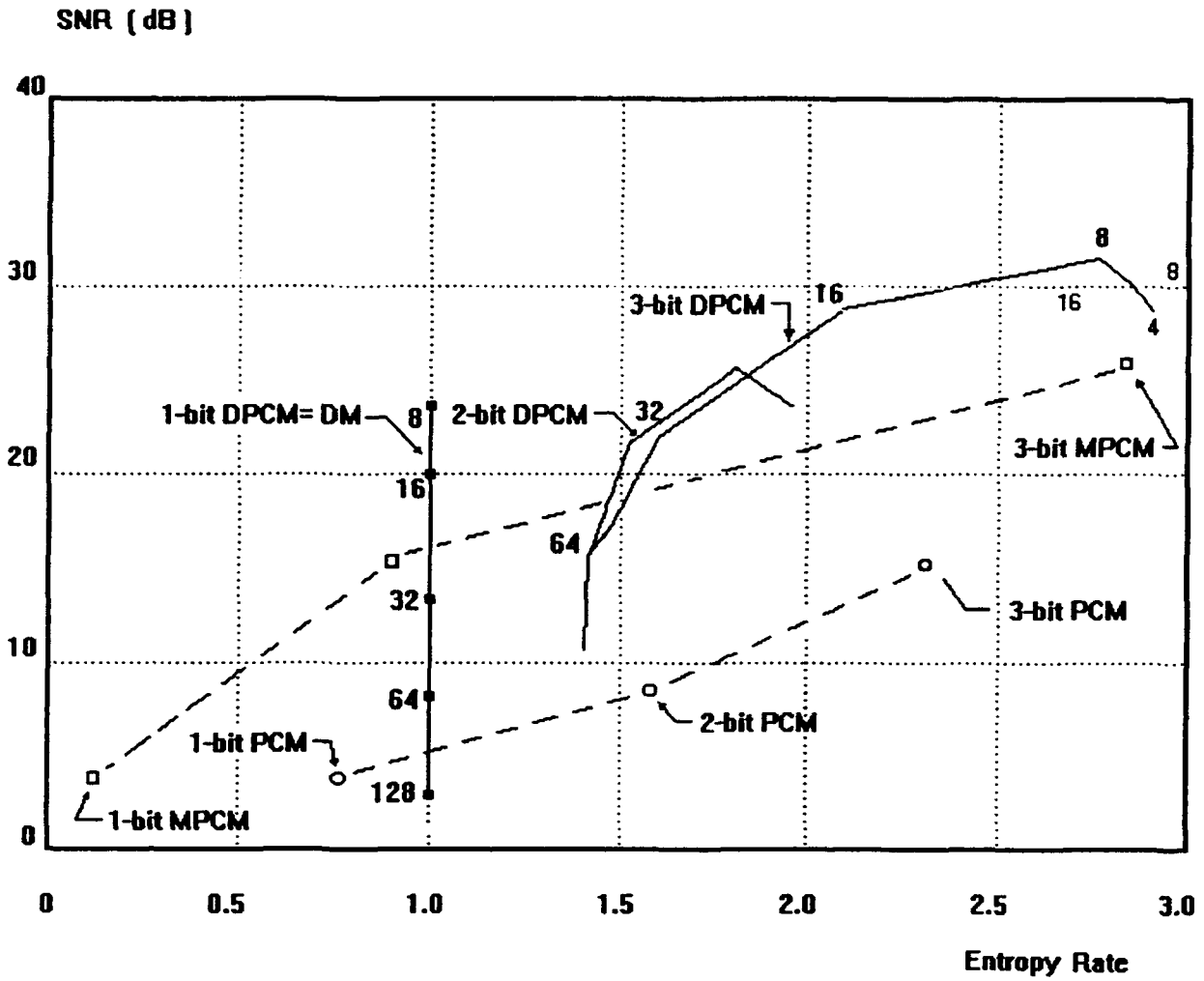


Figure 9: Effect of Spatial Resolution Changes on Different Methods



The signal-noise-ratio against entropy-rate curves with stage levels specified by 128, 64, ... 4 of 3-bit, 2-bit and 1-bit PCM, MPCM, DPCM .

Figure 10: Signal-to-Noise Curves

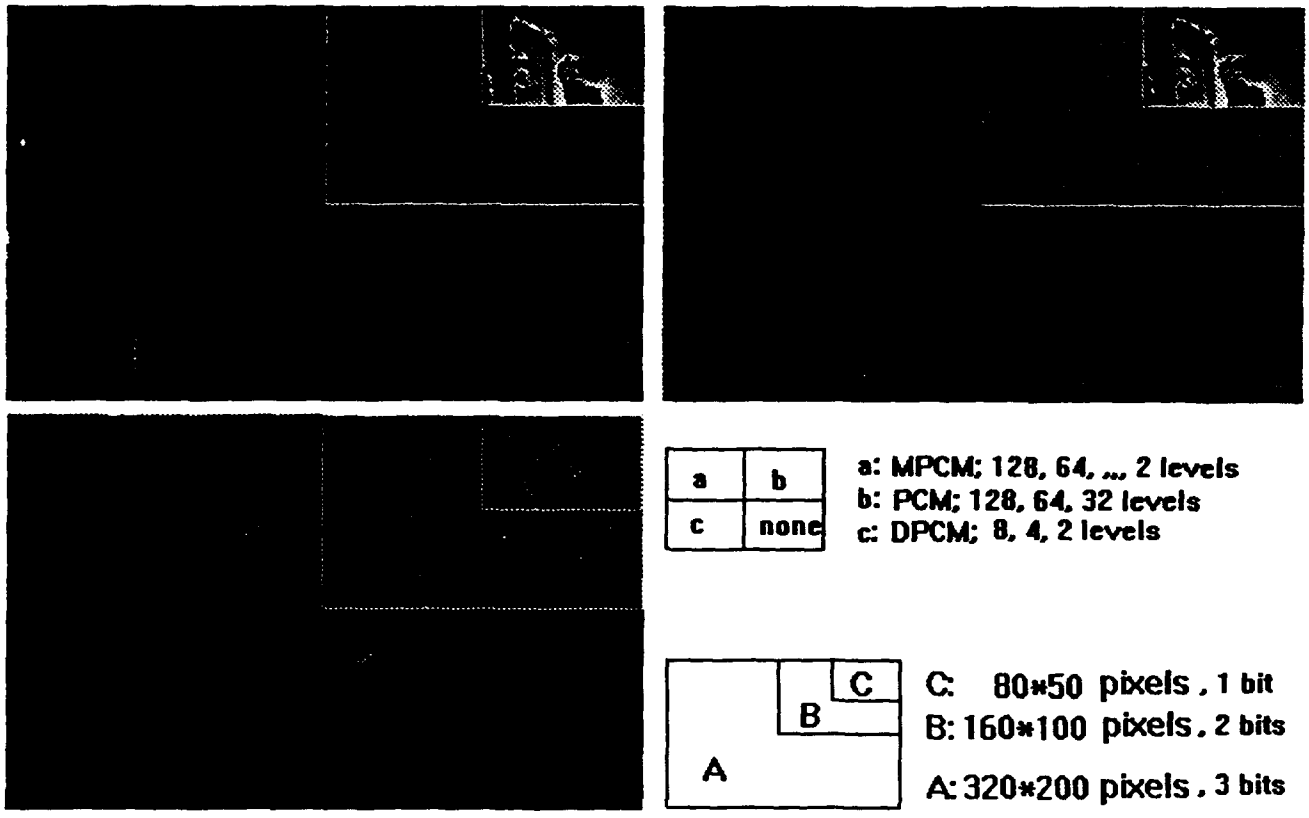


Figure 11: Error Images Corresponding to Figure 9

CHAPTER 3

MPCM: PROGRESSIVE EDGE DETECTION

In [1], an edge in an image is described to be a boundary or contour at which a significant change occurs in some physical aspect of an image, such as the surface reflectance, illumination, or the distances of the visible surfaces from the viewer. Change in physical aspects manifest themselves in a variety of ways, including changes in intensity, color, and texture. It is known that an image that consists of only edges is highly intelligible. Marr and Hildreth [2] observed that significant intensity changes occur at different scales (resolutions) in an image. Optimal detection of significant intensity changes, therefore, generally requires the use of operators that respond to several different scales. Marr and Hildreth suggest that the original image be bandlimited at several different cutoff frequencies and an edge detection algorithm be applied to each of the images. The resulting edge maps have edges corresponding to different scales. They also argue that edge maps of different scales contain important information about physically significant parameters. The visual world is made of elements such as contours, scratches, and shadows, which are highly localized at their own scale. This localization is also reflected in such physically important changes as reflectance change and illumination change. If the same edge is present in a set of edge maps of different scales, it represents the presence of an image intensity change due to a single physical phenomenon. If an edge is present in only one edge map, one reason may be that two independent physical phenomena are operating to produce intensity changes in the same region of the image.

3.1 Progressive Edge Detection

One of strengths of MPCM is its edge detection capability which is attributed to the priority code. It extracts edges of objects from the background according to priorities assigned by the priority code. Since an abrupt change between two contiguous pixels will affect at least one component in a certain stage, the priority code will record this information so that when the image is reconstructed, this information can be retrieved for updating the data in that stage. As a consequence, decoding the priority code based on a particular stage, say k generates a coded image composed of pixels with values changed in the k-th stage. The resulting coded image represents an edge map resulting from pixels which have a stage prioritized by k. Generally speaking, the most significant edges will be detected by stage 1 assigned by the highest priority integer and the least important edges will be extracted by the last stage with the lowest priority integer. A sequence of improved coded images can be generated by processing more than one stage. For example, a coded image obtained by processing stages 1 and 2 is an edge map composed of all pixels with stages prioritized by the two highest integers. Therefore, the more stages processed, the more edges detected. This idea is quite different from commonly used methods, zero-crossing of derivatives such as a Laplacian operator.

3.2 Experimental Results

In order to demonstrated the effectiveness of MPCM, an experiment is conducted on a vapor cloud image obtained from CRDEC. The result shows that MPCM is indeed a promising detection technique which is not only easy to implement, but also suitable for real-time implementation.

The experiments given below were performed based on two images provided by CRDEC which were taken from a thermal imager at two different time frames. One of the images has a chemical vapor cloud embedded in the background as shown in Figure 12(a) and the other is Figure 12(b) which contains no clouds but background clutter. Figure 12(c) is the differential image resulting from subtracting Figure 12(b) from Figure 12(a). The MPCM implemented here uses 7 stage levels specified by $\{\Delta_k = 2^{8-k}\}_{k=1}^{7}$ where the last stage specified by level 0 is of no interest. Figures 13 and 14 are decompositions of Figures 12(a) and 12(c) respectively. Figures 15 and 16 contain 7 coded images of Figure 12(a) and 12(c) respectively. Figures 17 and 18 are obtained by summing coded images successively from Figures 12(a) and 12(c). Figure 19 and 20 are reconstructed images of Figure 12(a) and Figure 12(c) respectively. From Figures 13, 15, 17 and 19 which were obtained by Figure 12(a), the clouds are heavily obscured by the clutter and are very hard to detect. However, as shown in Figures 14, 16, 18 and 20 obtained by the differential image Figure 12(c), the edges of the vapor cloud begin to emerge after the first stage is processed by the priority code. In this case, the highest priority integer is 7 and the corresponding stage level is 128. As more stages are processed according to decreasing priorities, more edges are detected and extracted. (It should be noted that a stage prioritized by k is stage $8-k$ with stage level 2^{8-k}.) As experimental results suggested, a progressive edge detection algorithm coupled with background suppression will significantly improve detectability. As a matter of fact, in most of practical cases, there is no need of processing all stages to detect objects. It generally requires a few stages to extract significant edges which are sufficient for object detection. Therefore, the cost of processing images is drastically reduced. This is very crucial in real-time signal processing.

References

1. J.S. Lim, *Two-Dimensional Signal and Image Processing*, Prentice Hall, 1990.

2. D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc.,*, London, vol. B207, 1980, pp. 187-217.

Text continues on page 36.

