Very Fast JPEG Compression Using Hierarchical Vector Quantization

Ricardo L. de Queiroz, *Senior Member, IEEE*, and Patrick Fleckenstein

***Abstract*—We derive a JPEG compliant image compressor, which is based on hierarchical vector quantization (HVQ). The goal is to reduce complexity while increasing compression speed. For each block, the DCT DC coefficient is encoded in the regular way, while the residual is mapped through HVQ to a precomputed bit stream corresponding to the compressed DCT AC coefficients. Approximation quality is generally good for high compression ratios. Color Fax is one possible application target for the proposed system.**

I. Introduction

BLACK and white facsimile has been widely used for many years, and more recently, color facsimile became a standard [1]. In color fax's baseline mode, the page is compressed using JPEG [2] in a CIELAB [1] color space. The protocols and transactions between transmitter and receiver assure that key transmission parameters such as resolution and color space are matched across both systems' capabilities. As the receiver capabilities are unknown until the negotiation is complete, image processing and compression for transmission have to be done on the fly. Thus, a system that would improve compression speed at lower complexity can largely reduce costs. Color fax systems [1] transmit full pages at moderate to high resolutions over phone lines. In order to keep the transmission time reasonable, images are often compressed at very high ratios. Furthermore, in color fax, images are likely to be printed on low cost printers. It is thus desirable to develop a system that

1) reduces implementation costs associated with JPEG
2) allows easy implementation in hardware;
3) allows fast software implementation;
4) approximates a JPEG coder well for high-compression ratios.

After establishing a motivation for a simple and fast substitute for a regular JPEG compressor, we highlight the point that the framework presented here is not limited to color fax systems and is applicable to any other JPEG compression system. In fact, it can be applied to other transform coders as well.

II. General Framework

The general idea is depicted in Fig. 1. The image is divided into blocks, as in JPEG [2]. The DCT DC coefficient (DCC), which is proportional to the average of the block entries, is computed. If the block entries are x_{ij} and their average is $\bar{x}$, the DCC is found as $x_{\rm dc} = (1/8)(\sum_{ij} x_{ij}) = 8\bar{x}$. The residual data is computed as $y_{ij} = x_{ij} - \bar{x}$ and its transform yields the desired DCT AC coefficients (ACC). In order to generate the compressed data, the ACC and DCC are quantized and encoded. In the case of the DCC, JPEG employs a differential coding approach, so that the quantized DCC is subtracted from its equivalent from a previous block. Thus, DCC's are not independent (see [2] for all details on JPEG). For this reason, we quantize and encode the DCC in the same way JPEG does.

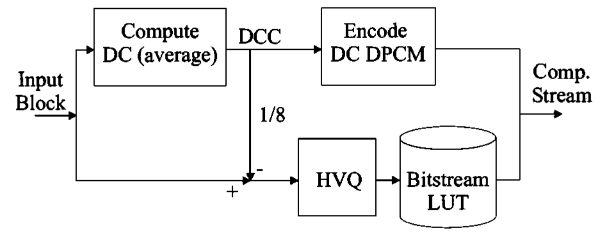

Fig. 1. General framework for HVQ-based JPEG. DCC is calculated by simple block averaging and encoded in the normal way. The residual is mapped through HVQ into a compressed bit-stream.

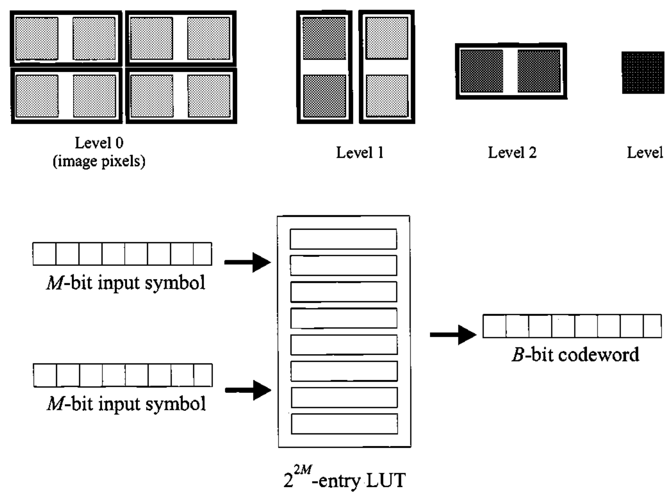

Fig. 2. HVQ look-up tables and a three-level encoding hierarchy.

