A Low-Complexity, Fixed-Rate Compression Scheme for Color Images and Documents

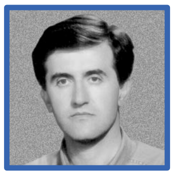
Nader Moayeri

Based on one-dimensional differential pulse code modulation, the coder is multiplication-free, codes each pixel as soon as it is available, and outputs a fixed number of bits for each pixel. Hence, there is no need for any buffering of the input image or coder output bitstream. The compression scheme is visually lossless and yields a modest compression ratio of 3 to 4. Because of its simplicity, it is useful when hardware is limited and coding delays cannot be tolerated.

There is often a need for some level of data compression in many imaging devices and products, such as scanners and facsimile machines. The reason can be the limited bandwidth of the data bus connecting the scanner to the host computer or the size of the memory available in the product. In addition, strict hardware limitations can preclude the use of compression schemes that are computationally intensive.

A desirable feature of a compression scheme is that it be a fixed-rate coder. Specifically, it is desirable for the compression scheme to produce the same number of bits in compressing an 8.5-by-11-inch sheet of paper *regardless of its content*. This guarantees that the memory of the device will never overflow and that the data is always transmitted to the host in a fixed time. Many popular compression schemes, such as the lossy JPEG and its newly adopted lossless JPEG-LS algorithms, are variable-rate coders. They have the advantage that they yield higher compression ratios than fixed-rate coders of comparable complexity. However, their variable-rate nature implies that in compressing a sheet of paper, there can be a large variation in the number of bits they generate, depending on the content of the sheet.

In this paper we present a very simple compression scheme that adequately addresses all the issues raised above. It compresses a given sheet of paper



Nader Moayeri
From 1994 to 1997, Nader Moayeri was a member of the technical staff of the Imaging Technology Department of HP Laboratories. Before joining HP, he was an assistant professor of electrical and computer engineering at Rutgers University, and he is currently manager of the Wireless Communications Technologies Group of the U.S. National Institute of Standards and Technology. A senior member of the IEEE, he received his PhD degree in electrical engineering from the University of Michigan at Ann Arbor in 1986. He is married, has a son, and enjoys volleyball, soccer, and opera.

(which may contain images, text, graphical art, etc.) pixel by pixel without any need for buffering the image data as it is being scanned. It produces a fixed number of bits for every pair of pixels, which can be stored or sent to the host over the data bus. An equally simple algorithm decompresses the received bits to produce a visually lossless rendition of the image data. The coder is lossy, but the loss level is so small that it is perceptually negligible. The coder is based on one-dimensional differential pulse code modulation (1D DPCM) using fully integer operation predictors and quantizers.

Proposed Compression Scheme

In this section we describe the various components and aspects of the compression scheme.