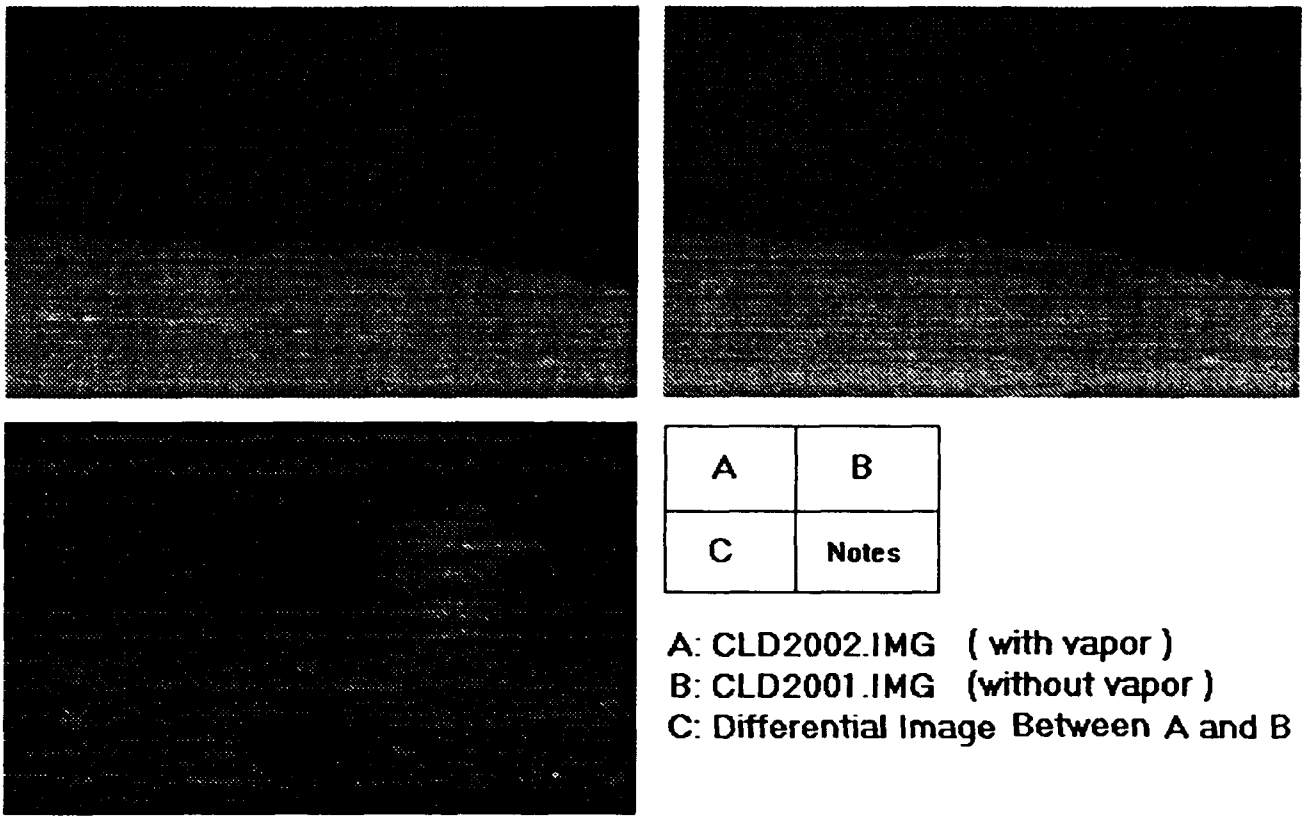


Figure 12: Vapor Cloud Image

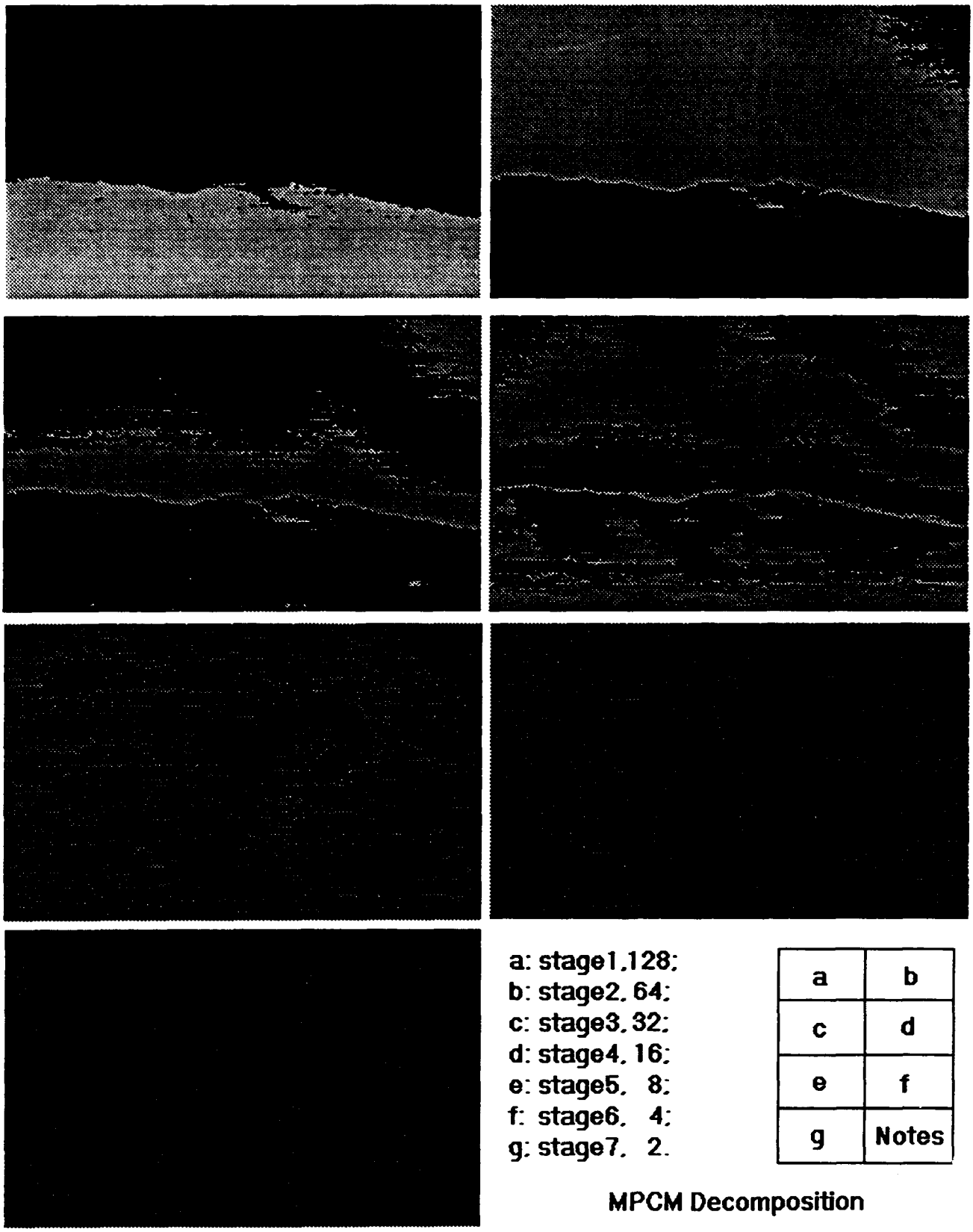


Figure 13: MPCM Decomposition

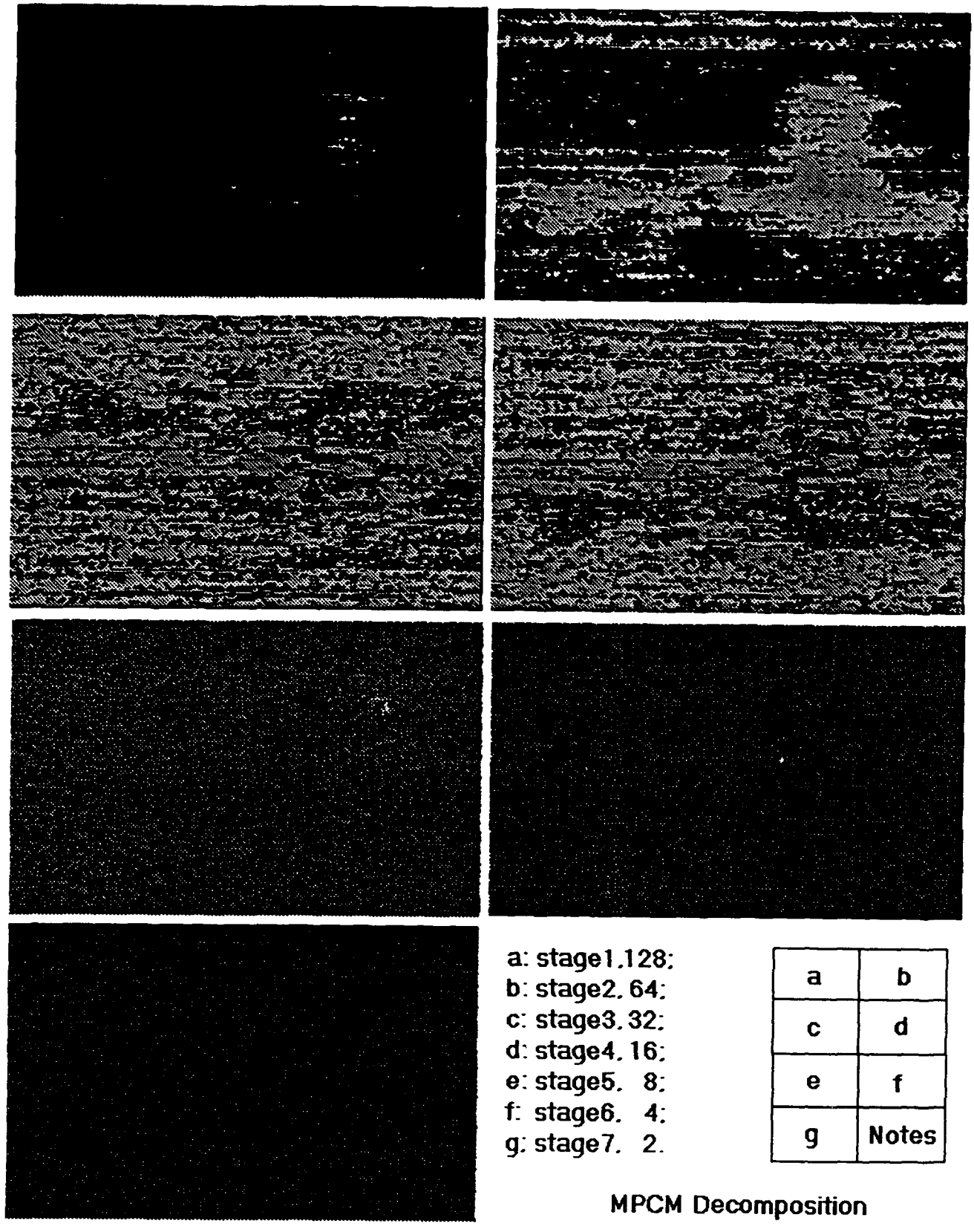


Figure 14: MPCM Decomposition

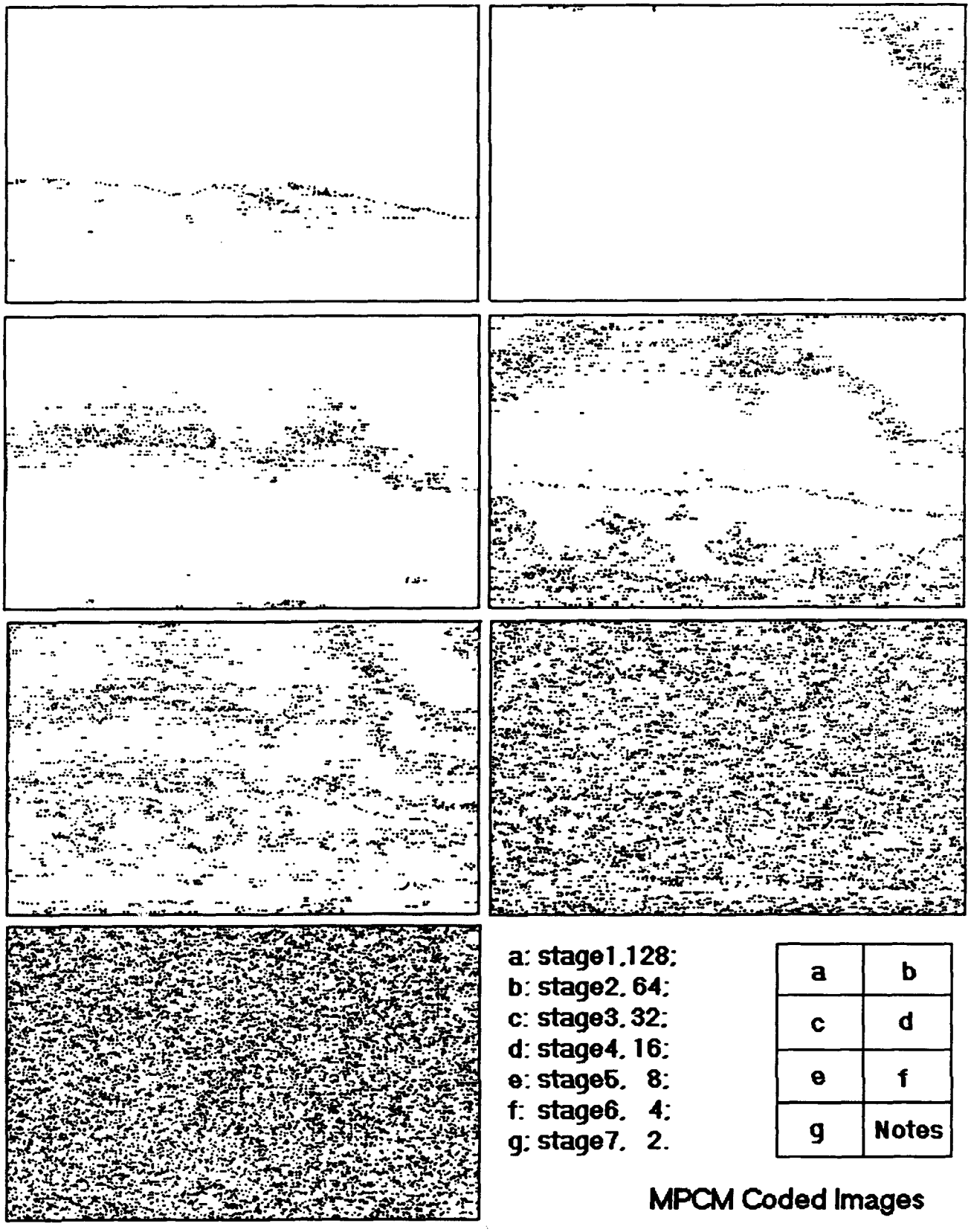


Figure 15: MPCM Coded Images

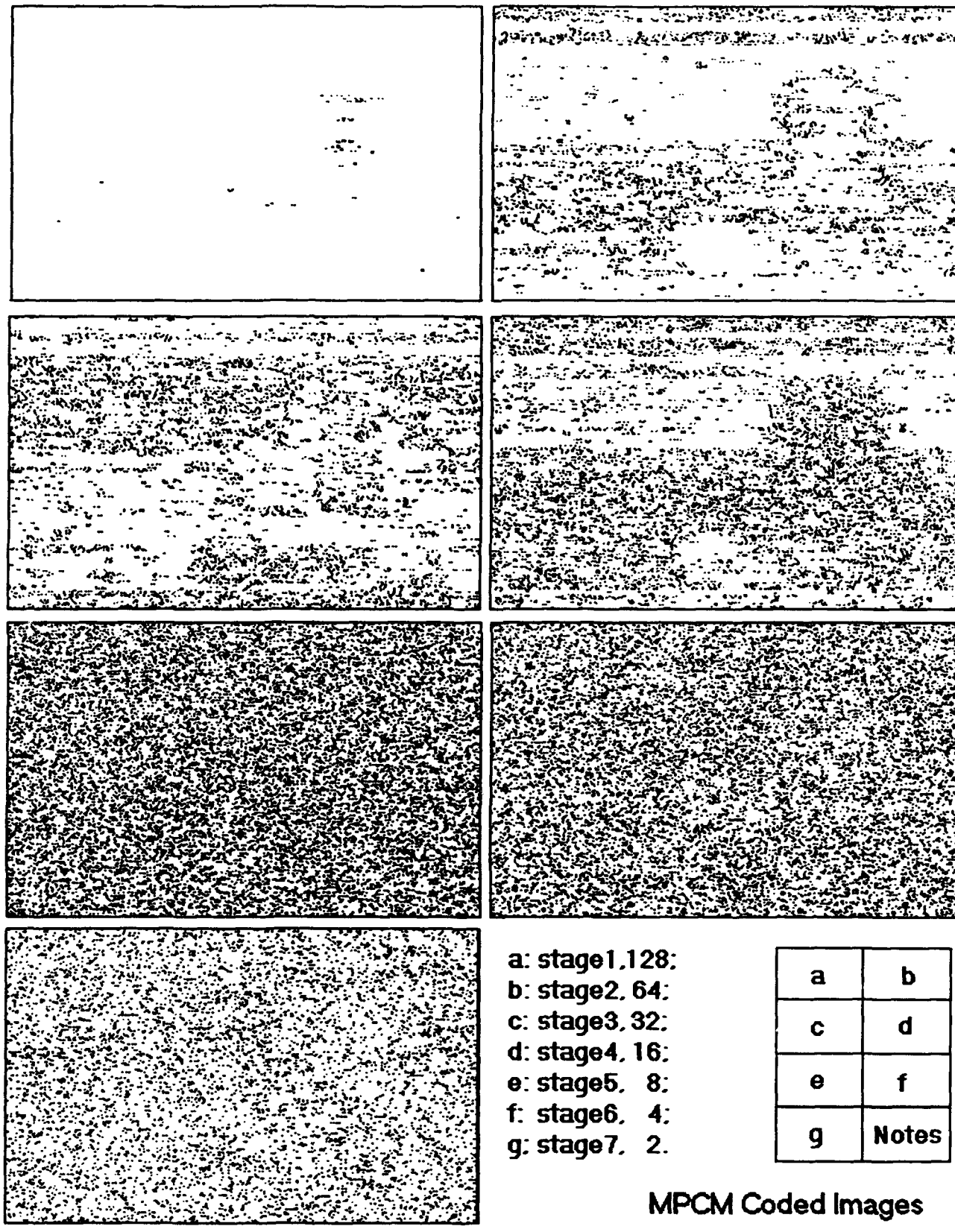


Figure 16: MPCM Coded Images

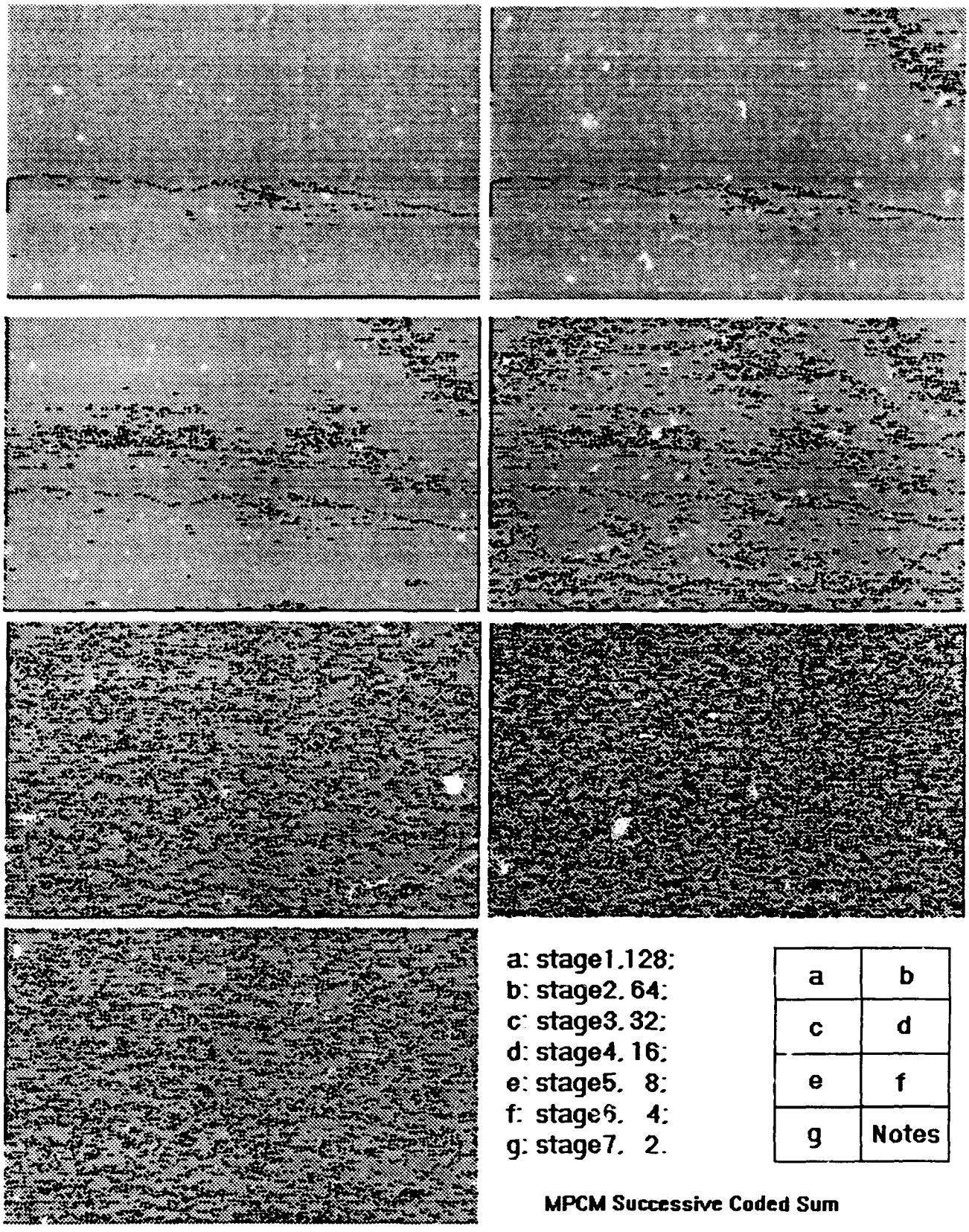


Figure 17: MPCM Successive Coded Sum

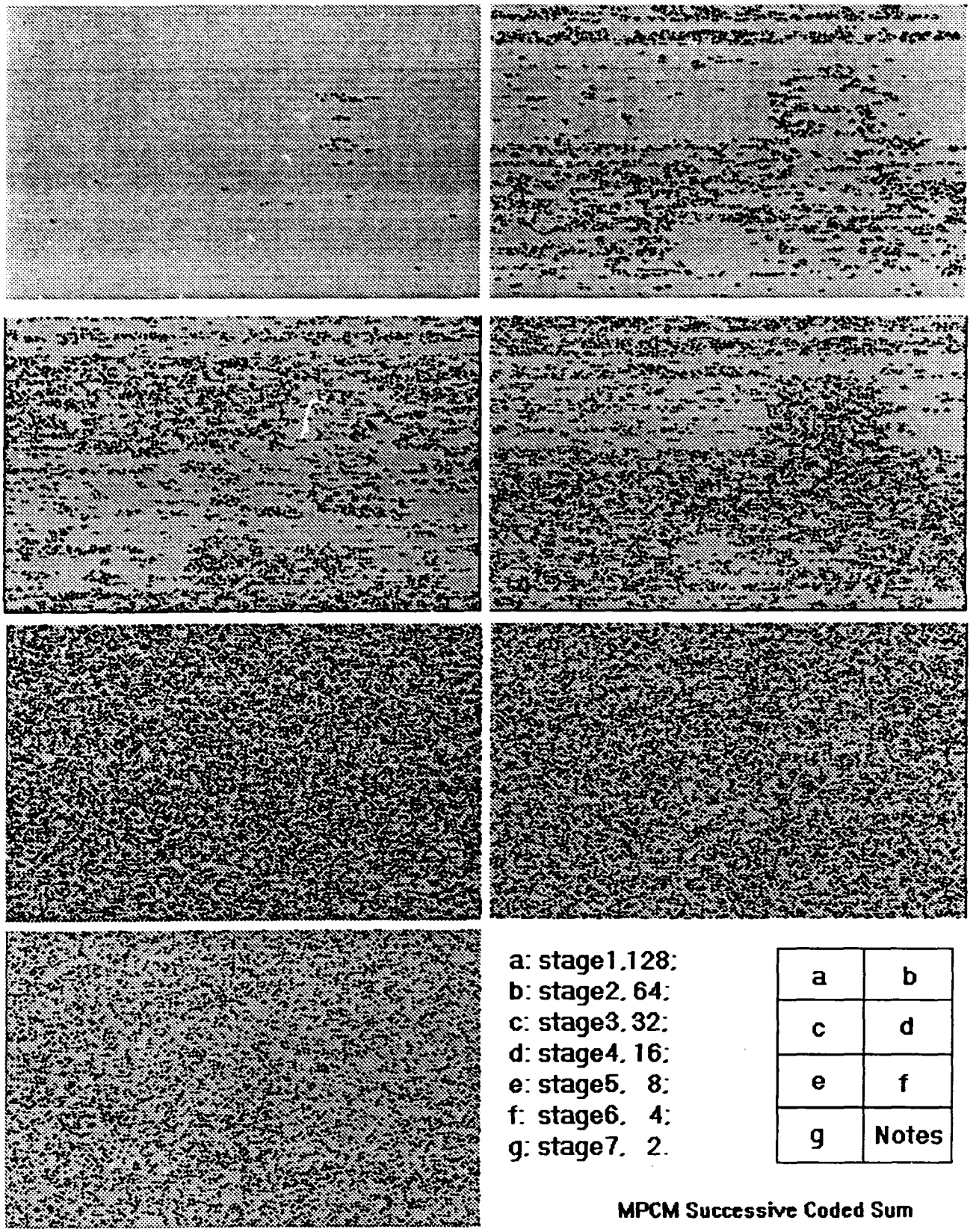


Figure 18: MPCM Successive Coded Sum

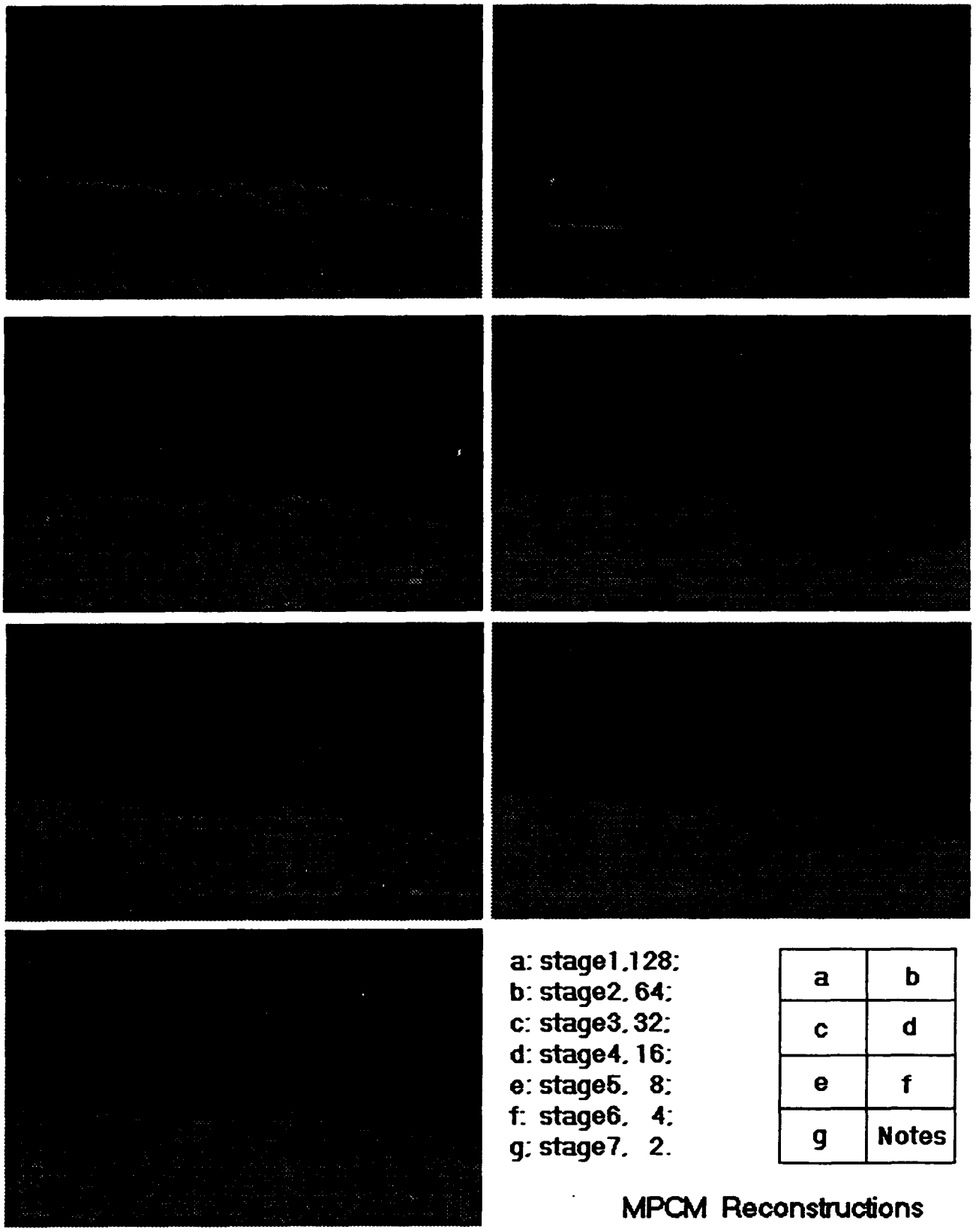


Figure 19: MPCM Reconstructions

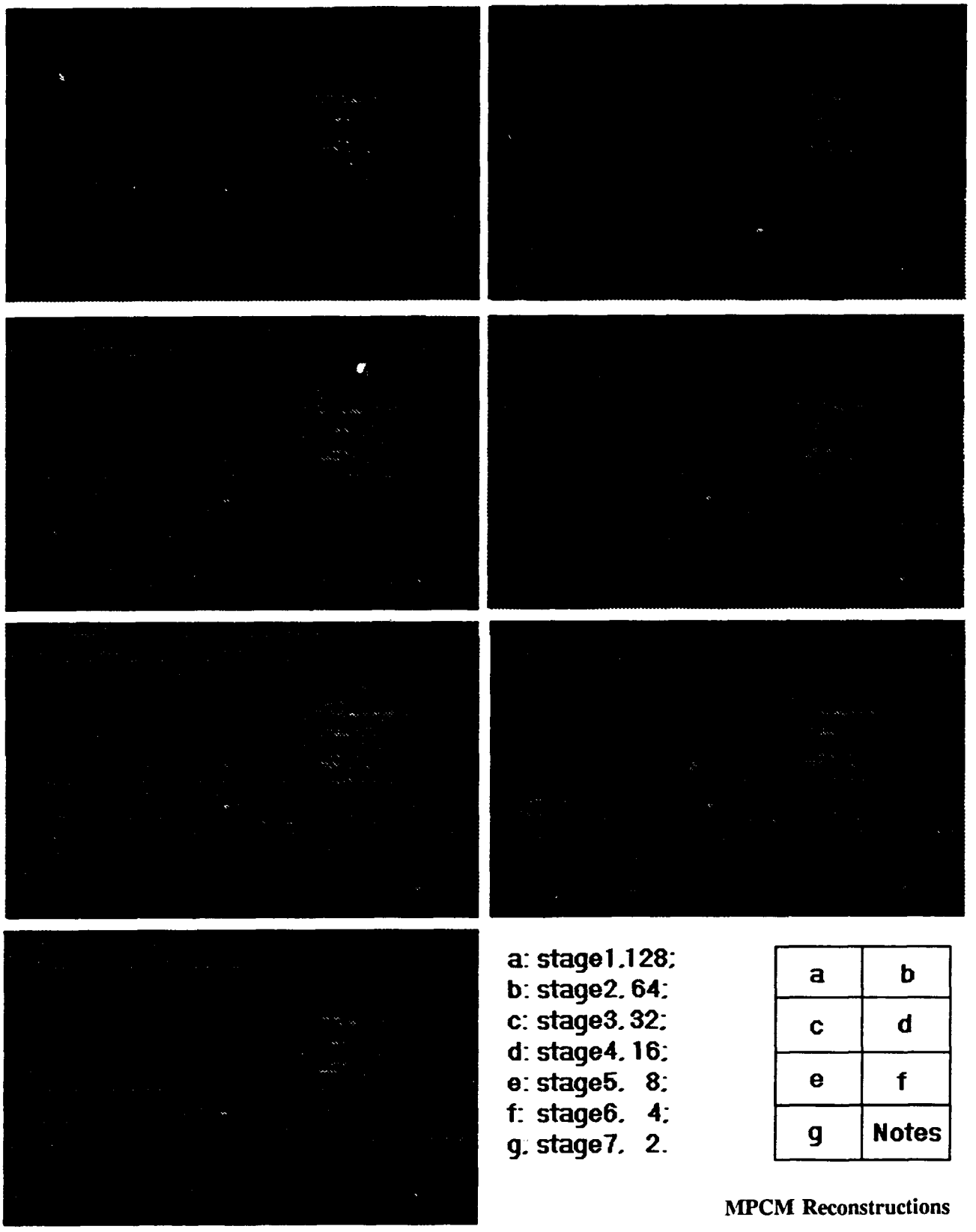


Figure 20: MPCM Reconstructions

CHAPTER 4

MPCM: TOP-DOWN GRAY-LEVEL TRIANGLE METHOD

A top-down M-tier gray-level triangle is a triangle structure generated by a sequence of M sets of gray levels, one stacked on top of another where each tier is a set of gray levels and the set of gray levels at the k-th tier, denoted by G_k is properly contained in that at the $k+1$-th tier, G_{k+1}, namely, $G_k \subsetneq G_{k+1}$. Since image quality is determined by the number of gray levels to be used, a top-down gray-level triangle generates a series of coarse-to-fine images with resolutions determined by tiers in the triangle. It first puts on the top tier of a triangle a set of gray levels, G_1 to be used to decompose an image into an image component in stage 1. The second tier of the triangle, G_2 is made of gray levels used to produce the image component in the second stage included in which are the gray levels contained in the top tier. The procedure is continued until a top-down gray-level triangle is completely generated. Each tier represents a stage and each stage generates an image component which is a coarse version of the original image. To reconstruct an image, we first use the gray levels contained in the top tier to produce the coarsest image, then move down to the next tier of the triangle to produce a finer image, etc. The process described is called a top-down gray-level triangle method.

According to the procedure described above, MPCM can be viewed as a top-down gray-level triangle method. It has several distinct differences in structure from the commonly used bottom-up pyramid methods [1]. (1) In image decomposition, MPCM uses a top-down gray-level triangle as a basis to represent an image from coarse to fine resolutions in multiple stages with the resolutions specified by tiers of the triangle from top to bottom, while a bottom-up pyramid method such as Burt-Adelson's Gaussian-Laplacian pyramid method produces an image pyramid from fine-to-coarse using various window functions. (2) Each tier of a top-down gray-level triangle represents a stage specified by a set of gray levels; thus, it requires different bit rates for coding. However, in pyramid methods, all layers of a bottom-up image pyramid require the same bit rate. (3) The number of tiers for a top-down gray-level triangle is determined by the number of bits to be used for coding, but the number of layers of a bottom-up image pyramid is independent of bit rates but determined by the window size used for masking. (4) The sizes of image components generated by MPCM are the same, whereas, the sizes of images at different layers in an image pyramid vary and generally are scaled by a constant such as 2. (5) The quality of images generated by a top-down gray-level triangle is gradually improved as tiers go down. However, it is opposite for a bottom-up image pyramid where the resolution