TECHNICAL RESEARCH REPORT

An Advanced Image Coding Algorithm that Utilizes Shape-Adaptive DCT for Providing Access to Content

by R. Haridasan

CSHCN T.R. 97-5
(ISR T.R. 97-16)

The Center for Satellite and Hybrid Communication Networks is a NASA-sponsored Commercial Space Center also supported by the Department of Defense (DOD), industry, the State of Maryland, the University of Maryland and the Institute for Systems Research. This document is a technical report in the CSHCN series originating at the University of Maryland.

Web site http://www.isr.umd.edu/CSHCN/

An Advanced Image Coding Algorithm that Utilizes Shape-Adaptive DCT for Providing Access to Content

Radhakrishnan Haridasan
University of Maryland, College Park, MD.

This work was carried out under the supervision of
Kiran Challapali
Video Communications Department
Philips Research
Briarcliff Manor, NY 10510
This report has previously appeared as a Technical Report at Philips Research

February 18, 1997

Contents

1 Objective

To determine the efficiency of Shape-Adaptive Discrete Cosine Transform (SA-DCT) for coding arbitrarily shaped objects.

2 Background

2.1 MPEG-2: The current video coding standard

The Moving Picture Experts Group (MPEG) was formed under the auspices of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) in 1988 to establish standards for coding of moving pictures and associated audio for various applications such as digital storage media, distribution and communication [1][2]. After the success of the first generation video coding standard MPEG-1, which was optimized for progressive video at 1.5 Mbit/sec, MPEG-2 work item started to cover a wide range of applications including all-digital transmission of broadcast quality audio-video television signals. In addition to "coding tools" provided in MPEG-1, the MPEG-2 standard provides for SNR scalability, temporal and spatial scalability and also allows for data partitioning. These extensions are needed to address applications such as video telecommunications, video on Asynchronous Transfer Mode (ATM) networks, interworking of video standards, HDTV with embedded TV, etc. Adopted as the international standard in November 1994, MPEG-2 has become widely accepted for generic coding of video and audio worldwide.

The recent trend towards wireless communications, interactive computer applications and the integration of audio-visual data into an ever increasing number of applications is resulting in new expectations and requirements not addressed by current video coding algorithms. A whole spectrum of new applications including interactive mobile multimedia communications, videophone, multimedia electronic mail, remote sensing, electronic newspapers, interactive multimedia databases, multimedia videotex, games, interactive computer imagery and sign language captioning are envis-

aged for the future[3]. MPEG-2 was not designed to accommodate functionalities such as support for manipulation of the content of audio-visual data, or content-based database access useful for these new applications. Hence, there is a need to develop a new video coding standard suited for such applications.

2.2 MPEG-4: The future video coding standard

In the MPEG meeting in September 1993 it became clear that there needs to be a new work item to accommodate *novel and conventional functionalities* for video coding. MPEG-4's main focus shifted from low bit rate coding of video to deal with, in addition to compression, *content-based interactivity*, and *universal accessiblity* by wireless as well as wired networks. Among the functionalities designated as *content-based* were multimedia data access, manipulation, bitstream editing, and scalability. The remaining functionalities include hybrid natural and synthetic data coding, improved temporal random access, coding of multiple concurrent data streams, an example of which is stereoscopic coding, and robustness in error-prone environments.

To realize these functionalities, greater freedom to create a coding algorithm of choice is sought. This flexibility is provided by combining various coding tools and giving all the required rules to the decoder to decipher the coded bitstream. MPEG-4 would then find the most essential coding tools and standardize them ; making them "normative". It would also standardize a set of *flexible* rules that allow for various combinations of these tools, in order to accommodate the functionalities needed by particular applications. This would happen using a flexible syntax, in the form of *MPEG-4 Syntactic Description Language* (MSDL), which would be object-oriented. The MSDL would also be *extensible*, so as to allow for different combinations using these tools to be developed in the future.

To summarize, the basic elements of the proposed MPEG-4 standard to be finalized by 1997 would be:

- **Tools:** Techniques accessible through the syntax or described using the syntax (examples of

tools are motion compensation and contour representation).

- **Algorithms:** Organized collections of tools providing one or more functionalities (examples of algorithms are MPEG-2 video, MPEG-1 systems etc.).