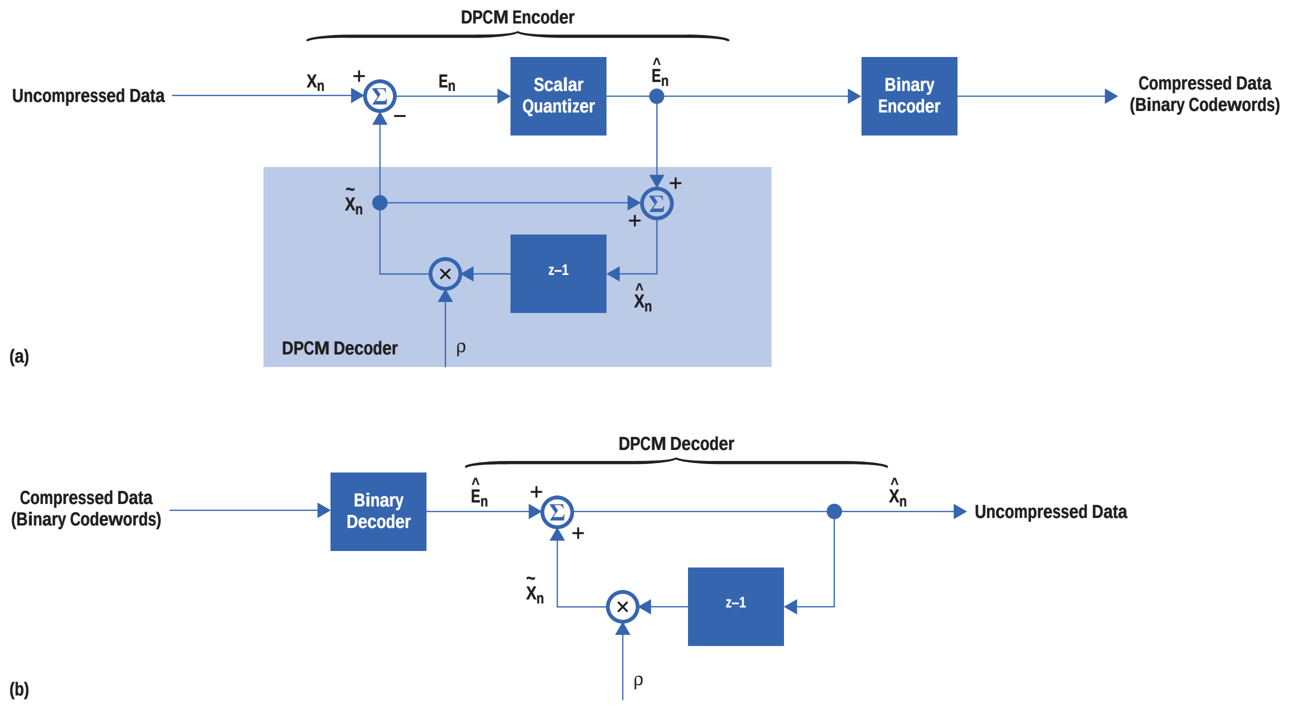
Differential pulse code modulation (DPCM) is a classical lossy data compression technique originally developed at Bell Laboratories for compressing the television signal.[1] It is the next level of sophistication in the compression technique hierarchy after pulse code modulation (PCM), which is the same as nonuniform scalar quantization.

The block diagram for a DPCM coder with a first-order predictor is shown in **Figure 1**. The task of the binary encoder is to map the output $\hat{E}_n$ of the scalar quantizer to a binary codeword in a one-to-one fashion. For example, if the quantizer has 16 output levels, then the binary encoder will assign a four-bit distinct codeword to each possible quantizer output level. The binary decoder is simply the inverse of the binary encoder and maps a given binary codeword back to its respective quantizer output level. In other words, the input to the DPCM decoder is simply $\hat{E}_n$. Also note that there is a copy of the DPCM decoder at the DPCM encoder as shown by the shaded box in **Figure 1**.

The main idea of DPCM is to decorrelate the source data before coding it. Correlations exist in many data sources, such as images. DPCM removes these correlations by making a prediction $\tilde{X}_n$ of the next source sample X_n on the basis of the past reconstructions and then coding the prediction error E_n instead of X_n itself. (A PCM system codes X_n directly.) In the system of **Figure 1**, the predictor uses the reconstruction for the most recent source

Figure 1

The block diagram of a 1D DPCM coder with a first-order predictor (a) Encoder. (b) Decoder.



sample to predict the next one. That is, $\tilde{X}_n = \rho\hat{X}_{n-1}$, where ρ is a constant coefficient. A key property of a DPCM coder, even those employing higher-order predictors, is that:

$$X_n - \hat{X}_n = E_n - \hat{E}_n. \tag{1}$$

Hence, the mean squared error (MSE) distortion of the system is given by:

$$D = E[(X_n - \hat{X}_n)^2] = E[(E_n - \hat{E}_n)^2]. \tag{2}$$

If the predictor does a good job of predicting the source, then E_n will have a small variance (or effective dynamic range) compared to X_n. Consequently, an N-level optimal scalar quantizer for E_n will yield a smaller average distortion than one for X_n.