is refined by upsampling from top to bottom. (6) MPCM can achieve bit-saving by processing the first few levels of a top-down gray-level triangle while it still produces a satisfactory result. A bottom-up image pyramid cannot do this because all layers of a bottom-up image pyramid requires the same bit rate. (7) The stage level represented by each tier in MPCM plays the same role as does that of the Laplacian pyramid in pyramid coding. Several experiments are studied and compared to Burt-Adelson's Laplacian bottom-up pyramid coding method.

4.1 Top-Down Gray-level Triangle

In what follows, we describe how MPCM can be implemented as a top-down gray level method. An M-tier top-down gray-level triangle to be used in MPCM is made of M gray levels $\{\Delta_k\}_{k=1}^{M}$ where the gray levels Δ_k are arranged in decreasing order.

It should be noted that in each tier, k there is a stage level, Δ_k associated with k. The k-th tier creates the k-th stage designated by its stage level Δ_k. With the stage level defined, the gray levels of $k+1$-th tier, G_{k+1} is formed by adding the $k+1$-th stage level Δ_{k+1} to G_k, the gray levels of the k-th tier, $G_{k+1} = \{\Delta_{k+1}\} \cup G_k$. For example, Figure 21 shows an 8-tier top-down gray-level triangle with $\{\Delta_k = 2^{8-k}\}_{k=1}^{8}$ where in the k-th tier of the triangle, the set of gray levels is $G_k = \{0\} \cup \{2^{8-n}\}_{n=1}^{k}$. That is, the top tier contains gray levels 0 and 128 and the 8-th tier has gray levels $G_8 = \{0, 2, 4, 8, 16, 32\ 64, 128\}$. The value in quotes, "$2^{8-k}$" is the k-th stage level and $G_{k+1} = G_k \cup \{2^{8-k-1}\}$. Based on the top-down gray-level triangle described above, MPCM creates one stage for each tier with which a stage level is assigned.

G_1	0 "128"	$\Delta_1 = 2^7 = 128$
G_2	0 "64" 128	$\Delta_2 = 2^6 = 64$
G_3	0 "32" 64 128	$\Delta_3 = 2^5 = 32$
G_4	0 "16" 32 64 128	$\Delta_4 = 2^4 = 16$
G_5	0 "8" 16 32 64 128	$\Delta_5 = 2^3 = 8$
G_6	0 "4" 8 16 32 64 128	$\Delta_6 = 2^2 = 4$
G_7	0 "2" 4 8 16 32 64 128	$\Delta_7 = 2^1 = 2$
G_8	0 "1" 2 4 8 16 32 64 128	$\Delta_8 = 2^0 = 1$

Figure 21: Top-Down Gray-Level Triangle

A multistage quantizer divides the encoding task into successive stages, where the first stage performs a crude quantization using a small codebook. Then a second stage quantizer performs a quantization on the error data resulting from the difference between the original data and quantized data by the first stage. A third stage quantizer further quantizes the second stage error data to refine error accuracy. Continuing this procedure results in a sequence of cascaded quantizers. MPCM employs such a multistage quantizer to achieve image decomposition where each stage level generates an image component with resolution determined by the gray levels in a top-down gray-level triangle. The stage levels used in the multistage quantizer for MPCM were depicted by Figure 1 where the stage levels Δ_k are used as quantization layers and an image is quantized by one layer at a time. More precisely, the multistage quantizer used in MPCM is a sequence of cascaded quantizers Q_k and each Q_k is defined by two quantization levels $\{0, \Delta_k\}$ and three threshold-decision intervals $\{(-\infty, 0), [0, \Delta_k], (\Delta_k, \infty)\}$. If the input to Q_k falls off the range $[0, \Delta_k]$, it will replaced by either Δ_k or 0 depending upon whether the input is positive or negative. Otherwise, a soft decision has to be made based on the prediction from a predictor.