The novel concept comes from the compression of the ACC. Regular JPEG requires transforming the data, quantization, zigzag scanning, run-length counting, Huffman coding, etc. We avoid all these steps by directly estimating the compressed bit stream. This is done by applying vector quantization (VQ) [3] to map the residual block to an index k, which is then mapped to a compressed bit stream.

Manuscript received June 4, 1999. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. K. K. Paliwal.

R. L. de Queiroz is with Xerox Corporation, Webster, NY 14580 USA (e-mail: queiroz@wrc.xerox.com).

P. Fleckenstein is with the Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: pat@csh.rit.edu).

Publisher Item Identifier S 1070-9908(00)04329-7.

1070–9908/00$10.00 © 2000 IEEE

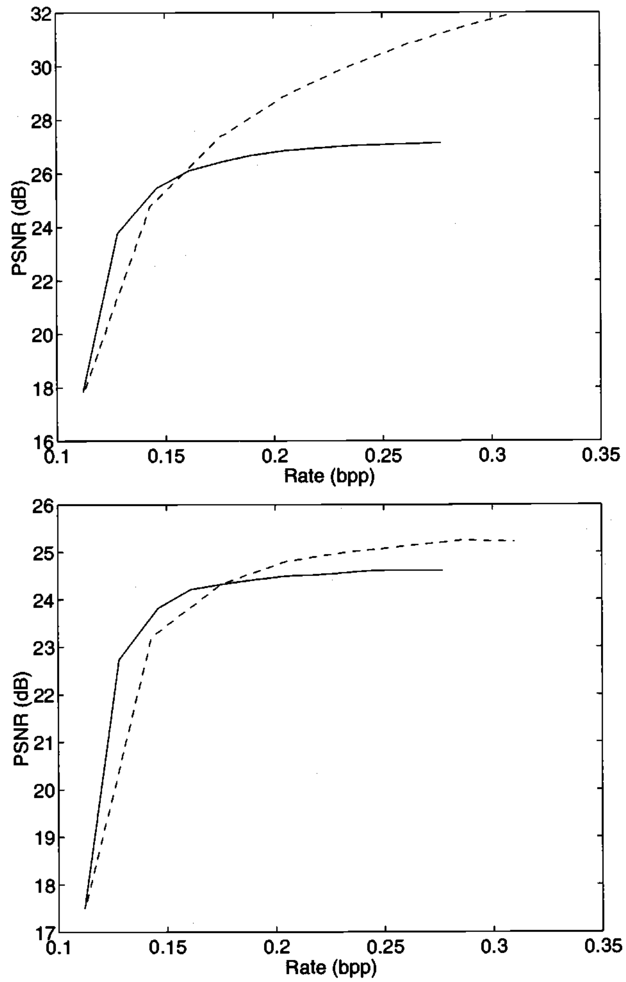

Fig. 3. Top: Peak SNR (PSNR) plots for different bit rates, comparing JPEG (dashed line) and HVQ-JPEG (solid line) for image Lena. Bottom: Same results after halftoning and inverse halftoning the decompressed images before computing the PSNR.