The predictor coefficient ρ depends on the samplewise correlation in the source; a larger value of ρ is used when the source is highly correlated. From a theoretical point of view, ρ should always be smaller than one so that the decoder filter remains stable. From a practical point of view, however, it is advantageous to use $\rho = 1$. We have opted for this choice in this work in the interest of making the coder as simple as possible. We have not experienced any instability problems in our experiments with the coder presented here.

We also note that two-dimensional DPCM, in which each pixel is predicted based on reconstructions for the previous pixel on the same image row and a couple of pixels on the previous row just above the present pixel, is a better choice for image compression. The predictor in 2D DPCM performs better than the one in 1D DPCM, making it possible to achieve better coded image quality at the same compression ratio. Using a 2D predictor requires buffering the reconstruction for the previous row of the image and the part of the present row that has already been encoded. In the interest of keeping the memory requirements of our coder at a minimum, we use the 1D predictor.

Quantization

In this work we assume that each output level of the scalar quantizer is assigned a binary codeword of length $\log_2 N$ by the binary encoder. (It is also possible to entropy-code the quantizer output. Since that would lead to a variable-rate coder, we are not interested in entropy-coded DPCM in this work.) If the probability distribution of the prediction error E_n is known, then an optimal N-level nonuniform scalar quantizer for E_n can be designed. Such quantizers are called Lloyd-Max quantizers and there are algorithms for their design based on a training sequence of samples from the source.[2,3] We used the Lloyd algorithm to design the quantizers needed in this work.

The problem of quantizer design in a DPCM system is complicated because the probability distribution of E_n depends on the quantizer and vice versa. This problem has been addressed by Arnstein,[4] who describes an iterative design algorithm for the case of a first-order Gauss-Markov source. Arnstein's work, however, cannot be readily extended to the case of a DPCM system for images. Therefore, we use an *open-loop* approach to quantizer design. Since we are interested in a visually lossless compression scheme, it is reasonable to assume that $\hat{X}_n \approx X_n$. This approximation can be used to generate a training sequence of samples of the prediction error. Using a set of training images, we form the training sequence for E_n by simply subtracting each image pixel from the next one. Note that:

$$\begin{aligned} E_n &= X_n - \tilde{X}_n = X_n - \rho\hat{X}_{n-1} = X_n - \hat{X}_{n-1} \\ &\approx X_n - X_{n-1}. \end{aligned} \tag{3}$$

Finally, to make the coder as simple as possible, we force the quantizer output levels to be integers by properly modifying the centroid condition in the Lloyd algorithm. Specifically, we replace each centroid by the integer closest to it. It is easy to show that this is indeed the optimal way of designing integer valued scalar quantizers. The problem that has to be solved is the following:

Problem: Given a random variable X, find an integer C that minimizes $E[(X-C)^2]$.

Solution: For any integer C we can write:

$$\begin{aligned} E[(X - C)^2] &= E[(\{X - E[X]\} + \{E[X] - C\})^2] \\ &= E[(X - E[X])^2] + (E[X] - C)^2 \\ &\quad + 2(E[X] - C)E[X - E[X]] \\ &= \sigma_X^2 + (E[X] - C)^2, \end{aligned} \tag{4}$$

where σ_X^2 is the variance of X. The righthand side is minimized by choosing C to be the integer closest to E[X]. In addition, since the input to each quantizer is integer valued, we implement each quantizer with a small lookup table.

DPCM for Image Compression

It is straightforward to design a compression system for monochrome images based on the simple ideas presented above. Usually the system specifications (bus bandwidth and memory size) dictate a maximum rate r in bits per pixel (bpp) for the images that are to be compressed by this system. We assume that the rate is integer valued. (Otherwise, several quantizer outputs should be grouped together before being assigned a binary codeword. This would imply a need for some buffering.) The number of quantizer output levels is then given by $N = 2^r$, and the quantizer is designed according to the procedure outlined earlier.