While MPCM decomposes an image, it also generates a specially designed code, a priority code. The task of the priority code is to prioritize stages in accordance with the significance of data information. Assume that an MPCM system decomposes an image into M stages. The priority code assigns to a stage a number ranging from 0 to $M-1$. The most significant stage will be assigned $M-1$. It implies that the image component in that stage has the highest priority to be processed. Contrarily, the last stage, stage M which contains the least important information will be assigned 0. Therefore, it has the lowest priority. According to the above rule, the higher the

number assigned to the stage, the higher priority the stage. Since the priority code is the only information needed for the encoding and decoding processes, the number of bits to represent the priority code is that required for an M-stage MPCM system.

4.2 Burt-Adelson's Pyramid Method

One of most popular multiresolution techniques is the *Laplacian Pyramid Method* developed by Burt and Adelson in 1983 [2] which decomposes an original image into a set of low-pass filtered images in a pyramid structure, called Gaussian pyramid from which the original image can be reconstructed. The bottom layer of the pyramid corresponds to the original image; then a sequence of reduced-resolution images is successively generated by a family of Gaussian filters so that the images are stacked on top of one another. During the construction of Gaussian pyramid, a Laplacian pyramid is also generated by taking difference between two images at consecutive layers in the Gaussian pyramid. (It should be noted that the image in a higher layer must be expanded to the same size of that in the lower layer before subtraction). In other words, a layer of Laplacian pyramid represents an image resulting from the difference between two Gaussian functions convolved with the original image.

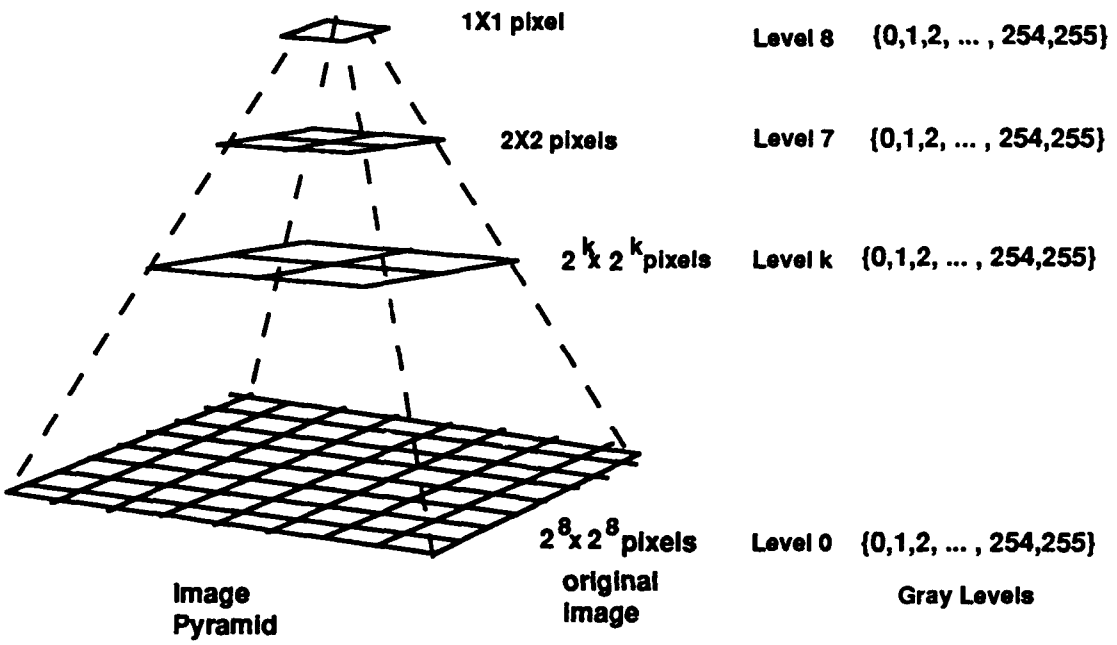


Figure 22: 8-level Bottom-Up Pyramid

Basically, Burt-Adelson's approach consists of two major operations which are *REDUCE* and *EXPAND*. The purpose of the *REDUCE* operation is to construct a Gaussian pyramid from the base layer to the top layer and that of *EXPAND* is the inverse of the *REDUCE* operation and constructs a pyramid from the top layer to the base layer. Suppose that the original image represented by an array G_0 which contains M rows and N columns of pixels where each pixel represents the light intensity. Let G_0 be the bottom layer (i.e., layer 0) of the pyramid which we want to construct. Then the layer 1 which is immediate above layer 0, contains image G_1, that is a reduced-resolution or low-pass filtered version of G_0. Each value within layer 1 is computed as a weighted average

of values in layer 0 within a specified window, say, 5×5 matrix. Then each value in layer 2, representing G_2, is obtained by the same pattern of weights from layer 1, etc. Generally, the layer-to-layer up averaging processing is performed by

$$G_k = REDUCE[G_{k-1}]$$

where for each node (i,j), $0 \leq i < M_k$, $0 \leq j < N_k$,

$$G_k(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) G_{k-1}(2i+m, 2j+n)$$

and M_k, N_k are rows and columns of G_k. An example of such construction with $2^8 = 256$ gray levels is given by Figure 22 above.

On the other hand, the layer-to-layer down averaging processing can be done by the *EXPAND* operation as follows. Let $G_{k,l}$ be the result of expanding G_k l times. Then $G_{k,0} = G_k$ and

$$G_{k,l} = EXPAND[G_{k,l-1}]$$

where for each node (i,j) and $0 < l \leq N$, $0 \leq i < M_k$, $0 \leq j < N_k$,

$$G_{k,l}(i,j) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) G_{k,l-1}(\frac{i-m}{2}, \frac{j-n}{2})$$

and M_k, N_k are rows and columns of G_k respectively.

(It should be noted that the values of $(i-m)/2$ and $(j-n)/2$ are limited to integers and if we apply the *EXPAND* operation k times to image G_k we obtain $G_{k,k}$ which is the same size as the original image G_0.)

Now if we start off with the value $L_N = G_N$ at the node of the top layer and for each layer $0 \leq l < N$, let

$$L_l = G_l - EXPAND[G_{l+1}],$$

then the above described layer-to-layer down averaging processing produces a Laplacian pyramid and the original image can be obtained by $G_0 = \sum_{l=1}^{N} L_l$. Note that because two layers differ in their sample density, it is necessary to interpolate new sample values between those in a given layer before that layer is subtracted from the next-lower layer. Such interpolation can be achieved by the *EXPAND* operation. The value L_l represents the new information which can not be obtained by the interpolation *EXPAND*.

4.3 Comparative Study Between MPCM and Burt-Adelson's Pyramid Method

In this section, we make a comparative study of MPCM and Burt-Adelson's pyramid coding method. Comparing Figure 22 to Figure 23 described in the next page, there are several distinctly different features between these two approaches both in structure and ideas.

(1) In Figure 12, MPCM generates a sequence of image components x_k at the k-th stage based on the k-th tier in a gray-level triangle, while a bottom-up pyramid method produces an image pyramid from fine-to-coarse using various window functions. (2) As shown in Figure 12, each tier of a top-down gray-level triangle represents a stage specified by a stage level which will be added

to the set of gray levels used in the previous stage. As a result, the number of gray levels is always increased by 1 from one tier to the next tier. Consequently, it requires different bit rates for coding image components in various stages. The lower the tier, the more gray levels, thus, the more bits are required. However, in pyramid methods, all layers of a bottom-up image pyramid require the same bit rate. (3) The number of tiers for a top-down gray-level triangle is determined by the number of bits to be used for coding. This is different from pyramid coding where the number of layers of a bottom-up image pyramid is independent of bit rates but determined by the window size used for masking. This can be seen from Figures 11 and 12. (4) The sizes of image components generated by MPCM are the same, however, the sizes of images in a Gaussian pyramid produced by pyramid coding are reduced by a scale, e.g., the image size in Figure 11 is reduced by 2 and the image size in Figure 12 is fixed at a constant 256×256. (5) The quality of images generated by a top-down gray-level triangle is gradually improved as tiers go down. It is completely opposite for a bottom-up image pyramid where the resolution is refined when layers move from top to bottom. (6) MPCM can achieve bit-saving by processing the first few tiers of a top-down gray-level triangle and still produces a satisfactory result. A bottom-up image pyramid cannot do it because all layers of a bottom-up image pyramid require the same bit rate. (7) The stage level associated with each tier in MPCM plays the same role as that of each layer in Laplacian pyramid. That is, a stage level always represents new information not contained in previous tiers which is the same for a layer of the Laplacian pyramid.

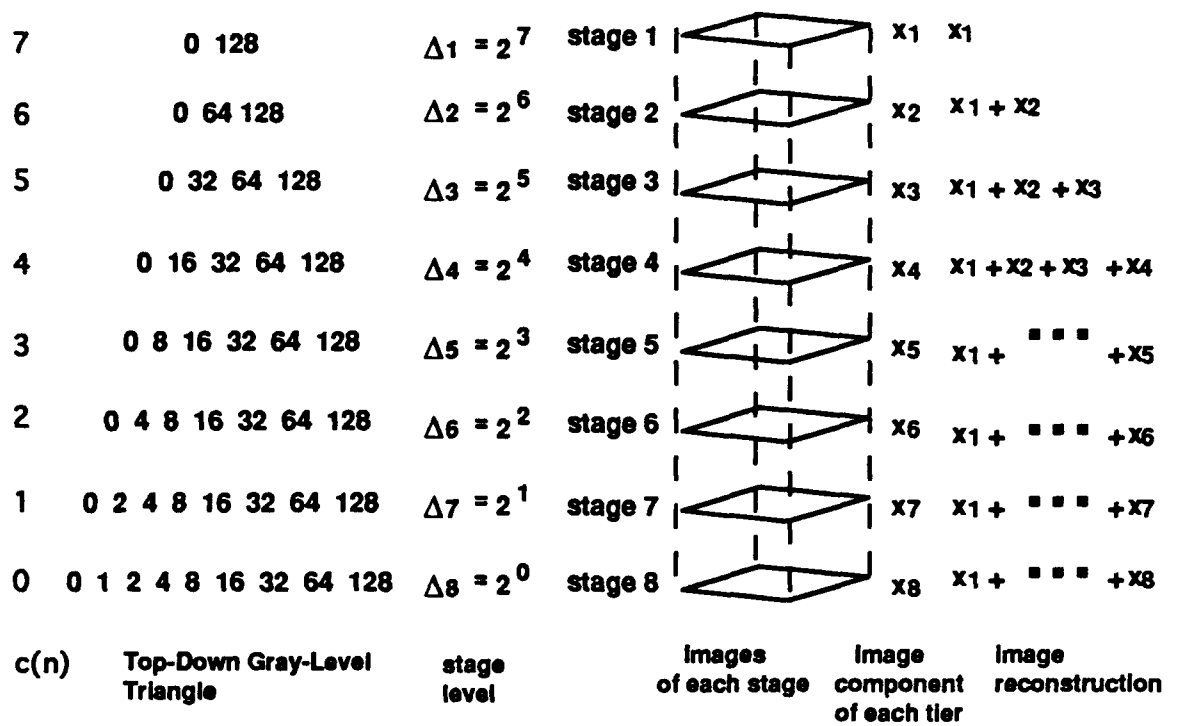


Figure 23: Implementation of 8-Stage Top-Down Gray-Level Triangle

In the following section, several experiments are studied and compared to Burt-Adelson's Lapla-

cian bottom-up pyramid coding method.

4.4 Experimental Results

The experiments are conducted based on MPCM and Burt-Adelson's pyramid coding method with the window function $w(m,n)$ chosen to be separable, namely, $w(m,n) = w(m)w(n)$ with the following conditions,

$$w(0) = a, w(-1) = w(1) = b, w(-2) = w(2) = c, a + 2c = 2b,$$

where a is a free variable set to 0.5, $b = 0.25$, and $c = 0$.

Both MPCM and the Burt-Adelson pyramid coding generate images at 8 different resolutions. Figure 24 is the image decomposition of Lena by MPCM and Figure 25 is a sequence of image components determined by 7 stage levels, {1,2,4,8,16,32,64,128} where stage level 0 is not shown. Figure 26 contains 7 images obtained by summing successive image components in Figure 25. Figure 27 shows reconstructed images produced by MPCM. Figures 28 and 29 represent an 8-layer Gaussian pyramid and an 8-layer Laplacian pyramid respectively. Since the sizes of the images in the Gaussian pyramid are scaled down by 4 from one layer to the above layer, they must be expanded to the size of the original image prior to reconstruction. Figure 30 is the corresponding expanded images of the 8-layer Gaussian pyramid in Figure 28 based on the *EXPAND* operation defined in Section 4.2. By adding Figure 30 to the Laplacian pyramid in Figure 29 we produce Figure 31 which a sequence of 8 reconstructed images at different resolutions. Namely, $\mathrm{R}(i-1) = \mathrm{R}(i) + \mathrm{L}(i-1)$ for $i = 1, 2, \cdots, 8$ where R(0) is the original image; R(i) is the image reconstruction at layer i for $i = 1, \cdots, 8$; L(i) is the Laplacian pyramid at layer $i = 0, 1, \cdots, 7$; and L(8) is the expanded Gaussian pyramid at layer 8. By comparing Figure 31 to Figure 27, we can conclude that MPCM reconstructs better images than does the pyramid coding.

As demonstrated in experimental results, MPCM can generally do better than pyramid coding methods, which have shown to be promising techniques in image analysis.

References

1. A. Rosenfeld, ed., *Multiresolution Image Processing and Analysis*, Springer-Verlag, 1984.

2. P.J. Burt and E.H. Adelson, "Laplacian pyramid as a compact image code," *IEEE Trans. on Comm.*, vol. COM-31, Apr. 1983, pp. 532-540.

Text continues on page 51.

