An Edge Preserving Differential Image Coding Scheme*

Martin C. Rost
Sandia National Laboratories
Albuquerque, NM

Khalid Sayood
Department of Electrical Engineering and
Center for Communication and Information Science
University of Nebraska
Lincoln, NE 68588-0511

Abstract

Differential encoding techniques are fast and easy to implement. However, a major problem with the use of differential encoding for images is the rapid edge degradation encountered when using such systems. This makes differential encoding techniques of limited utility especially when coding medical or scientific images, where edge preservation is of utmost importance. We present a simple, easy to implement differential image coding system with excellent edge preservation properties. The coding system can be used over variable rate channels which makes it especially attractive for use in the packet network environment.

Introduction

The transmission and storage of digital images requires an enormous expenditure of resources, necessitating the use of compression techniques. These techniques include relatively low complexity predictive techniques such as Adaptive Differential Pulse Code Modulation (ADPCM) and its variations, as well as relatively higher complexity techniques such as transform coding and vector quantization [1,2]. Most compression schemes were originally developed for speech and their application to images is at times problematic. This is especially true of the low complexity predictive techniques. A good example of this is the highly popular ADPCM scheme. Originally designed for speech [3], it has been used with other sources with varying degrees of success. A major problem with its use in image coding is the rapid degradation in quality whenever an edge is encountered. Edges are perceptually very important and occur quite often in most images. Therefore, the degradation of edges can be perceptually very annoying. If the images under consideration contain medical or scientific data, the problem becomes even more important, as edges provide position information which may be crucial to the viewer. This poor edge reconstruction quality has been a major factor in preventing ADPCM from becoming as popular for image coding as it is for speech coding.

While good edge reconstruction capability is an important requirement for image coding schemes, another requirement that is gaining in importance with the proliferation of packet switched networks, is the ability to encode the image at different rates. In a packet switched network, the available channel capacity is not a fixed quantity, but rather fluctuates as a function of the load on the network. The compression scheme must therefore be capable of operating at different rates as the available capacity changes. This means that it should be able to take advantage of increased capacity when it becomes available while providing graceful degradation when the rate decreases to match decreased available capacity.

In this paper we describe a DPCM based coding scheme which has the desired properties listed above. It is a low complexity scheme with excellent edge preservation in the reconstructed image. It takes full advantage of the available channel capacity providing lossless compression when sufficient capacity is available, and very graceful degradation when a reduction in rate is required.

Notation and Problem Formulation

The DPCM system consists of two main blocks, the quantizer and the predictor (see Fig. 1). The predictor uses the correlation between samples of the waveform to predict the next sample value. This predicted value is removed from the waveform at the transmitter and reintroduced at the receiver. The prediction error is quantized to one of a finite number of values which is coded and transmitted to the receiver. The difference between the prediction error and the quantized prediction error is called the quantization error or the quantization noise. If the channel is error free, the reconstruction error at the receiver is simply the quantization error. To see this, note (Fig. 1) that the prediction error $e(k)$ is given by

$$e(k) = s(k) - p(k) \tag{1}$$

where the predicted value is given by

$$p(k) = \sum a_j \hat{s}(k-j) \tag{2}$$

and

*This work was supported by the NASA Goddard Space Flight Center under Grant NAG 5-916.

$$\tilde{s}(k) = e_q(k) + p(k). \quad (3)$$

Assuming an additive noise model, the quantized prediction error $e_q(k)$ can be represented as

$$e_q(k) = e(k) + n_q(k) \quad (4)$$

where $n_q(k)$ denotes the quantization noise. The quantized prediction error is coded and transmitted to the receiver. If the channel is noisy this is received as $\tilde{e}_q(k)$ which is given by

$$\tilde{e}_q(k) = e_q(k) + n_c(k) \quad (5)$$

where $n_c(k)$ represents the channel noise. The output of the receiver $\tilde{s}(k)$ is thus given by

$$\tilde{s}(k) = \tilde{p}(k) + \tilde{e}_q(k) \quad (6)$$

where

$$\tilde{p}(k) = p(k) + n_p(k) \quad (7)$$

the additional term $n_p(k)$ being the result of the introduction of channel noise into the prediction process. Using (1), (4), (5), and (7) in (6) we obtain

$$\tilde{s}(k) = s(k) + n_q(k) + n_c(k) + n_p(k). \quad (8)$$

If the channel is error free, the last two terms in (8) drop out and the difference between the original and reconstructed signal is simply the quantization error.

When the prediction error is small, it falls into one of the inner levels of the quantizer, and the quantization noise is of a type referred to as granular noise. If the prediction error falls in one of the outer levels of the quantizer, the incurred quantization error is called overload noise. Because of the way the granular noise is generated it is generally smaller in magnitude than the overload noise and is bounded by the size of the quantization interval. The overload noise on the other hand is essentially unbounded and can become very large depending on the size of the prediction error. As edge pixels are rather difficult to predict, the corresponding prediction error is generally large, and this leads to large overload noise values. Furthermore, because this error effects not only the reconstruction of the current pixel, but also future predictions, the prediction errors corresponding to the next few pixels also tend to be large, leading to an edge "smearing" effect.