In the case of color images the situation is more complicated. Of course, it is always possible to compress the image data in the RGB color space. However, it is advantageous to first transform the data into some other color space and then code it. Such transformations result in energy compaction in the data, which makes it possible to achieve better compression performance. For our work we picked the YUV color space. (The image data in some HP scanner pipelines is already in the YUV format.) It is well known that most of the image information is in the Y signal. Hence, in many color image compression algorithms, such as JPEG, the U and V signals are subsampled by a factor of 2 in both the horizontal and vertical directions before being coded. This subsampling compresses the U and V data by a factor of 4 even before any coding is done. In this work we have investigated both alternatives—subsampling the U and V signals and not doing any subsampling. In the latter case we have to use quantizers of lower rate for the U and V signals so that the overall coding rate (or compression ratio) is the same as in the subsampling case.

Yet another crucial factor in color image compression is the allocation of the available bit rate to the three color planes. There are optimal bit allocation algorithms that maximize the peak signal-to-noise ratio (SNR) and are commonly used in monochrome image coding.[5,6] In the case of color images there are three color planes to worry about, and the peak SNR is even less relevant than in the monochrome case. In this work we have taken an experimental approach for finding the best rate allocation. Our criterion is to get the best possible subjective image quality for a given overall rate. We are interested in rate values that yield visually lossless coded image quality. Since we have restricted quantizer rates to be integers, there are only a small number of bit allocations possible for any given overall rate. We found the best bit allocation experimentally and by trail and error.

Experimental Results

We designed a compression system for color images based on the ideas presented in the previous section and tested it on a test image we obtained from the HP Greeley Hardcopy Division. This is a compound image having photographic content, text, line art, and a rainbow ramp. In coding the U and V signals we investigated three alternatives:

(i) Coding these signals at full resolution

(ii) Subsampling them in the horizontal direction by simply discarding every other pixel and using pixel replication after the coding process

(iii) Replacing each pair of successive samples by their average and then using a 3-tap area-based interpolation filter[7] after coding.

Since the choice of the subsampling strategy or lack thereof affects the samplewise correlation of the U and V signals, hence influencing the probability distribution of the prediction error, we had to design two quantizers for each of the above alternatives. Since the Y signal is not subsampled, one quantizer is sufficient for that signal. Therefore, we designed a total of seven sets of quantizers, each set containing quantizers with 2, 4, 8, 16, 32, and 64 output levels. We used a training sequence consisting of the above test image and four other large images to design the needed quantizers. We transformed these images into the YUV color space, formed three *mother* training sequences for the Y, U, and V signals by taking differences of successive samples as described under "Quantization" above, and formed two more sequences from each of the mother sequences for U and V signals by properly subsampling them according to the alternatives (ii) and (iii) above. Using these seven training sequences we designed the needed seven sets of quantizers. Then we experimented with different systems, with or without U and V subsampling and with various choices of quantizers, to code the test image. **Table I** shows the compression performance obtained with various choices for compressing the 24-bit test image at 12, 11, 10, 9, and 8 bits per pixel (bpp). The first two lines in the table correspond to transmitting certain numbers with infinite precision to the receiving end. This is not practical, of course, but it sets an upper bound on the best performance that can be expected from strategies (ii) and (iii).

For all the coders in the table designated with an asterisk (*), the quality of the coded image is visually lossless. (Of course, the image quality is somewhat better at 12 bpp than at 8 bpp when the image is scaled.)

In reading the bit rates in **Table I**, note that with strategies (ii) and (iii) there are half as many U and V samples to be coded as there are Y samples. Also note that the test image is a particularly difficult image to work with. With other photographic images, such as the other four images in the training set, we were easily able to get a compression ratio of 4 to 6 and yet maintain visually lossless image quality.