Figure 24: MPCM Decomposition of Lena

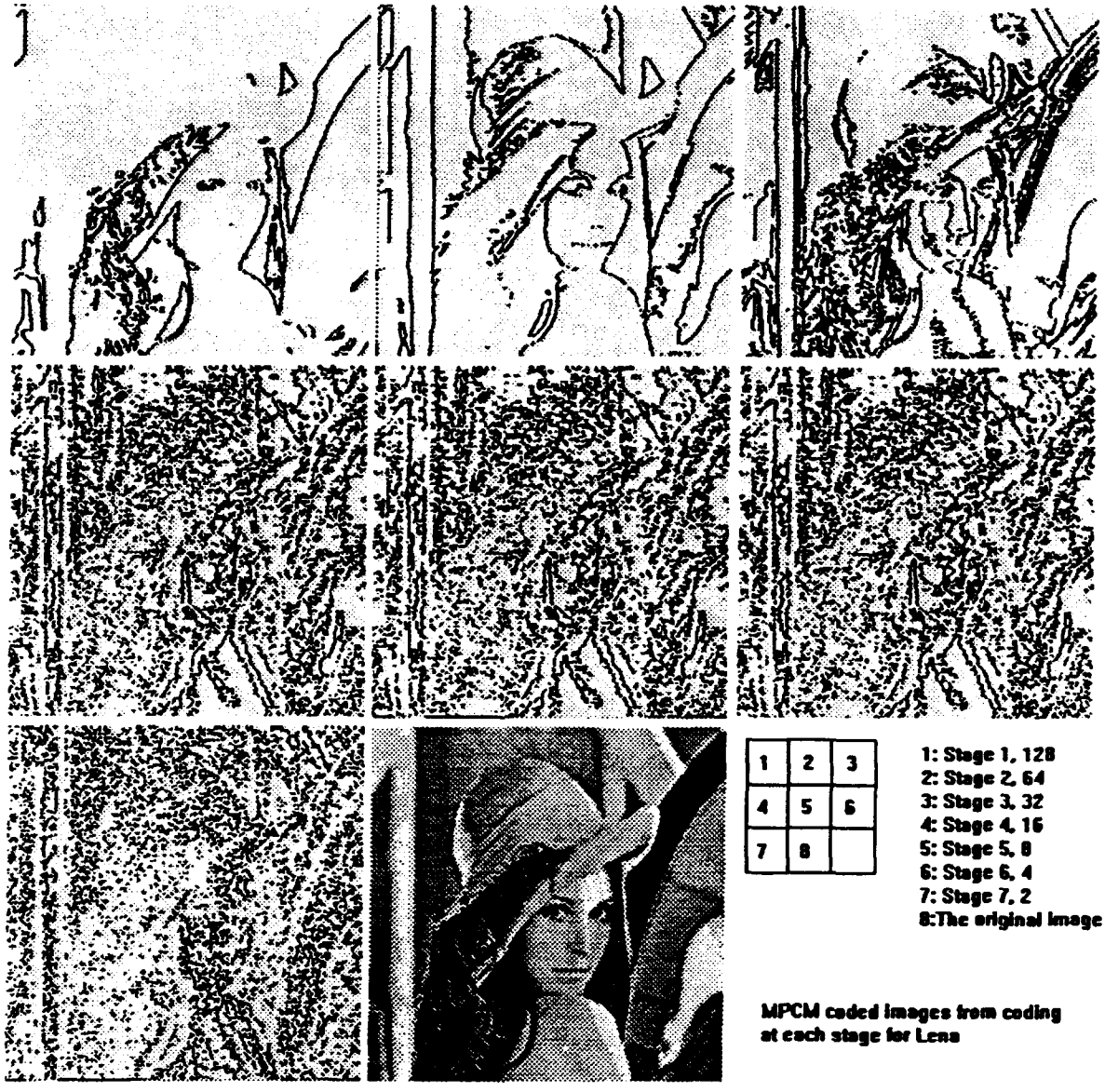


Figure 25: MPCM Coded Images of Lena

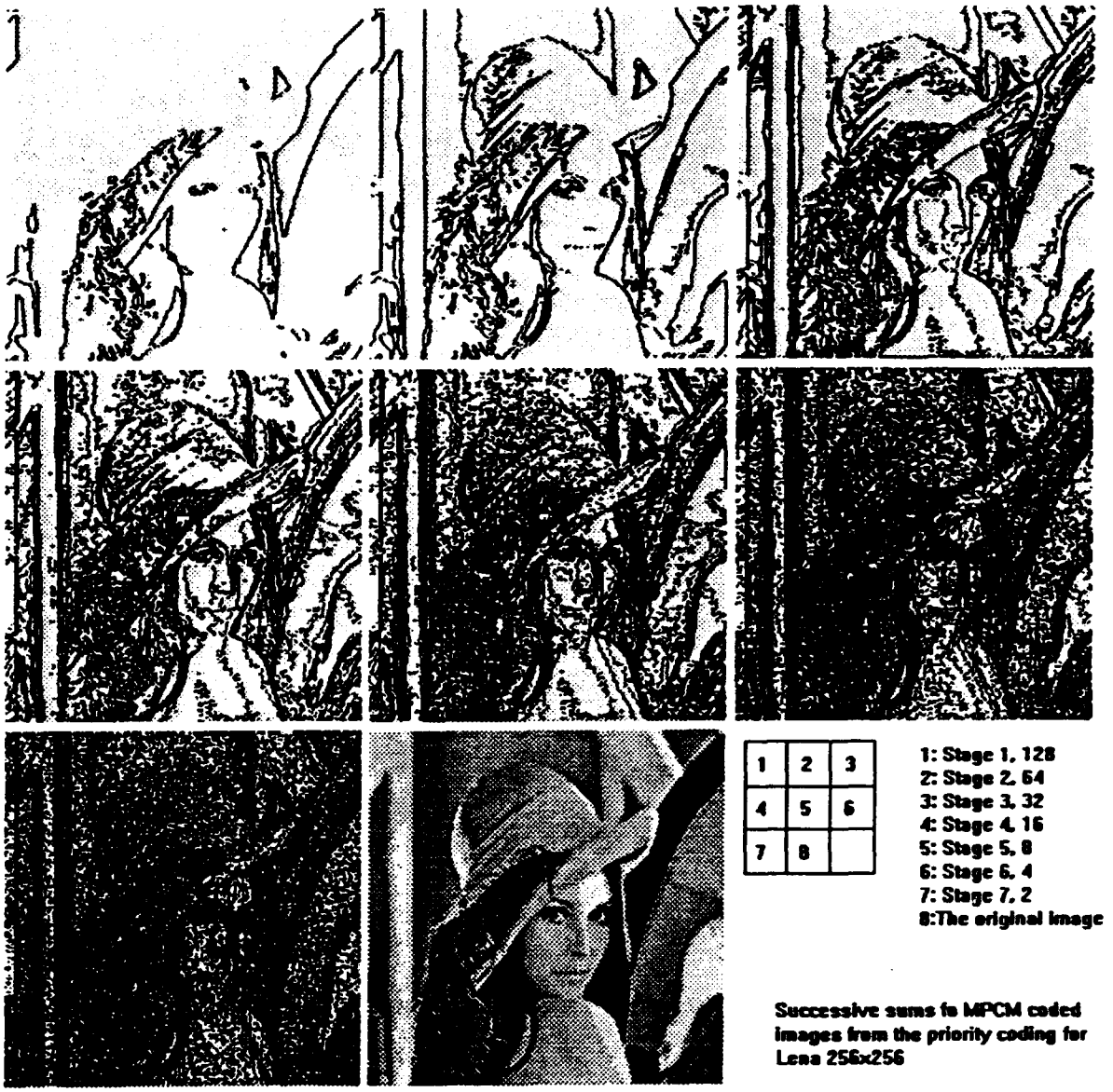


Figure 26: MPCM Successive Sums for Lena

Figure 27: MPCM Reconstruction for Lena

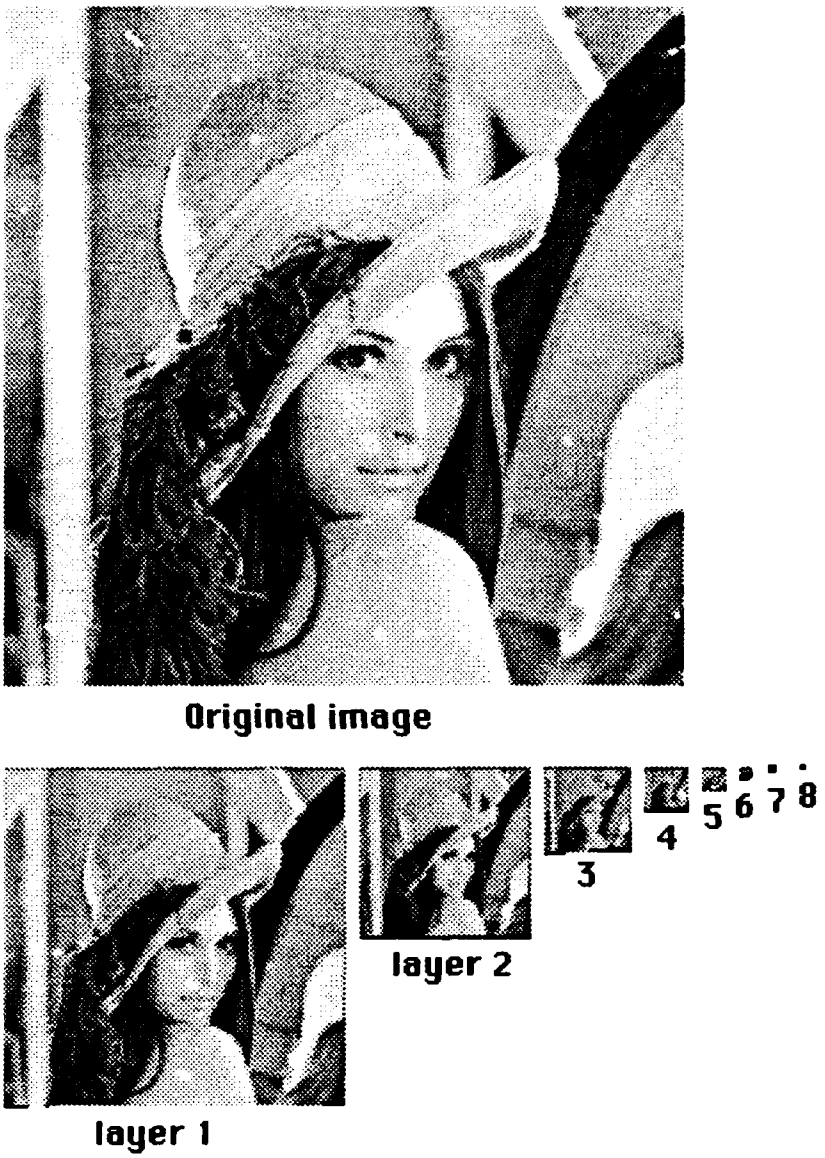


Figure 28: Gaussian pyramid a=0.5, b=0.25, c=0.0

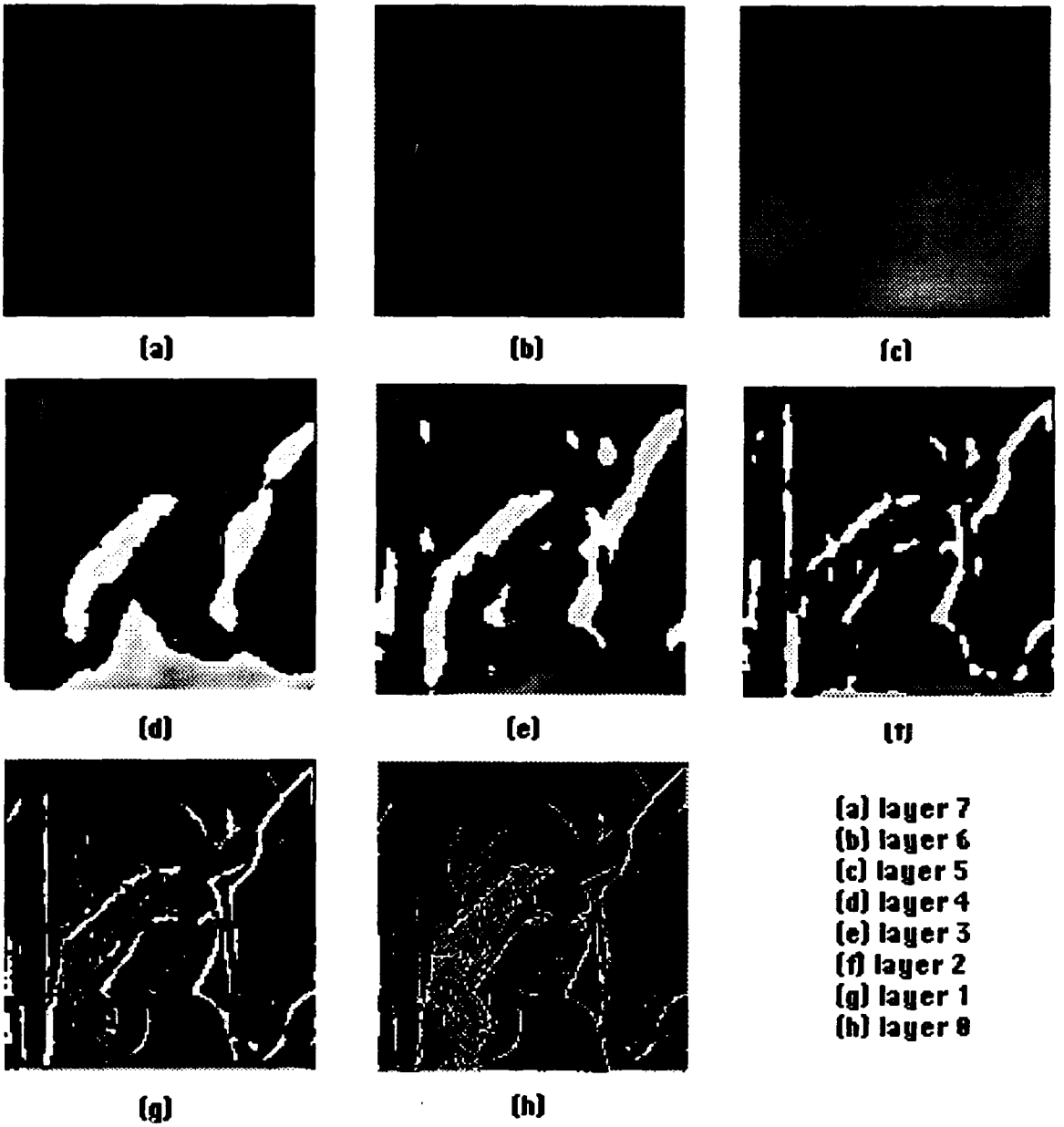


Figure 29: Laplacian pyramid

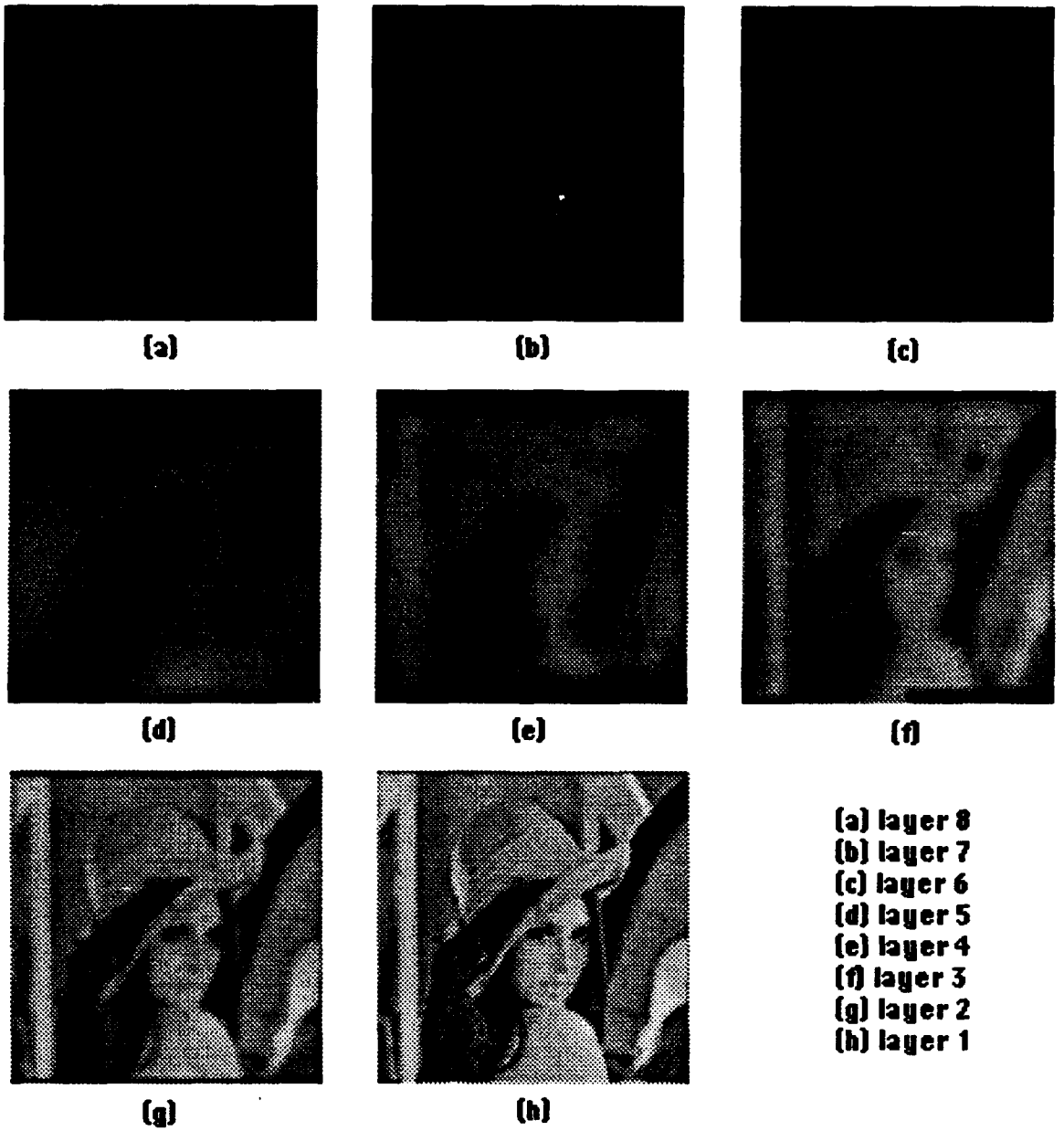


Figure 30: Expanded Gaussian Pyramid
a=0.5, b=0.25, c=0.0

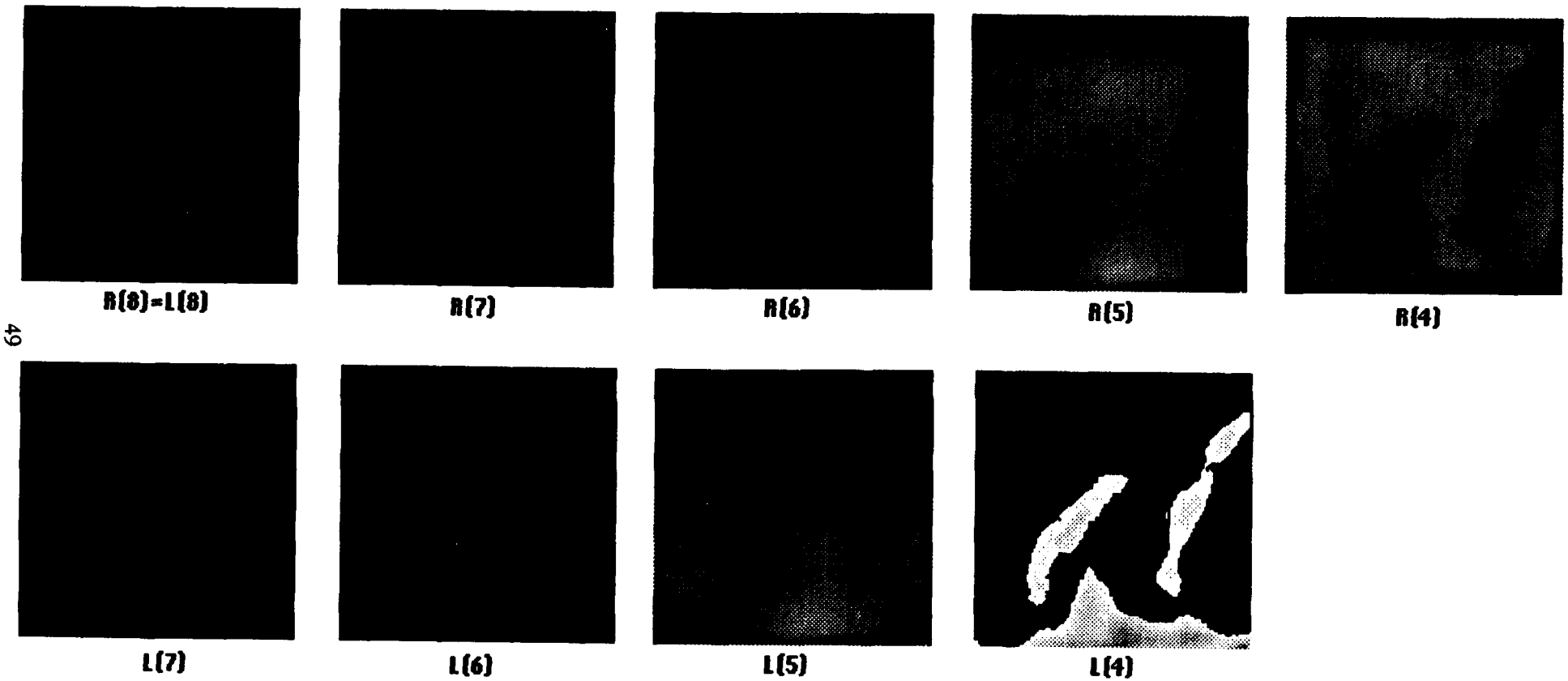
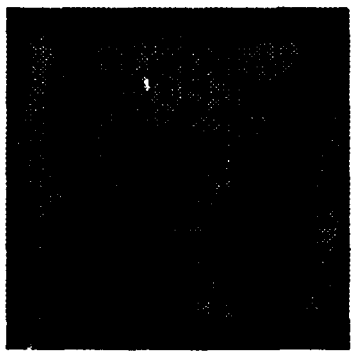


Figure 31: 8-layer reconstruction of Lena image

R(i) Reconstruction at layer i, for i=8,7,6,5,4;
L(i) Laplacian pyramid at layer i, for i=8,7,6,5,4;
R(i)+L(i-1)=R(i-1), i=8,7,6,5.