- **Profiles:** Algorithms or combinations of algorithms constrained in a specific way to address a particular class of applications (an example is MPEG-2 MP@ML).

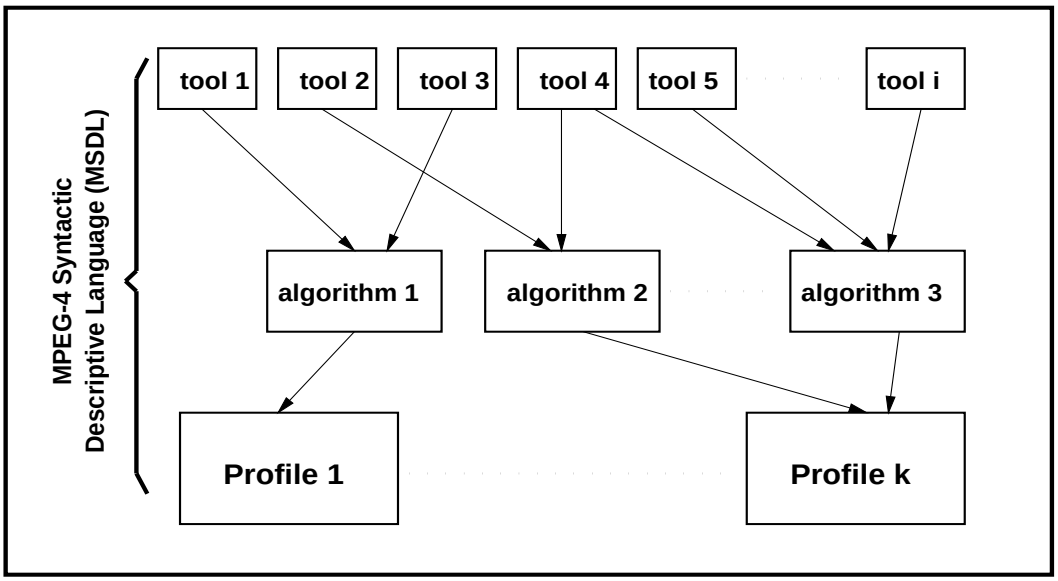

Figure 1: MPEG-4 Elements

2.3 Using an MPEG-2 like syntax for MPEG-4

As discussed in the preceding discussion, the key enabling technology of MPEG-4 applications is the ability to access objects that comprise an image. At present the entire image is treated as a single entity. In order to provide for the new functionality it is clear that arbitrarily shaped image segments should be encoded efficiently in a manner that each object can be decoded independently of other objects. An extension of MPEG-2 to incorporate *content-based addressability* is to use the Shape-Adaptive DCT in lieu of the standard block based DCT. This would provide for segment or object-based coding of images and video while providing backward compatibility to existing coding

standards. In section 5 and Appendix A, specific details for using MPEG-2 like syntax for MPEG-4 is provided.

3 Shape-Adaptive Discrete Cosine Transform (SA-DCT)

Shape-Adaptive DCT is a transform coding scheme which is of low complexity and can be used to extend current block-based DCT schemes to accomplish generic coding of segmented video over a wide range of bit-rates [4]. In using this transform, first an image is partitioned into adjacent blocks of $M \times M$ pixels. Each $M \times M$ block is then encoded using SA-DCT. The SA-DCT coefficients are quantized and run-length coded. The contour information is assumed to be encoded separately and transmitted to the receiver for decoding the image.

3.1 The SA-DCT Algorithm

The SA-DCT algorithm is based on representing $M \times M$ blocks in terms of predefined orthogonal sets of basis functions. The algorithm does not require any more computations than the ordinary DCT. Figure 2 illustrates the steps involved in coding an arbitrarily shaped image foreground segment contained within an 8×8 reference block. Figure 2(a) shows an image block segmented into a foreground (dark) and background (light). To perform vertical SA-DCT of the foreground, the vector length $N (0 < N < 9)$ of each column $j (0 < j < 9)$ of the foreground is calculated and the columns are shifted and aligned to the upper border of the 8×8 reference block (Figure 2(b)). Depending on the vector size N of each particular column of the segment, a DCT transform matrix $\underline{DCT_N}$, where

$$\underline{DCT_N}(p,k) = c_0 \cdot cos\Big[p\Big(k+\frac{1}{2}\Big)\cdot\frac{\pi}{N}\Big] \qquad p,k = 0,1,..,N-1. \tag{1}$$

containing a set of N DCT_N basis vectors is selected. Here $c_0 = \sqrt{\frac{1}{2}}$ if $p = 0$, and $c_0 = 1$ otherwise. p denotes the pth DCT basis vector. The N vertical DCT-coefficients $\underline{c_j}$ for each