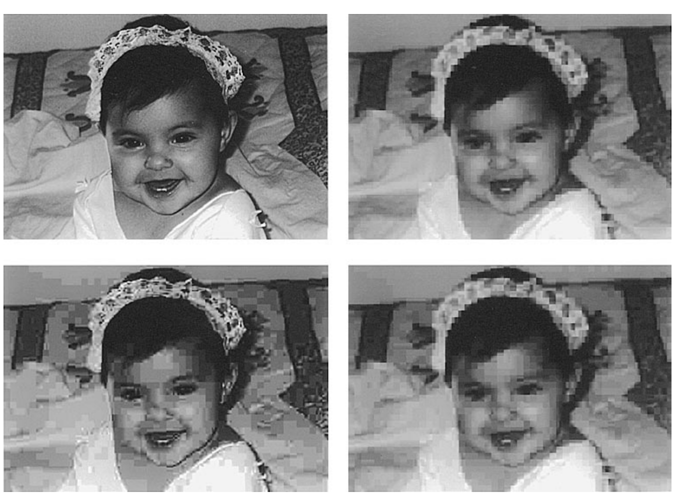

Fig. 4. (Top left) Original image of 400 $\times$ 280 pixels, (i.e., image obtained if there was no quantization in JPEG); (top right) image obtained if there was no scalar quantization in HVQ-JPEG (PSNR 23.6 dB); (bottom left) JPEG compressed image (PSNR 24.4 dB); (bottom right) HVQ-JPEG compressed image (PSNR 22.4 dB). Compression ratio is 36:1.

In a traditional VQ-based coding system, an input vector (image block) is matched to one of N available representative blocks, for which a codeword or index is generated. Codebooks and representative blocks are known to both transmitter and receiver. The selected codeword is transmitted to a receiver, which decodes the codeword and retrieves the associated block as an approximation of the input data block.

Here, as in regular VQ, we match the input block to one entry of a set of N representative blocks B_k. The representative blocks B_k, however, are pre-encoded using JPEG for a particular selection of quantizer and Huffman tables, generating N associated bit streams S_k of L_k bits each. As the residual block y_{ij} is actually mapped through VQ, only the part of the compressed bit stream corresponding to ACC is retained. The bit streams S_k are prestored and represent the ACC compressed data relative to a VQ approximation of the input block (i.e., the residual is mapped directly to S_k). Summarizing, the algorithm is

Offline

1) Select a training set of images.
2) Remove the mean of each block.
3) Design a set of representative blocks for the training data, for example using the LBG algorithm [3].
4) Compress each block in the set, discard the compressed DCC and store the part corresponding to the ACC in a bit-stream look-up table.

Online

1) Retrieve input image block.
2) Compute scaled average (DCC).
3) Quantize and encode DCC as in JPEG.
4) Transmit encoded DCC.
5) Compute residual (block entries less average).
6) Map residual to ACC bit-stream using VQ.
7) Retrieve and transmit the ACC bitstream.

III. HVQ Briefly

Although any VQ system could theorectically be employed, hierarchical VQ (HVQ) [4], [5] is the system of choice since it can be implemented solely with 63 look-ups over six tables for a block of 8 $\times$ 8 pixels. In HVQ, as illustrated in Fig. 2, two M-bit input symbols are mapped to one B-bit codeword. Next, two codewords are mapped to one codeword in a next level and the process is repeated until there is only one resulting codeword in the block. As pairs of M bit codewords are mapped to a B-bit codeword, a table of 2^{2M} entries of B bits each has to be generated for each level. The HVQ decoder is also based on look-up tables as well, but it is of no concern to us in this paper. In effect, HVQ is only used in this paper as a very fast mapping tool.

We refer to our approach as HVQ-JPEG.

IV. Performance Evaluation

For simulations, we designed codebooks for HVQ using 10-bit codewords for every stage. Hence, the ACC of the input block are mapped to at most 1024 different patterns. Furthermore, because HVQ design is greedy in nature [3] (so that we can have a hierarchy of 2-to-1 mappings), the representative block set is not as good as it could be using full VQ or by directly compressing the image using regular JPEG. Because of that, the quality of the compressed image is limited. At this stage, we are unable to design useful tables for six-level HVQ. Hence, we downsample the input block in Fig. 1 (using 2×2 pixel averaging) obtaining the downsampled block samples x'_{ij} whose average is the same of that of x_{ij}. The DCC is then computed as $x_{\mathrm{dc}} = (1/2)(\sum_{ij} x'_{ij}) = 8\bar{x}$ and the residual is $y_{ij} = x'_{ij} - \bar{x}$. In the design phase, the 1024 4×4 residual patterns are interpolated to reconstruct 8×8 blocks in order to find the library of ACC compressed bit streams S_k.