R(4)

R(3)

R(2)

R(1)

R(0)

L(3)

L(2)

L(1)

L(0)

Figure 31: 8-layer reconstruction of Lena image (Continued)

R(i) Reconstruction at layer i, for i=4,3,2,1;
L(i) Laplacian pyramid at layer i, for i=3,2,1,0;
R(0) Original image;
R(i)+L(i-1)=R(i-1), i=4,3,2,1.
(Continued)

CHAPTER 5

RELATIVE ENTROPY-BASED IMAGE THRESHOLDING

In Chapter 3, we have seen that MPCM employed a priority code to extract edges progressively. Since the primary role of the priority code is to detect edges and contours of objects, MPCM can be viewed as a multi-level object edge detection technique. Each level represents one stage corresponding to a thresholding scale. The levels, $\{\Delta_k\}_{k=1}^{M}$ selected by MPCM are preset prior to coding. Since no algorithms are required to generate the set of $\{\Delta_k\}_{k=1}^{M}$, these levels are neither adaptive nor optimal. However, the pay-off is that they are very easy to implement. In this chapter, we consider another distinct edge detection method, *relative entropy-based image thresholding technique* which will generate a better value for thresholding than that prefixed by MPCM.

The idea proposed to obtain a good quality thresholded image is to use relative entropy as a criterion to measure the discrepancy between the probability transition distributions of the original image and the thresholded image is proposed. A gray level which minimizes the relative entropy between the original image and the resulting bilevel image will be used to threshold the image. The relative entropy approach is different from existing entropy-based thresholding techniques. To see the performance we test different types of image and compare the results to those produced by the local entropy-based and joint entropy-based algorithms developed by Pal [1]. The experiments show that the new approach generally performs better than Pal, *et al.*'s entropy-based algorithms. In particular, the relative entropy-based algorithm possesses a very good capability for picking up edges, while Pal, *et al.*'s algorithms do not. In addition, another advantage of the relative entropy approach is the computational saving compared to Pal *et al.*'s algorithms, particularly, computational complexity is largely reduced.

5.1 Introduction

Gray level image thresholding is a simple image segmentation method to extract objects in an image from the background. It often represents a first step in image understanding. In an ideal image where objects are clearly distinguishable from the background, the gray level histogram of the image turns out to be bimodal. In this case, a best threshold segmenting objects from the background is one placed right in the valley of two peaks of the histogram. It is generally not true, however, that images to be segmented always have bimodal gray level histograms. Therefore, finding an appropriate threshold for images is not straightforward. Various thresholding techniques have been proposed to resolve this problem.

One approach is to reduce a multimodal problem to a bimodal problem in which case it requires some modification of the gray level histograms. This is usually done on the basis of geometric properties. Unfortunately, such modifications are either empirical or arbitrary without justification. Accordingly, there are no acceptable objective criteria to determine which thresholding technique is best. An alternative approach is to look into the information provided by gray levels rather than images themselves [1]. Along this direction, using Shannon's entropy concept as a measure of goodness for image thresholding has received considerable interest recently [1-6]. Although Pun [2-3] and Kapur, *et al.* [4] define the entropy of histogram differently, they basically assumed that an image can be completely characterized by its gray level histogram. Since they did not take into consideration the spatial correlation of neighboring pixels, which is very important for images,

particularly for highly correlated images such as medical images, there is room for improvement on Pun's and Kapur, *et al.*'s methods. In order to account for spatial dependency between pixels, it seems natural to extend Pun [2-3] and Kapur, *et al.*'s first-order entropy to a second-order entropy.

Several studies applying second-order entropy to image thresholding have been reported in [4-6]. Of particular interest is the use of a co-occurrence matrix to define second-order entropy. The co-occurrence matrix is basically a transition matrix generated by changes in pixel intensities. For any two arbitrary gray levels i and j (i, j are not necessarily distinct), the co-occurrence matrix describes all intensity transitions from gray level i to gray level j. By means of the co-occurrence matrix, two definitions were proposed for second-order entropy; local entropy and joint entropy [6]. Suppose that t is the desired threshold. t then classifies all pixels in an image into either background or objects where the background contains pixels with gray levels below or equal to t and the objects have gray levels above t. The two-region classification further divides the co-occurrence matrix into four quadrants which correspond respectively to transitions from background to background (BB), background to object (BO), object to background (OB) and object to object (OO). The local entropy is defined only on two quadrants, BB and OO, whereas the joint entropy is defined only on the other two quadrants, BO and OB. Based on these two definitions, Pal, *et al.* [1] developed two algorithms each of which maximizes local entropy and joint entropy respectively.

The relative entropy-based approach is different from those in [1-6]. Rather than maximizing entropies of background or object separately, we introduce the concept of relative entropy (or cross entropy, Kullback-Leiber's discrimination information and directed divergence) which has been widely used in source coding to measure discrepancy between two probability distributions. Since an image is a source and can be completely described by its probability distribution, the relative entropy can be interpreted as a measure of the distance between two images. To implement the relative entropy approach, one calculates the gray-level transition probability distributions of the co-occurrence matrices for an image to be segmented and a thresholded bilevel image, then determine which threshold minimizes the discrepancy between these two transition probability distributions, i.e., their relative entropy. The threshold which has the smallest relative entropy will be used to segment the original image. As a result, the thresholded bilevel image will be the best approximation of the original image. Because transitions of OB and BO generally represent edge changes in boundaries and transitions of BB and OO indicate local changes in regions, a thresholded bilevel image resulting from relative entropy thresholding must have the best transition match to that of the co-occurrence matrix of the original image.

As shown in experimental results, the relative entropy-based algorithm generally outperforms the local entropy-based and joint entropy-based algorithms. While the local entropy and joint entropy approaches perform similarly, the relative entropy approach produces a more pleasing result. In particular, the relative entropy approach is very good at picking up edges, which the other two are not. The experiments also show that in some images the local entropy approach may perform a little bit better than the joint entropy approach or possibly the other way around. There is no evidence to indicate which one is generally better than the other. However, among all images tested the relative entropy approach demonstrated unique capability and is preferred to the other two approaches. In addition, an advantage of the relative entropy approach is the computational saving compared to the local and joint entropy approaches, particularly, computational complexity is largely reduced.

5.2 Review of Previous Work

5.2.1 Co-occurrence Matrix

Given a digitized image of size $M \times N$ with L gray levels G={0, 1, 2, ..., $L-1$}, we denote $F=[f(x,y)]_{M\times N}$ to represent an image, where $f(x,y) \in G$ is the gray level of the pixel at the spatial location (x,y). A co-occurrence matrix of an image is an $L \times L$ dimensional matrix, $T = [t_{ij}]_{L\times L}$, which contains information regarding spatial dependency of gray levels in image F as well as the information about the number of transitions between two gray levels specified in a particular way. A widely used co-occurrence matrix is an asymmetric matrix which only considers the gray level transitions between two adjacent pixels, horizontally right and vertically below [1]. More specifically, let t_{ij} be the (i,j)th entry of the co-occurrence matrix T. Following the definition in [1],

$$t_{ij} = \sum_{l=1}^{M}\sum_{k=1}^{N} \delta(l,k) \tag{1}$$

where

$$\delta(l,k) = 1, \quad \text{if} \begin{cases} f(l,k) = i, \quad f(l,k+1) = j \\ \quad \text{and/or} \\ f(l,k) = i, \quad f(l+1,k) = j \end{cases}$$
$$\delta(l,k) = 0, \quad \text{otherwise}$$

One may like to make the co-occurrence matrix symmetric by considering horizontally right and left, and vertically above and below transitions. It, however, has been found [1] that including horizontally left and vertically above transitions does not provide more information about the matrix or significant improvement. Therefore, it is sufficient to consider adjacent pixels which are horizontally right and vertically below so that the required computation can be reduced.

Normalizing the total number of transitions in the co-occurrence matrix, we obtain the desired transition probability from gray level i to j [1] as follows.

$$p_{ij} = t_{ij} / \left(\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} t_{ij}\right) \tag{2}$$

5.2.2 Quadrants of Co-occurrence Matrix

Let $t \in G$ be a threshold of two groups (foreground and background) in an image. The co-occurrence matrix, T defined by (1) partitions the matrix into four quadrants, namely, A, B, C, and D, shown in Figure 32.

These four quadrants may be separated into two types. If we assume that pixels with gray levels above the threshold be assigned to the foreground (objects), and those below, assigned to the background, then, the quadrants A and C correspond to local transitions within background and foreground respectively; whereas, quadrants B and D represent transitions across the boundaries of background and foreground. The probabilities associated with each quadrant are then defined by

$$
\begin{aligned}
P_A(t) &= \sum_{i=0}^{t}\sum_{j=0}^{t} p_{ij} \\
P_B(t) &= \sum_{i=0}^{t}\sum_{j=t+1}^{L-1} p_{ij} \\
P_C(t) &= \sum_{i=t+1}^{L-1}\sum_{j=t+1}^{L-1} p_{ij} \\
P_D(t) &= \sum_{i=t+1}^{L-1}\sum_{j=0}^{t} p_{ij}
\end{aligned} \tag{3}
$$

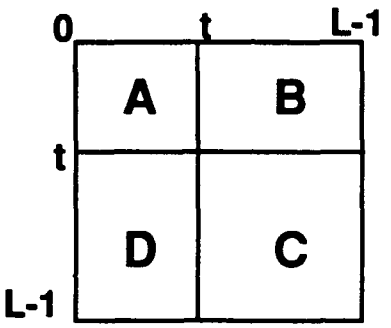


Figure 32: Quadrants of a Co-occurrence Matrix

The probabilities in each quadrant can be further defined by the so called "cell probabilities" and obtained as follows by normalization.

$$
\begin{aligned}
p_{ij}^{A} = p_{ij}/P_A &= \frac{t_{ij}/\left(\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} t_{ij}\right)}{\sum_{i=0}^{t}\sum_{j=0}^{t}\left(t_{ij}/\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} t_{ij}\right)} \\
&= \frac{t_{ij}}{\sum_{i=0}^{t}\sum_{j=0}^{t} t_{ij}}, \quad \text{for } 0 \le i \le t,\ 0 \le j \le t
\end{aligned} \tag{4}
$$

$$
p_{ij}^{B} = p_{ij}/P_B = \frac{t_{ij}}{\sum_{i=0}^{t}\sum_{j=t+1}^{L-1} t_{ij}}, \quad \text{for } 0 \le i \le t,\ t+1 \le j \le L-1 \tag{5}
$$

$$
p_{ij}^{C} = p_{ij}/P_C = \frac{t_{ij}}{\sum_{i=t+1}^{L-1}\sum_{j=t+1}^{L-1} t_{ij}}, \quad \text{for } t+1 \le i \le L-1,\ t+1 \le j \le L-1 \tag{6}
$$

$$
p_{ij}^{D} = p_{ij}/P_D = \frac{t_{ij}}{\sum_{i=t+1}^{L-1}\sum_{j=0}^{t} t_{ij}}, \quad \text{for } t+1 \le i \le L-1,\ 0 \le j \le t \tag{7}
$$

5.2.3 Algorithms Developed by Pal, et al. [1]

The algorithms suggested by N.R. Pal and S.K. Pal in [1] attempted to take advantage of spatial correlation in an image. By doing so, Pal and Pal introduced two concepts of second-order entropy based on Eqs. (4-7), which are called local entropy and joint entropy.

Since quadrant A and quadrant C reflect the local transitions from background to background (BB), and object to object (OO), they defined local entropy of background and local entropy of object by $H_B(t)$ and $H_O(t)$ respectively as follows.

$$H_B^{(2)}(t) = -\frac{1}{2}\sum_{i=0}^{t}\sum_{j=0}^{t} p_{ij}^A \log p_{ij}^A \tag{8}$$

$$H_O^{(2)}(t) = -\frac{1}{2}\sum_{i=t+1}^{L-1}\sum_{j=t+1}^{L-1} p_{ij}^C \log p_{ij}^C \tag{9}$$

It should be noted that (8) and (9) are determined by the threshold t, thus they are functions of t.

By summing up the local entropies of the object and the background, the second-order local entropy can be obtained by

$$H_{local}^{(2)}(t) = H_A^{(2)}(t) + H_C^{(2)}(t). \tag{10}$$

The algorithm proposed by Pal and Pal in [1] is one to select a threshold which maximizes the $H_{local}^{(2)}$ over t. In this paper, it will be called the local entropy-based algorithm.

Alternatively, quadrant B and quadrant D provides edge information on transitions from background to object (BO) and object to background (OB). In analogy with the local entropy defined above, another second-order joint entropy of the background and the object was also defined in [1] and given as follows by averaging the entropy $H(B;O)$ resulting from quadrant B, and the entropy $H(O;B)$ from quadrant D.

$$\begin{aligned} H_{joint}^{(2)}(t) &= (H(B;O) + H(O;B))/2 \\ &= -(\sum_{i=0}^{t}\sum_{j=t+1}^{L-1} p_{ij}^B \log p_{ij}^B + \sum_{i=t+1}^{L-1}\sum_{j=0}^{t} p_{ij}^D \log p_{ij}^D)/2 \end{aligned} \tag{11}$$

The algorithm maximizing (11) is called the joint entropy-based algorithm which is the second algorithm developed by Pal and Pal in [1].

5.3 Relative Entropy-Based Approach

5.3.1 Definition of Relative Entropy

Let S be an L-symbol source and p_j and p'_j be two probability distributions defined on S. The relative entropy between p and p' (or equivalently, the entropy of p relative to p') is defined by

$$L(p;p') = \sum_{j=0}^{L-1} p_j \log \frac{p_j}{p'_j}. \tag{12}$$

Equation (12) was first introduced by Kullback [7] as a distance measure between two probability distributions and later was found very useful in many applications [8-12]. Since the information

contained in an image source can be fully described by its entropy, which in turn can be completely characterized by source symbol probabilities, the relative entropy defined by (12) basically provides a criterion to measure the discrepancy between two images determined by probability distributions p_j and p'_j respectively. It is natural to use relative entropy as a measure of difference between an image and its segmented image, in our case, a bilevel thresholded image. There are several synonyms of relative entropy, e.g., cross entropy, Kullback-Leiber's discrimination information and directed divergence.

In order to obtain a good quality bilevel image, our aim is to find a threshold to segment an image such that the resulting thresholded bilevel image will best match the original image. Using the measure of relative entropy, one can choose the threshold t in such a manner that the gray level probability distribution p'_j of the thresholded image has minimum relative entropy $L(p;p')$ with respect to the original image. More specifically, the desired threshold t minimizes the discrepancy between p, and p' where p and p' are the gray level probability distributions of the original image and the resulting thresholded image respectively.

5.3.2 Joint Relative Entropy-Based Method

As indicated previously, a thresholding method based on first-order statistics of an image does not consider spatial correlation of an image. Therefore, exploiting the spatial dependency of the pixel values in the image can help determine a good threshold. It seems reasonable to extend the first-order relative entropy to a second-order joint relative entropy between p_{ij} and p'_{ij} where p_{ij} and p'_{ij} are the transition probability distributions of the co-occurrence matrices defined by eqs. (1-2) generated by the original image and the thresholded image respectively. Since transition probability distributions defined by the co-occurrence matrix contain the spatial information which reflects homogeneity within groups (quadrant A and quadrant C in Figure 1), and changes across boundaries (quadrant B and D in Figure 32), one can envision that a better result may be obtained if we choose the thresholded bilevel image to be the one which has the best transition match to that of the original image in terms of relative entropy.