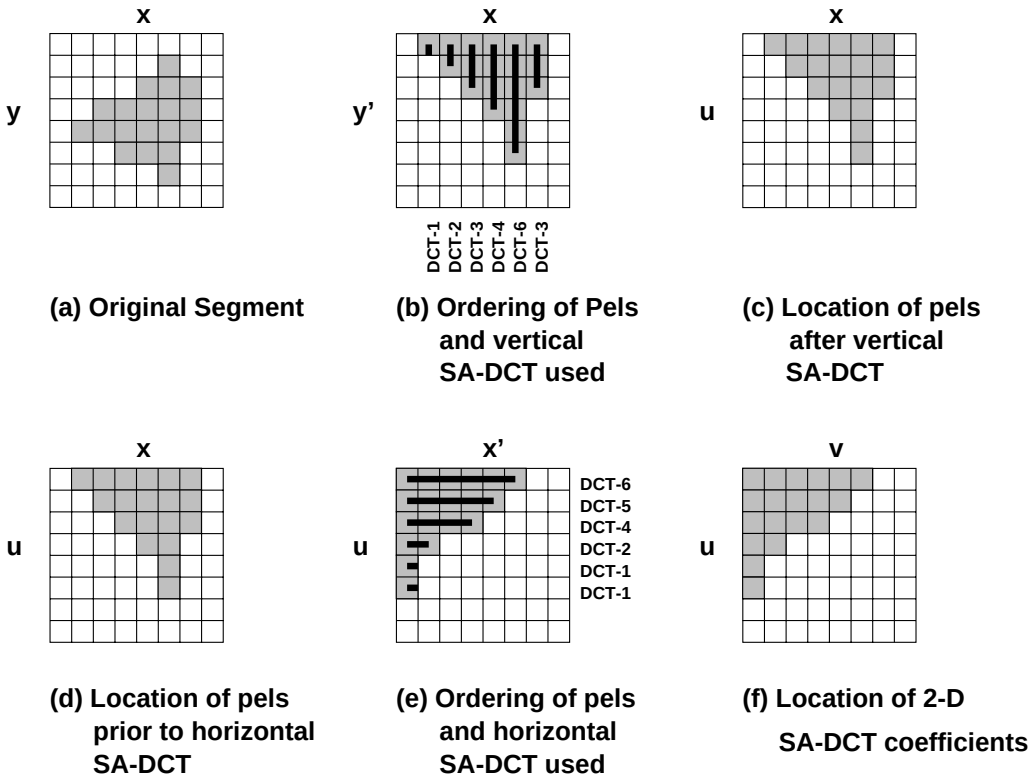

Figure 2: Successive steps to perform a forward SA-DCT on an arbitrarily shaped image foreground segment

segment column data $\underline{x_j}$ are calculated according to the formula

$$\underline{c_j} = \left(\frac{2}{N}\right) \cdot \underline{DCT_N} \cdot \underline{x_j} \tag{2}$$

In Figure 2(b) the 4th column belonging to the object is transformed using DCT_3 basis vectors. After SA-DCT in the vertical direction, the lowest DCT-coefficients (DC values $\bullet$) for the segment columns are found along the upper border of the 8×8 reference block (Figure 2(c)). To perform the horizontal DCT transformation (Figure 2(e)), the length of each row is calculated, and the rows are shifted to the left border of the 8×8 reference block, and a horizontal DCT adapted to the size of each row is calculated using equations (1) and (2). Note that horizontal SA-DCT is performed along vertical SA-DCT coefficients with the same index (i.e. all vertical DC coefficients ($\bullet$) are grouped together and are SA-DCT transformed in the horizontal dimension). Figure 2(f)

shows the final location of the resulting DCT coefficients within the 8×8 image block.