Table I
Bit Rate and Peak SNR (Signal-to-Noise Ratio) of the Proposed Coder with Various Test Image Parameters

Overall Coder Rate (bpp)	Subsampling Strategy for U and V	Subcoder Rate (bpp)			Peak SNR (dB) YUV Color Space				Peak SNR (dB) RGB Color Space				
		Y	U	V	Y	U	V	Avg.	R	G	B	Avg.	
∞	(ii)	∞	∞	∞	∞	33.73	33.77	∞	31.21	35.28	29.22	31.90	
∞	(iii)	∞	∞	∞	∞	37.91	37.72	∞	35.07	38.75	33.39	35.73	
12	(i)	6	3	3	45.26	39.54	40.06	41.62	36.58	40.56	34.53	37.22	*
12	(i)	4	4	4	38.27	44.84	45.14	42.75	36.81	37.85	36.08	36.91	
12	(ii)	6	6	6	45.26	33.60	33.65	37.51	30.95	34.76	29.01	31.57	
12	(iii)	6	6	6	45.26	37.42	37.39	40.02	34.41	37.61	32.65	34.89	
11	(i)	5	3	3	42.48	39.54	40.06	40.69	36.07	39.46	34.21	36.58	*
11	(ii)	6	5	5	45.26	33.50	33.43	37.40	30.71	34.61	28.91	31.41	
11	(ii)	5	6	6	42.48	33.60	33.65	36.58	30.82	34.42	28.92	31.39	
11	(iii)	6	5	5	45.26	36.94	37.07	39.76	34.08	37.36	32.20	34.54	
11	(iii)	5	6	6	42.48	37.42	37.39	39.10	34.10	37.03	32.43	34.52	
10	(i)	4	3	3	38.27	39.54	40.06	39.29	34.72	36.97	33.24	34.98	
10	(ii)	6	4	4	45.26	33.24	33.41	37.30	30.73	34.57	28.67	31.32	
10	(ii)	5	5	5	42.48	33.50	33.43	36.47	30.58	34.29	28.82	31.23	
10	(ii)	4	6	6	38.27	33.60	33.65	35.18	30.37	33.44	28.63	30.82	
10	(iii)	6	4	4	45.26	37.02	36.84	39.71	33.93	37.29	32.33	34.52	*
10	(iii)	5	5	5	42.48	36.94	37.07	38.83	33.79	36.82	31.99	34.20	
10	(iii)	4	6	6	38.27	37.42	37.39	37.30	33.21	35.41	31.79	33.47	
9	(ii)	6	3	3	45.26	32.37	32.35	36.66	29.80	33.88	27.88	30.52	
9	(ii)	5	4	4	42.48	33.24	33.41	36.37	30.61	34.25	28.59	31.15	
9	(ii)	4	5	5	38.27	33.50	33.43	35.07	30.15	33.35	28.54	30.68	
9	(iii)	6	3	3	45.26	35.16	34.92	38.45	32.11	36.07	30.64	32.94	
9	(iii)	5	4	4	42.48	37.02	36.84	38.78	33.65	36.74	32.14	34.18	*
9	(iii)	4	5	5	38.27	36.94	37.07	37.43	32.93	35.29	31.40	33.21	
8	(ii)	5	3	3	42.48	32.37	32.35	35.73	29.70	33.61	27.81	30.37	
8	(ii)	4	4	4	38.27	33.24	33.41	34.97	30.19	33.31	28.31	30.60	
8	(iii)	5	3	3	42.48	35.16	34.92	37.52	31.91	35.66	30.50	32.69	
8	(iii)	4	4	4	38.27	37.02	36.84	37.38	32.83	35.21	31.56	33.20	*

The asterisks in the rightmost column designate the coder that gave the best image quality for each rate.