Let the joint relative entropy of the probability distributions p_{ij} and p'_{ij} be defined by:

$$L(p;p') = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_{ij} \log \frac{p_{ij}}{p'_{ij}} \tag{13}$$

where p_{ij} and p'_{ij} are the transition probabilities from gray level i to gray level j of the original image and the bilevel image respectively. Minimizing $L(p;p')$ over t generally renders a bilevel image which best matches the original image.

It should be noted that when we threshold an image, we basically assign all gray levels in an original image to either 0 or 1 which corresponds to background or objects. As a result, there are only two gray levels in the thresholded image. The indices i, j used in the transition probability p'_{ij} still refer to those of gray levels of the original image. In addition, the statistics of pixels not adjacent to one another could also be considered, but the estimation of probabilities for such cases would be very difficult.

5.3.3 Co-occurrence Matrix of a Thresholded Bilevel Image

Let us assume that t is the selected threshold. By assigning 1 to all gray levels above threshold t, $G_1 = \{t+1, \ldots, L-1\}$ and 0 to all gray levels below t, $G_2 = \{0, 1, \ldots, t\}$, we obtain a bilevel image. It should be noted that the gray levels in G_1 will be treated equally likely in probability, so are gray levels in G_2. Consequently, the p'_{ij} can be found as follows (see Figure 32).

$$p'^{(A)}_{ij}(t) = q_A(t) = \frac{P_A(t)}{(t+1)\times(t+1)}; \text{ for } 0 \le i \le t, \tag{14}$$

$$p'^{(B)}_{ij}(t) = q_B(t) = \frac{P_B(t)}{(t+1)\times(L-t-1)}; \text{ for } 0 \le i \le t,\ t+1 \le j \le L-1 \tag{15}$$

$$p'^{(C)}_{ij}(t) = q_C(t) = \frac{P_C(t)}{(L-t-1)\times(L-t-1)}; \text{ for } t+1 \le i \le L-1,\ t+1 \le j \le L-1 \tag{16}$$

$$p'^{(D)}_{ij}(t) = q_D(t) = \frac{P_D(t)}{(L-t-1)\times(t+1)}; \text{ for } t+1 \le i \le L-1,\ 0 \le j \le t \tag{17}$$

where, $P_A(t)$, $P_B(t)$, $P_C(t)$, and $P_D(t)$ are defined by (3). For each selected t, $p'^{(A)}_{ij}(t)$, $p'^{(B)}_{ij}(t)$, $p'^{(C)}_{ij}(t)$, and $p'^{(D)}_{ij}(t)$ are constants in each individual quadrant and only depend upon the quadrant to which they belong. Therefore, we can denote them by $q_A(t)$, $q_B(t)$, $q_C(t)$, and $q_D(t)$ respectively.

5.3.4 Relative Entropy-Based Algorithm

By expanding Equation 13, we have:

$$\begin{aligned} L(p;p') &= \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} p_{ij}\log\frac{p_{ij}}{p'_{ij}} \\ &= \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} p_{ij}\log p_{ij} - \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} p_{ij}\log p'_{ij} \end{aligned} \tag{18}$$

Because the first term in the above equation is independent of the threshold t, minimizing the relative entropy described by Equation 13 is equivalent to maximizing the second term of Equation 18.

We can even further simplify the second term of the right side of the equation 18 as follows.

$$\begin{aligned} \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} p_{ij}\log p'_{ij} &= \sum_{A} p_{ij}\log q_A(t) + \sum_{B} p_{ij}\log q_B(t) + \\ &\quad \sum_{C} p_{ij}\log q_C(t) + \sum_{D} p_{ij}\log q_D(t) \\ &= P_A(t)\log q_A(t) + P_B(t)\log q_B(t) + \\ &\quad P_C(t)\log q_C(t) + P_D(t)\log q_D(t) \end{aligned} \tag{19}$$

This implies that in order to obtain a desirable threshold for classifying the object from the background, we need only maximize the last expression in Eq. 19 over t. The expression consists of four terms only, each of which is a product of P_i and $\log_i(t)$ for $i = A, B, C, D$. In comparison with Eq. 19 and Eq. 10 required for the local entropy and Eq. 11 for the joint entropy, the computational load for the relative entropy is significantly reduced. From Eqs. (8) and (9), $(t+1)^2 + (L-t)^2$ multiplications are required to calculate $p_{ij}\log p_{ij}$ for finding the local entropy and from Eq. (11), $2(t+1)(L-t)$ multiplications for the joint entropy. It is, however, that only 4 multiplications and 4 divisions are needed in Eq. (19) for the relative entropy. As a result, computational saving can be tremendous when the size of an image is very large.

5.4 Experimental Results

In order to see performance of the relative entropy-based thresholding method we tested a set of various images including a medical image and a thermal image. As shown in experiments, the relative entropy approach provides a very efficient and effective image thresholding tool. All test images have 256 gray levels. Figure 33(a) represents an original image of peppers and Figure 33(b)-(d) show the thresholded images obtained based on the local entropy and the joint entropy and the relative entropy approaches respectively. Figure 33 shows that the relative entropy based approach performs better than the joint entropy and as well as the local entropy. In this image the local entropy method produces a better result than does the joint entropy method.

Figure 34(a) is the original digitized image of an F-16 jet. As shown in Figure 34(b-d), the relative entropy technique produces a better result than both the local and joint entropy approaches. This can be seen from the "US AIR FORCE" logo. Interestingly, unlike the previous image of peppers, the joint entropy method generates better image quality than the local entropy. Figures 2 and 3 show that there is no absolute preference between the local entropy and joint entropy approaches. However, our relative entropy approach performs quite well and produces satisfactory results on both images. Figure 35 is an image of a couple. Evidently, the image thresholded by the relative entropy is much better than those by local and joint entropies.

Most surprising is Figure 36 which is a building image. The local and joint entropy approaches give similar thresholded images, Figure 36(b) and 36(c), however, both fail to pick up edges of the building, particularly, they miss all detail of the stairs in the front of the building. In this test image, the relative entropy approach not only gives details of stairs including the building front but also edges out the building, specifically the middle edges of the building which are not shown in Figure 36(b) and 36(c). The reason for this we believe is that the relative entropy can best match all possible transitions made from one gray level to another. The capability of the relative entropy approach for finding edges is again demonstrated in Figure 37 where a part of the open edge of the coffee cup is missing in Figures 37(b) and 37(c). However, Figure 37(d) picks up all open edge of the cup. Figures 38 is an image of two jets taken by a thermal imager. For this two-jet image, the relative entropy gives more details such as clouds. Two more experiments were discussed in [13].

Through the above experiments, it is demonstrated that the relative entropy-based approach performs quite differently from the local and joint entropy-based methods. More importantly, these experiments show that using relative entropy as a criterion for thresholding results in better understanding of images than the other two entropy-based approaches.

References

[1] Nikhil R. Pal and Sankar K. Pal, "Entropy thresholding," *Signal Processing*, vol. 16, pp.97-108, 1989.

[2] T. Pun, "A new method for grey-level picture thresholding using the entropy of the histogram," *Signal Processing*, Vol. 2, 1980, pp. 223-237.

[3] T. Pun, "Entropic thresholding, a new approach," *Comput., Graphics and Image Proc.*, Vol.16, pp.210-239, 1981.

[4] J.N. Kapur, P.K. Sahoo and A.K.C. Wong, "A new method for grey-level picture thresholding using the entropy of the histogram," *Comp. Graphics, Vision and Image Proc.*, vol. 29, pp. 273-285, 1985.

[5] Ahmed S. Abutaleb, "Automatic thresholding of gray-level pictures using two-dimensional entropy," *Comput. Vision Graph. Image Process.* vol. 47, pp. 22-32, 1989.

[6] Nikhil R. Pal and Sankar K.Pal, "Entropy: a new definition and its application," *IEEE Tran. Syst., Man, Cybern.*, vol. 21, No.5, pp. 1260-1270, 1991.

[7] S. Kullback, *Information Theory and Statistics*. New York:Dover, 1968.

[8] L.D. Davisson and A. Leon-Garcia, "A source matching approach to finding minimax codes," *IEEE Trans. on Information Theory*, vol. IT-26, Mar. 1980, pp. 166-174.

[9] C.-I Chang, S.C. Fan and L.D. Davisson, "On numerical methods of calculating the capacity of a continuous-input discrete-output memoryless channel," *Information and Computation*, vol. 86, no. 1 May 1990 pp. 1-13.

[10] C.-I Chang and L.B. Wolfe, "Source matching problems revisited," *IEEE Trans. on Inform. Theory*, vol. 38, no. 4, July 1992, pp. 1391-1395.

[11] L.B. Wolfe and C.-I Chang, "A simple method for calculating the rate distortion function of a source with an unknown parameter," accepted for publication in *Signal Processing*.

[12] C.-I Chang, L.C. Fan and L.D. Davisson, "Computation of the rate-distortion function of a source with uncertain statistics," *Proc. IEEE 1988 International Conference on Communication Systems*, Oct. 31-Nov. 3, 1988, Singapore, pp. 1180-1184.

[13] K. Chen, M.L.G. Althouse and C.-I Chang, "A relative entropy approach to image thresholding," Proceedings of 26th Annual Conference on Information Sciences and Systems, Princeton University, Princeton, N.J., March 18-20, 1992.

Text continues on page 66.

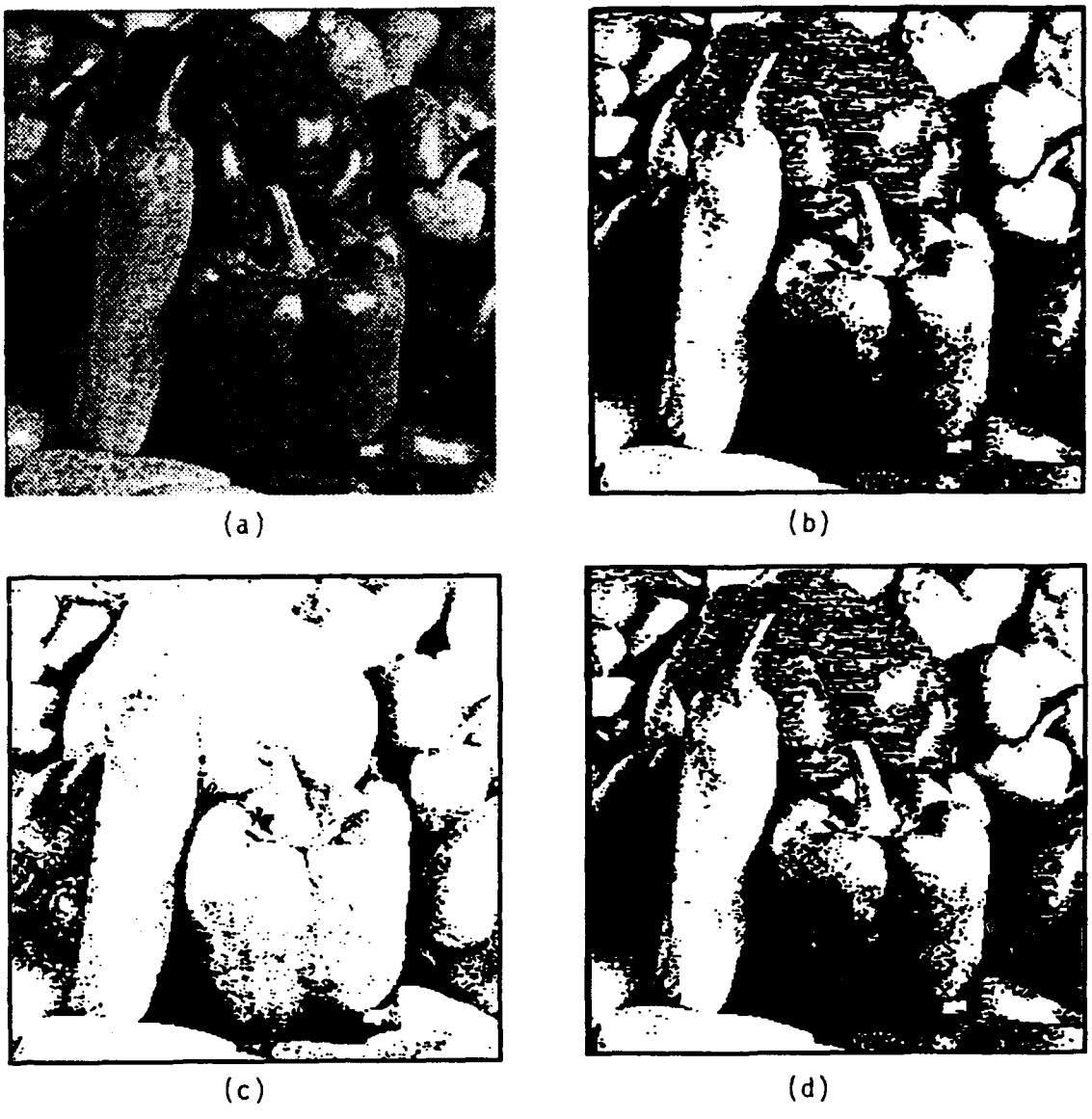


Figure 33: Peppers Image

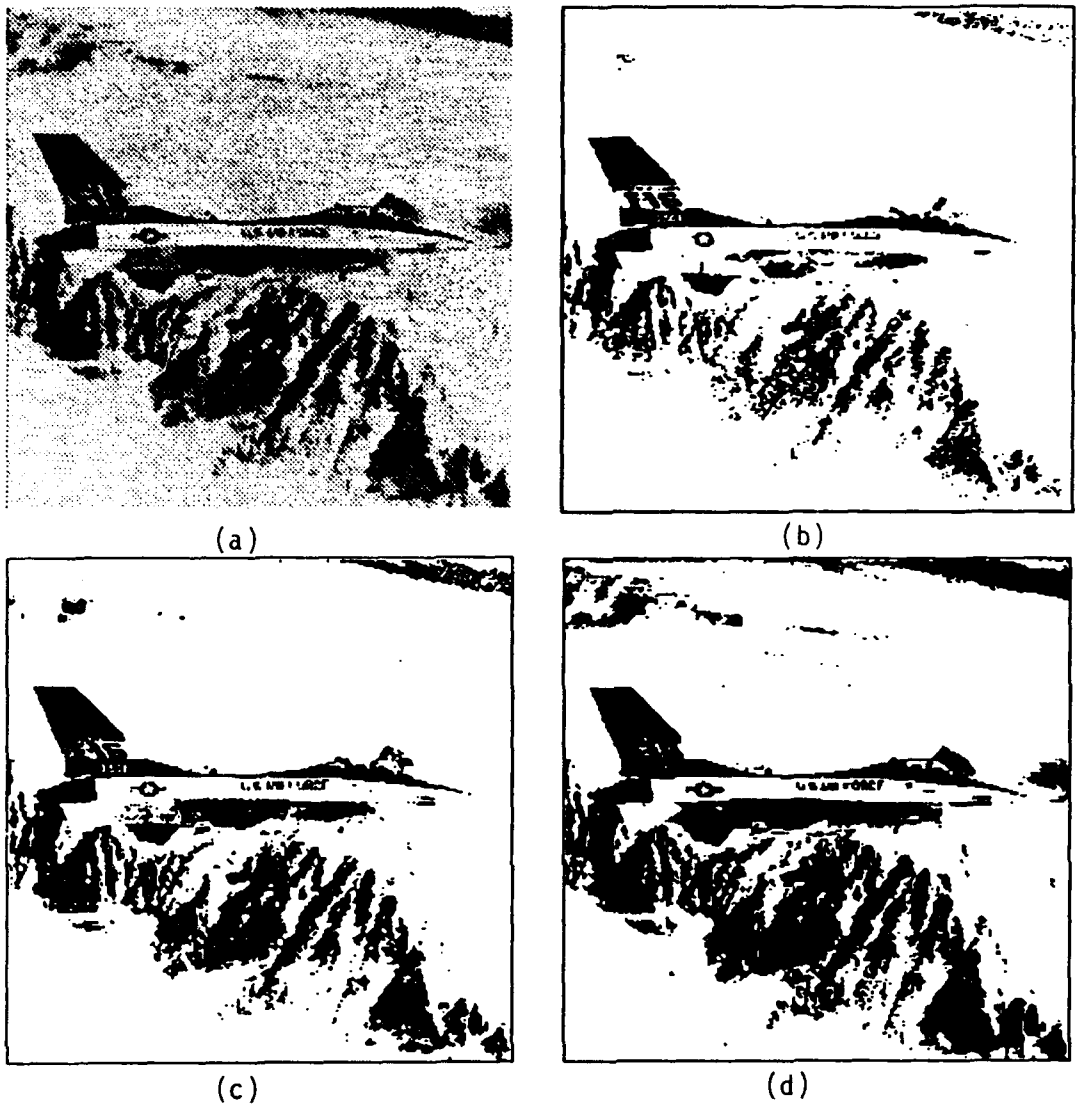
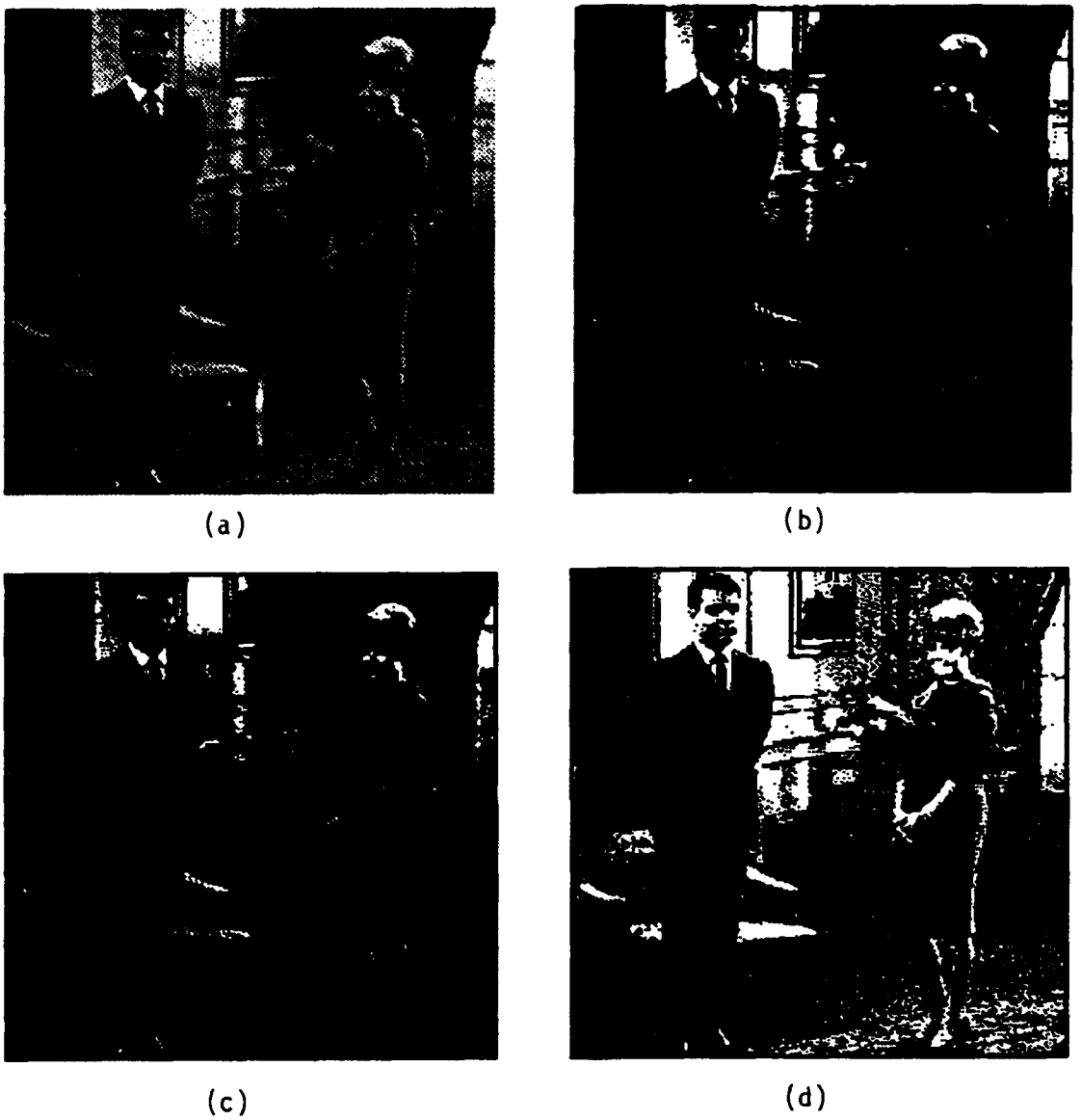