Performing SA-DCT results in the final number of DCT-coefficients that is identical to the number of pels contained in the image segment. Also, the coefficients are located in comparable positions as in a standard 8×8 block. The DC coefficient ($\bullet$) is located in the upper left border of the reference block, and depending on the actual shape of the segment, the remaining coefficients are concentrated around the DC coefficient. The contour of the segment is transmitted to the receiver the decoder using chain-difference coding [6][7]. The decoder performs the Inverse Shape-Adaptive DCT using the equation

$$\underline{x_j} = \underline{DCT_N}^T \cdot \underline{c_j} \tag{3}$$

in both horizontal and vertical segment direction. Here $\underline{c_j}$ denotes a horizontal or vertical coefficient vector of size N.

4 The MPEG-2 video bitstream: Syntax and Semantics

MPEG standards specify a syntax for the bitstream and the decoding process. An input video sequence is divided into units of **group of pictures (GOP's)** which can be decoded and reconstructed independently of other GOPs, if necessary (Figure 3). Each GOP has three kinds of pictures depending on how the picture is coded: Intra-coded (I) pictures, Predictive-coded (P) pictures and Bi-directionally predictive-coded (B) pictures[5]. An I-picture is coded using information only from itself and can be decoded independently of other pictures. A P-picture is a picture encoded using motion-compensated prediction from a past I-picture or P-picture. A B-picture is a picture which is coded using motion-compensated prediction from a past and/or future I or P-picture. The order of the coded pictures in the coded bitstream is the order in which the decoder processes them and not necessarily the order in which the pictures are displayed.

Each picture within the GOP is composed of one or more **slices**. A slice consists of a horizontal row of **macroblocks**. Slices occur in the bitstream in the order in which they are encountered, starting at the upper-left corner of the picture and proceeding by raster-scan order from left to

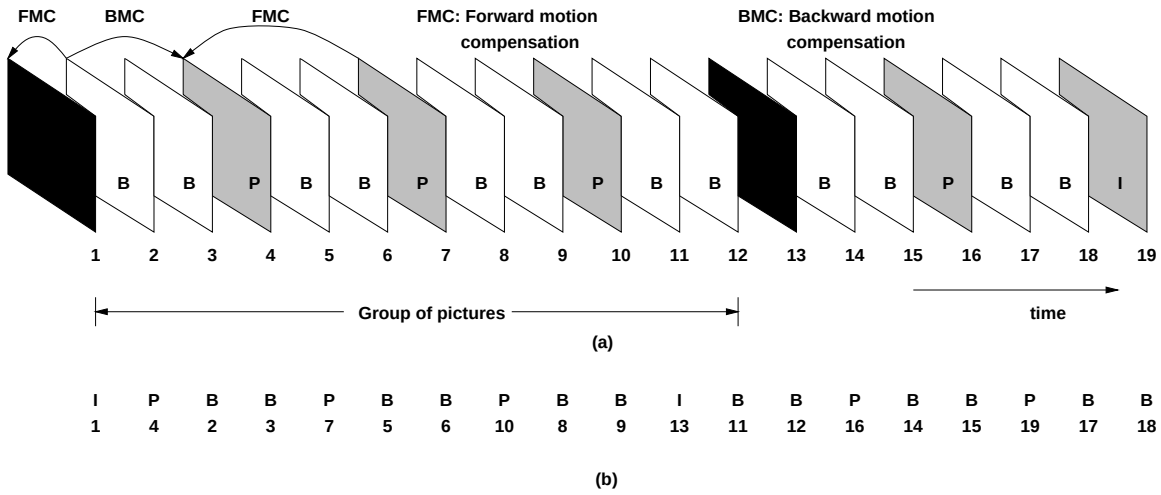

Figure 3: Example of a coded video sequence (a) display and (b) bitstream order

right and top to bottom (see Figure 4). A macroblock consists of a 16×16 block of luminance and associated 8×8 chrominance blocks. Each luminance **block** consists of 8×8 pixels. The 8×8 blocks are fundamental units of SA-DCT based transform coding.

5 Incorporating Content-Based Accessibility into the MPEG-2 framework

It was decided to use the current MPEG-2 syntax and modify its semantics involving the DCT to accomodate Shape-Adaptive DCT. Due to time constraints, we restricted ourselves to intra-frame coding of monochrome images and video sequences. To provide content-based addressability we introduced **object-based slices**. This means that we now have as many slices on a horizontal row of macroblocks as there are objects (Figure 5). The introduction of object-based slices allows us to decode only those slices which pertain to the object of interest. To incorporate this new concept into the MPEG-2 slice structure we used the **extra_information_slice** data element to specify upto 256 different objects. We also had to break away from the MPEG-2 specification to allow for skipped macroblocks in case of I-pictures and overlap of slices.