Table II

Number of Arithmetic Operations per Image Pixel Needed by the Encoder and Decoder With Each of the Three U and V Subsampling Strategies

	Encoder			Decoder		
Strategy	Adds, Subtracts	Table Lookups	Shifts	Adds, Subtracts	Table Lookups	Shifts
(i)	6	3	0	3	3	0
(ii)	4	2	0	2	2	0
(iii)	5	2	1	5	2	3

It can be seen from the table that the naive subsampling strategy (ii) of the U and V signals is always inferior to the more careful strategy (iii). This is also true in terms of coded image quality. However, implementing strategy (iii) adds a little bit to the computational requirements of the coder. We also note that strategy (iii) is in some cases inferior to strategy (i), namely not subsampling at all. In fact, we made the observation that at high bit rates, even if no quantization at all is done after subsampling with either strategy (ii) or (iii), the quality of the reconstructed image can be inferior to an image obtained with coding and strategy (i). This can be seen by comparing the first two lines of **Table I** with the third line, for example. The reason is the inherent loss of the subsampling and reconstruction operations. Finally, strategy (iii) seems to be better than strategy (i) at bit rates 10, 9, and 8 bpp. This might be because in our experiments we restricted our attention to cases where the U and V signals get equal shares of the available bit rate and the quantizer rates are all integers.

Finally, we like to stress that all three coding strategies proposed in this paper are very simple. **Table II** shows the exact number of encoding and decoding operations required by each strategy. All three methods have negligible computational complexity and hence would have a very simple implementation and would run very fast. The amount of memory needed at the encoder is simply that for the three lookup tables implementing the three quantizers. Each method would need three tables of 511 bytes each with a conservative allocation of one byte per table entry. Each decoding method would also need three lookup tables to map the quantizer indices to the quantizer output levels. These tables would be on the host and not on the scanner. They are even smaller than those needed at the encoder, because each would have as many entries as the number of quantizer output levels.

Conclusion

In this paper we have presented a compression scheme for color images based on one-dimensional differential pulse code modulation (DPCM). The image data is assumed to be in YUV or YCrCb color space, and the chrominance information may or may not have been subsampled. There is one DPCM coder for each of the three color components, and an optimal nonuniform scalar quantizer (a Lloyd-Max quantizer) for each coder. The three quantizers have been designed in such a way that all their output levels are integer valued. Therefore, the coder uses integer operations exclusively. The quantizers are implemented by lookup tables and the predictors are simple. Consequently, the coder is multiplication-free. The coder operates in real time in the sense that each pixel is coded as soon as it is available (this would be useful in a scanner). In addition, the coder outputs a fixed number of bits for each pixel. Hence, there is no need for any buffering of the coder output bitstream before transmission. All these features make the coder very simple and easily implementable with minimal hardware. Even the memory requirement of the coder is negligible because the lookup tables are small and there is no need to buffer the image pixels.

This compression scheme has been tested on several images of different types and contents. The coded image quality is very good and visually lossless in all cases. The coder yields a modest compression ratio of 3 to 4. The coder’s compression performance is not as high as the industry-standard JPEG algorithm, but then JPEG is a variable-rate coder that is orders of magnitude more complex. Because of its extreme simplicity, the coder is suitable for applications in which there is limited hardware capability and coding delays cannot be tolerated. It may be the simplest coder that achieves modest compression without compromising the image quality.

References

1. C.C. Cutler, *Differential quantization for television signals*, U.S. Patent 2605361, July 1952.

2. S.P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, Vol. IT-28, Part I, March 1982, pp. 129-137.

3. J. Max, "Quantizing for minimum distortion," *IEEE Transactions on Information Theory*, Vol. IT-6, March 1960, pp. 7-12.

4. D.S. Arnstein, "Quantization error in predictive coders," *IEEE Transactions on Communications*, Vol. COM-23, April 1975, pp. 423-429.

5. Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-36, September 1988, pp. 1445-1453.

6. E.A. Riskin, "Optimal bit allocation via the generalized BFOS algorithm," *IEEE Transactions on Information Theory*, Vol. IT-37, March 1991, pp. 400-402.

7. P.W. Wong and C. Herley, "Area-Based Interpolation for Scaling of Images from a CCD," *Proceedings of the IEEE International Conference on Image Processing*, Santa Barbara, California, November 1997, pp. I905-908.