Figure 34: Air Force Jet Image



(a) (b)

(c) (d)

Figure 35: Couple Image

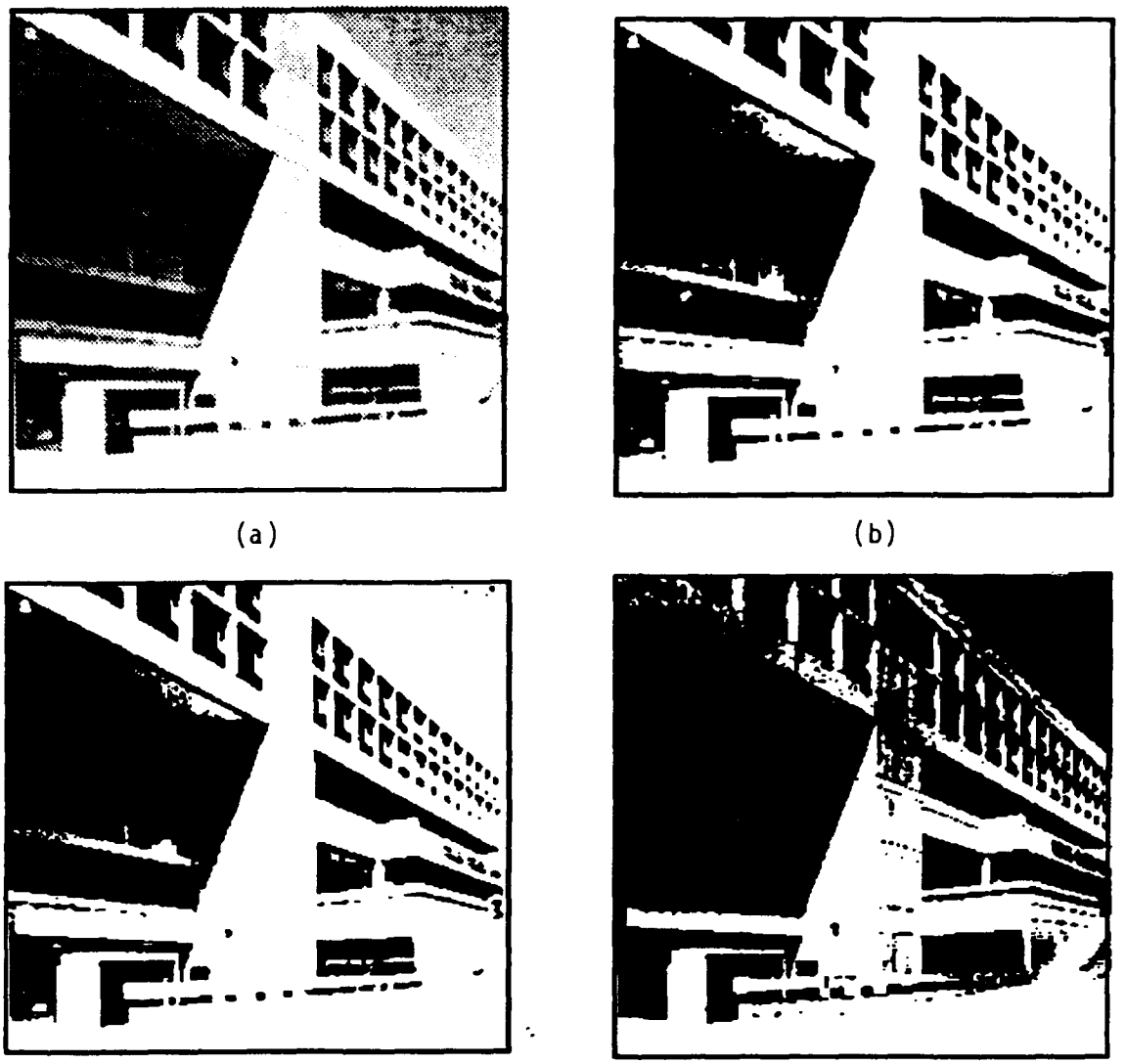


Figure 36: Building Image

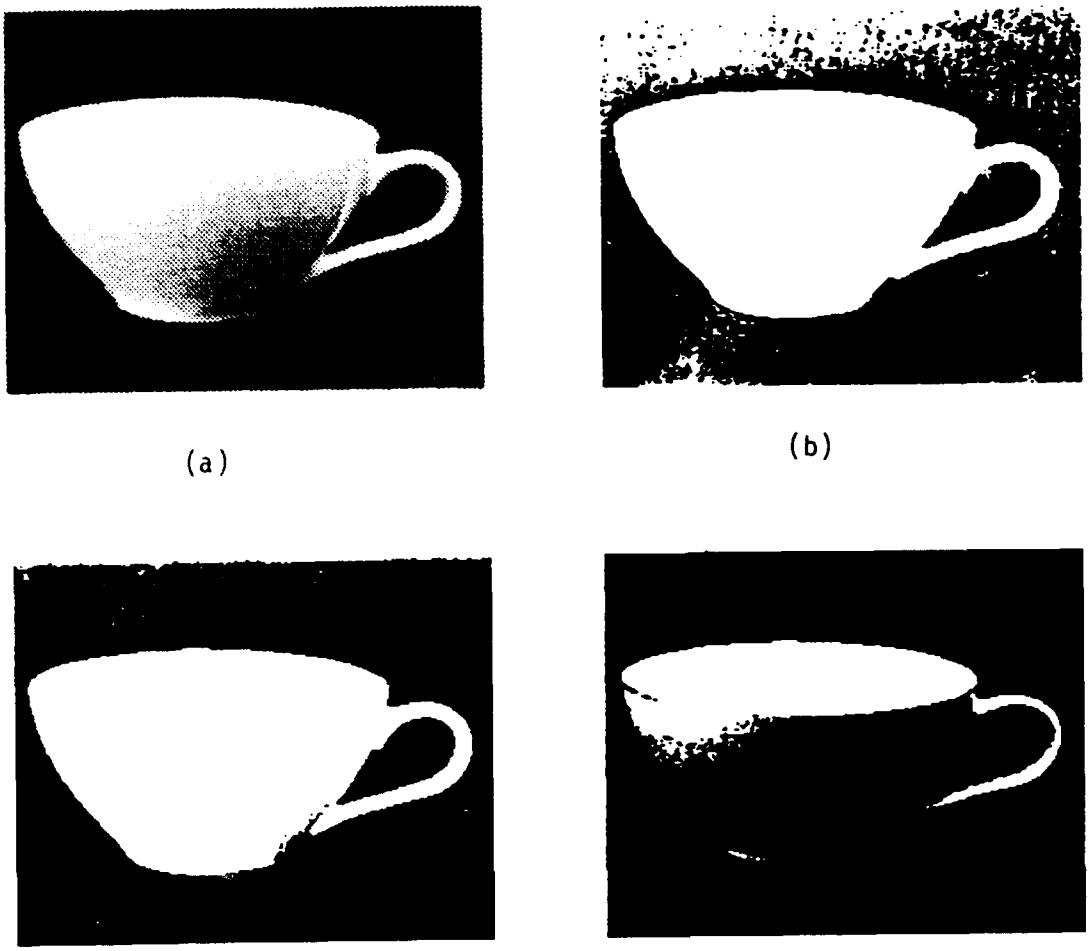


Figure 37: Teacup Image

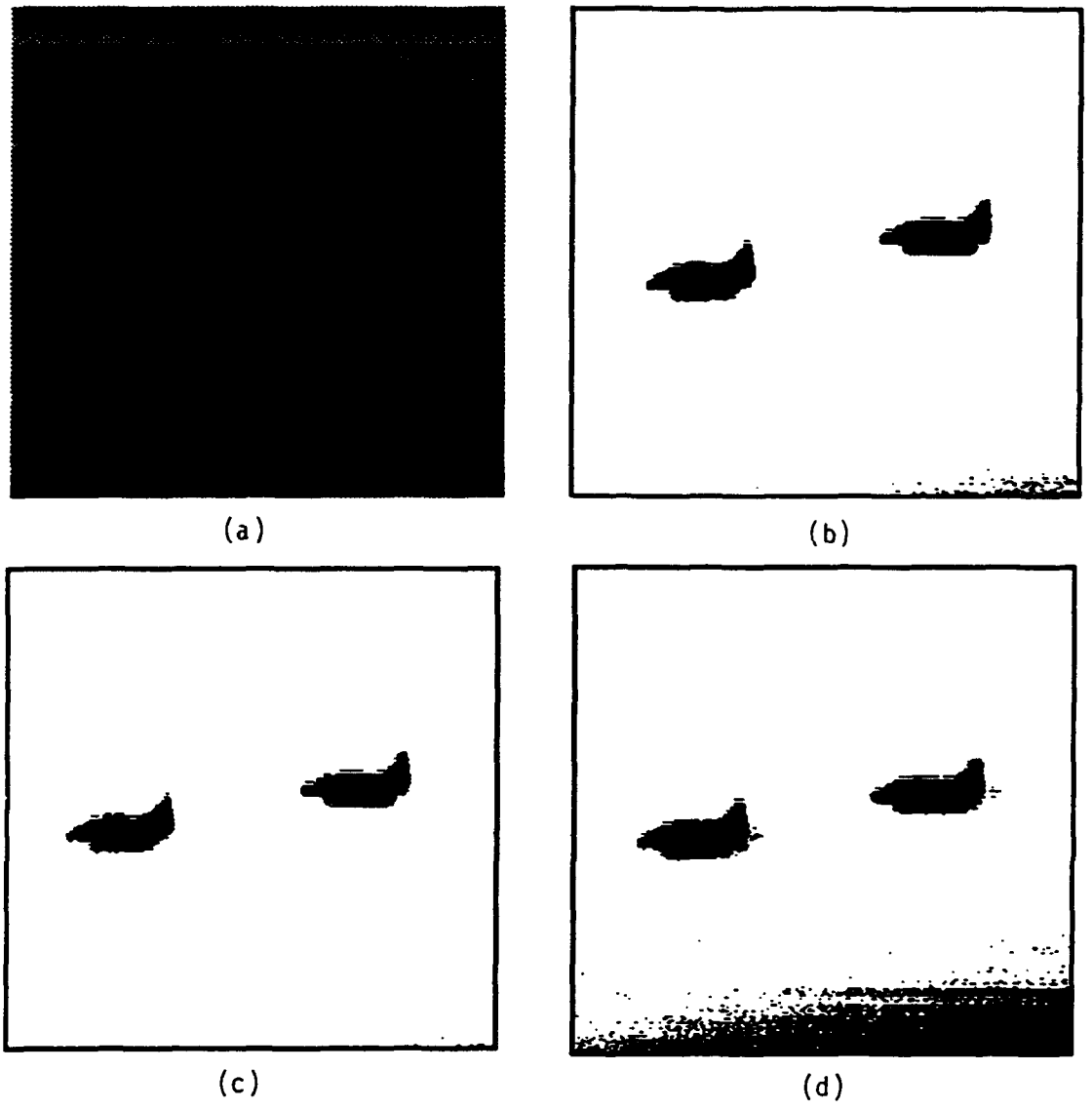


Figure 38: Jet Image

CHAPTER 6

CONCLUSIONS

This chapter concludes a summary of the work carried out during the period of September 1 1991-August 31 1992. Two new image edge extraction methods, "*MPCM*" and "*Relative Entropy-Based Thresholding*" were introduced in this report. The theory of MPCM is described in detail in Chapter 2 followed by two chapters on its applications. In Chapter 3, MPCM was suggested to be used as a progressive edge detection which shows great promise in image detection. Chapter 4 interpreted MPCM as a top-down gray-level triangle method which can be used as a multiresolution technique. Compared to widely used pyramid coding methods, MPCM provides an alternative approach to process images in multiple scales. The work based on this report was presented in various conferences [1-3].

In addition to image detection investigated in this report, some progress which continues the work in [4] has been made on chemical vapor detection for multispectral imagery and also presented in different conferences [5-7]. In order to make the report complete, the copies of these conference papers are included in this chapter for reference.

References

1. K. Chen, M.L.G. Althouse and C.-I Chang, "A relative entropy approach to image thresholding," *Proceedings of 26th Annual Conference on Information Sciences and Systems*, Princeton University, Princeton, N.J., March 1992, pp. 907-911.

2. C.-I Chang, Y. Cheng, M.L.G. Althouse, L. Zhang and J. Wang, "Multistage image coding: a top-down gray-level triangle method," to be presented and appear in *Proc. International Symposium on Spectral Sensing Research (ISSSR)*, Kauai, Hawaii, Sep. 15-20, 1992.

3. C.-I Chang and Y. Cheng and M.L.G. Althouse, "Chemical vapor detection using multistage predictive coding," to be presented and appear in *Proc. Scientific Conference on Chemical Defense Research*, CRDEC, Aberdeen Proving Ground, MD, Nov. 17-20, 1992.

4. M.L.G. Althouse and C.-I Chang, "Chemical vapor detection with a multispectral thermal imager," *Optical Engineering*, vol. 30, no. 11, November 1991, pp. 1725-1733.

5. M.L.G. Althouse, C.-I Chang and D. Smith, "Single frame multispectral thermal imagery," *Conference Proceedings SPIE*, vol. 1689, Orlando, Florida, April, 1992, pp. 20-24.

6. M.L.G. Althouse and C.-I Chang, "Chemical vapor detection and mapping with a multispectral thermal imager, " *Proc. 4th Int. Symp. Protection Against Chemical Warfare Agents*, Stockholm, June 1992. pp. 195-200.

7. M.L.G. Althouse and C.-I Chang, "Chemical vapor detection with multispectral thermal imagery," to be presented and appear in *Proc. International Symposium on Spectral Sensing Research (ISSSR)*, Kauai, Hawaii, Sep. 15-20, 1992.