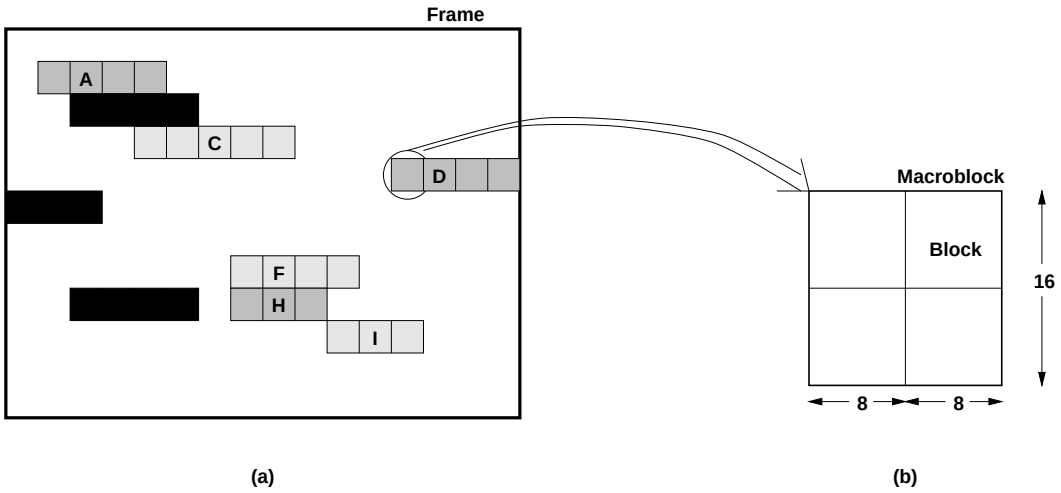

Figure 4: (a) General slice structure (b) Luminance macroblocks and blocks

6 Results

Still images such as *Lena* and *Barbara* were segmented by hand. The mask so obtained was used along with the input image to perform a Shape-Adaptive DCT of 8×8 image blocks. The SA-DCT coefficients were zig-zag scanned and run-length coded. MPEG-2 intra-vlc-tables were then used to Huffman encode the coefficients. At the decoder the reverse process was carried out in a manner that only objects of interest are decoded. The block diagram for the system is shown in Figure 6.

The images were encoded as explained above. A software simulation of the end-to-end system was built. Selected objects in the image were decoded, thus verifying access to objects in the image. In order to verify that we were not gaining this new functionality of accessing objects at the cost of compression efficiency, the bits-per-pixel (bpp) needed to encode *Lena* and *Barbara* at various quantization levels were computed. The results are tabulated in Table 1. Figure 7 shows *Lena* and *Barbara* coded using the scheme described at around 0.5 and 1.0 bpp.