Downsampling and HVQ imprecision are limiting factors for the HVQ-JPEG performance. However, its implementation complexity is unrivaled. For processing each block, the complete system demands 63 table look-ups (HVQ), 63 additions and one shift (DCC), 64 subtractions (residual), plus the complexity to quantize and encode the DCC. However, the preferred method is the one using subsampling, wherein the implementation complexity for each image block is the complexity of JPEG encoding a DCC plus 15 table look-ups, 16 subtractions, 63 additions, and 17 shifts. In order to make a comparison, a regular JPEG coder would use hundreds of operations only to perform the DCT, plus hundreds of other operations to quantize all 64 coefficients to zigzag scan the quantized data and to encode the result. HVQ-JPEG is about an order of magnitude faster than regular JPEG in software implementation. The code for an HVQ-JPEG encoder is also very simple, making it suitable for software implementation in small systems. The tradeoff is the extra memory, which is needed to accomodate HVQ tables.

The plots shown in Fig. 3 illustrate the performance of the HVQ-JPEG scheme against regular JPEG for several bit rates. The gap in objective performance is significant but the errors can be largely masked for systems like color fax. In these, the images are likely to be scanned and printed at resolutions around 300 pixels/in so that blocks may be smaller than 1 mm. Also, the image is likely to be halftoned. In order to partially account for the imaging degradation, we repeated the objective experiments in Fig. 3, this time halftoning and inverse halftoning [6] the decompressed image before computing the objective distortion. In this case, the gap in performance is largely reduced, as can be seen in Fig. 3. In this example, we used the popular image Lena, while the codebook was designed using seven digitized photos. Examples are shown in Fig. 4 for a 400×280 pixels image that should be printed at a height slightly under 1 in at 300 dpi. Compression ratio is a modest 36:1, which is conservative for Color Fax applications, since it yields compressed page sizes in excess of 200 KB. Despite the 2 dB difference in PSNR, the image produced by HVQ-JPEG in Fig. 4 is competitive with its JPEG counterpart (the edges are softer but there are less ringing and blocking artifacts).

V. Concluding Remarks

The proposed system, HVQ-JPEG, is extremely simple to implement and does not require most of the standard JPEG steps (e.g. DCT, quantization, Huffman coding etc.). The only complex operation is the codebook design, which is performed offline. It is only suitable for high compression because of HVQ's inherent limitations. For large compression ratios, artifacts caused by HVQ mapping start to be masked by the artifacts caused by the scalar quantization in JPEG. As a bonus, since HVQ-JPEG tables are designed by training, one can fine tune the HVQ-JPEG performance to particular image characteristics (e.g., text) by including characteristic images in the training set.

This paper's objective was to expose the concept of HVQ mapping for indexing compressed bit streams. Several improvements can be made and were not included in this work, such as optimizing Huffman tables, which can be done offline at no implementation cost. We also would like to mention that HVQ-JPEG is an encoder, and its use is somewhat limited to transmission-only systems. This is so because JPEG chips are usually both encoders and decoders, and the requirement of having an onboard decoder may likely defeat the HVQ-JPEG advantage. HVQ-JPEG is easily implementable in either hardware or software. Color fax is only an application example, as HVQ-JPEG can be targeted to several other applications requiring cheap JPEG compressors.

Future research will be focused on codebook design and on the use of multiple codebooks with region adaptation.

References

[1] "Standardization of Group 3 Facsimile Apparatus for Document Transmission," ITU-T, Recommendation T.4, Annex E, July 1996. Optional Continuous Tone Color Mode.

[2] W. P. Pennebaker and J. L. Mitchell, *JPEG: Still Image Compression Standard*. New York, NY: Van Nostrand-Reinhold, 1993.

[3] A. Gersho and R. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.

[4] P. C. Chang, J. May, and R. Gray, "Hierarchical vector quantization with table look-up encoders," in *Proc. Int. Conf. Communications*, Chicago, IL, 1985, pp. 1452–1455.

[5] M. Vishwanath and P. Chou, "An efficient algorithm for hierarchical compression of video," in *Proc. Int. Conf. Image Processing*, Austin, TX, 1994, pp. 275–279.

[6] J. Luo, R. L. de Queiroz, and Z. Fan, "A robust technique for image descreening based on the wavelet transform," *IEEE Trans. Signal Processing*, vol. 46, pp. 1179–1185, Apr. 1998.