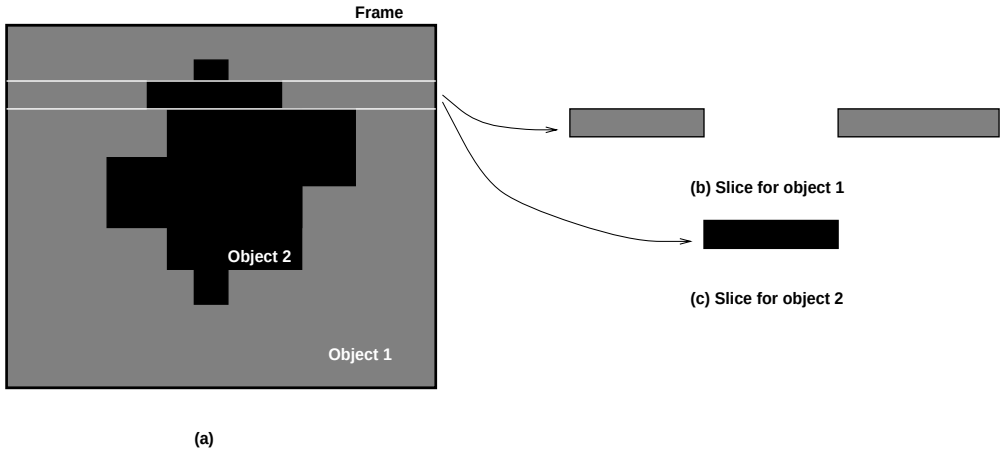

Figure 5: MPEG-4 slice structure

Table 1: Quantization v/s BPP

	Lena		*Barbara*	
Qscale	Bits per pixel	SNR	Bits per pixel	SNR
1	2.93	45.43dB	4.01	44.17dB
6	0.73	38.41dB	1.12	36.01dB
11	0.48	35.32dB	0.73	32.57dB
16	0.41	33.69dB	0.59	30.73dB
21	0.36	32.50dB	0.50	29.33dB
26	0.33	31.61dB	0.44	28.25dB
31	0.31	30.83dB	0.40	27.35dB

7 Conclusion

The suitability of SA-DCT to object-based encoding was established by designing an encoder and a decoder based on the MPEG-2 framework. It was verified that we would not be trading-off compression ratio to gain this additional functionality. Tests were also run to find out the overhead involved in encoding the object boundary using chain-difference coding. It was found that contour imformation contributed to an insignificant ($< 1\%$) part of the coded data. We now have a method for object based encoding that is efficient. Efforts are underway to develop methods for automatic segmentation of objects and extending SA-DCT based codec to inter-frame coding and full-color image sequences.

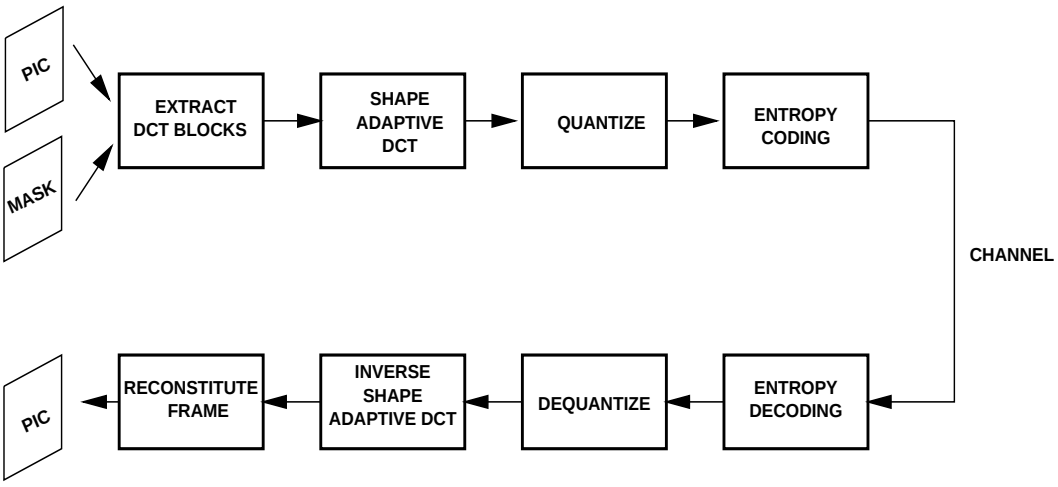

Figure 6: Block diagram of the system

A Appendix

The following are the changes that had to be made to the existing MPEG-2 code.

1. Allowing skipped macroblocks in INTRA frames.
2. Using SA-DCT routines instead of DCT.
3. Using the **extra_information_slice** data element to specify the object to be encoded/decoded.
4. Reconstituting the frame using slices belonging to different objects and allowing for overlap of slices.

References

[1] *Information Technology- Generic Coding of Moving Pictures and Associated Audio*, Recommendation ITU-T H.262, ISO/IEC 13818-2.

[2] *MPEG Moving Pictures Expert Group FAQ*, http://www.crs4.it/~luigi/MPEG/mpegfaq.html.

[3] *MPEG-4 Proposal Package Description (PPD) - Revision 2 (Lausanne Revision)*, ISO/IEC JTC1/SC29/WG11, MPEG 95/March 1995.

[4] Thomas Sikora and Bela Makai, Shape-Adaptive DCT for Generic Coding of Video, *IEEE Transactions on Circuits and Systems*, Vol.5, No.1, February 1995.

[5] Kiran Challapali, Video Compression for Digital Television Applications, *Philips Journal of Research*, August 1995.

[6] H. Freeman, On the Encoding of Arbitrary Geometric Configurations, *IRE Transactions on Electroncs and Computers*,pp. 260-268, Vol. EC-10, June 1961.

[7] H. Freeman, Computer Processing of Line-Drawing Images, *Computing Surveys*, pp. 57-97, Vol. 6, No.1, March 1974.

Figure 7(a): From top left corner clockwise: Original *Lena*, Decoded *Lena* Decoded foreground, Decoded background at 0.52bpp

Figure 7(b): From top left corner clockwise: Original *Lena*, Decoded *Lena* Decoded foreground, Decoded background at 1.1bpp

Figure 7(c): From top left corner clockwise: Original *Barbara*, Decoded *Barbara* Decoded foreground, Decoded background at 0.51bpp

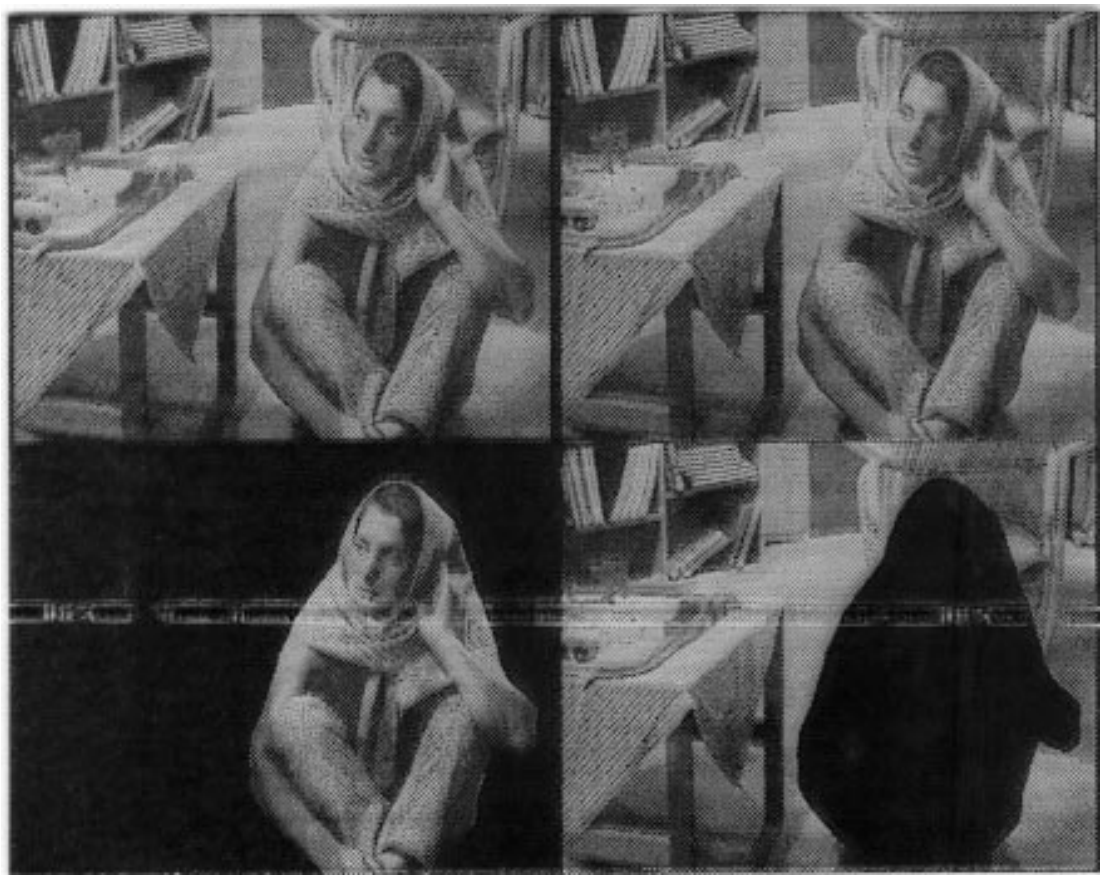

Figure 7(d): From top left corner clockwise: Original *Barbara*, Decoded *Barbara* Decoded foreground, Decoded background at 0.97bpp