Reduction of the edge degradation can therefore be obtained by reducing or eliminating the slope overload noise. Reduction of the slope overload noise can be obtained by improving the prediction process. Gibson [4] analyzed ADPCM systems with backward adaptive prediction, and showed that the tracking ability of the adaptive predictor can be improved by the addition of zeros in the predictor. Motivated by these results, Sayood and Schekall [5] designed ADPCM systems for image coding with ARMA predictors. Their results show that some reduction in the edge degradation is possible with the use of adaptive zeros in the predictor. While the use of these predictors improves the edge reconstruction there is still significant degradation in the edges. One technique to further improve the edge performance was developed by Schekall and Sayood [6], which uses the Jayant quantizer as an edge detector. The overload noise is then reduced by sending a quantized representation of the noise through a side channel. The advantage of this approach is that it can be added to existing ADPCM systems. The disadvantage is that the use of a side channel introduces synchronization problems. In this paper we propose a different approach for edge preservation which does not require a side channel. This approach is described in the following section.

Proposed Approach

The approach taken in this paper is a variation on the standard rate-distortion tradeoff. The basic idea is that the slope overload noise can be reduced by increasing the rate. However rather than increasing the rate for encoding each and every pixel, there is only an instantaneous rate increase whenever slope overload is encountered. The way this is implemented is outlined in the block diagram of Figure 2. A DPCM system is followed by a lossless encoder at the transmitter. At the receiver the inverse operations are performed. The DPCM system differs from standard DPCM systems in that the quantizer being used has an unlimited number of levels. In practice what this means is that if the input has 256 levels, which is standard for monochrome images, then the DPCM quantizer will have 512 levels. This effectively eliminates the overload noise making the distortion a function of the quantizer stepsize Δ. Of course by itself it also eliminates any compression that may have been desired, in fact it requires an increase of one bit in the rate. The compression is obtained by use of the lossless encoder. The lossless encoder output alphabet consists of N codewords. These codewords correspond to N consecutive levels in the quantizer. Let the smallest level be labeled x_L and the largest level be labeled x_H. If the quantizer output $e_q(k)$ is a level between x_L and x_H, then the lossless encoder puts out the corresponding channel symbol. If, however, $e_q(k)$ is greater than x_H the encoder puts out the symbol corresponding to x_H. A new value $e_{q1}(k)$ is then obtained by subtracting x_H from $e_q(k)$. If this value is less than x_H then it is encoded using the corresponding codeword in the lossless encoder output alphabet. Otherwise, x_H is again subtracted from $e_{q1}(k)$ to generate $e_{q2}(k)$. This process is continued till some $e_{qn}(k)$ where

$$e_{qn}(k) = e_q(k) - nx_H$$

and $e_{qn}(k)$ is less than x_H. A similar strategy is followed when $e_q(k) \leq x_L$. Thus the instantaneous rate is increased by a function of n whenever the prediction error falls outside the closed interval $[x_L, x_H]$.

Example : Consider a DPCM system with a stepsize Δ of 2 where the input output relationship is given by

$$Q[x] = 2k \quad \text{if} \quad 2k-1 \leq x < 2k+1; \quad k = 0, \pm 1, \pm 2, \ldots$$

Let the lossless encoder output alphabet be of size eight with $x_L = -4$, and $x_H = 10$. If the input $e(k)$ is 7 the quantiger output $e_q(k)$ is 8, which is in the lossless encoder output alphabet and therefore this value is encoded as a single codeword. If $e(k)$ is 15 then $e_q(k)$ is 16, which is larger than x_H. In this case, the encoder puts out the codeword corresponding to x_H and generates $e_{q1}(k) = 16 - 10 = 6$ which is in the encoder output alphabet. Therefore, the encoder output consists of two codewords representing $x_H(10)$ and 6. If the input is $-7, e_q(k) = -6$ which is less than x_L. Thus the lossless encoder output consists of two symbols. One corresponding to the value of $x_L(-4)$ and

one corresponding to the value of -2. Note that if the input is 10 or -4 (i.e. x_H or x_L) then the output will be the sequence $10,0$ or $-4,0$.

One of the consequences of this type of encoding is that it can generate runs of x_L and x_H whenever the image contains a large number of edges. Fortunately the encoding scheme also provides a significant number of special symbols that can be used to encode these runs. For example, the sequence x_H followed by a negative value and the sequence x_L followed by a positive value would not occur in the normal course of events. These sequences can therefore be used to encode the runlengths of x_L and x_H. Consider for example a system in which Δ is 2 and x_L is -4. The output of the lossless encoder therefore corresponds to the values $-4,-2,0,2,4$. In the standard system a value of 4 is always followed by a value of 0 or 2. Similarly a value of -4 is always followed by a value of 0 or -2. Therefore the sequences $4-2$ and $-4+2$ can be used as special symbols to denote runs of 4 or -4. A simple strategy is to replace every two 4's (or -4's) after the initial 4 by a -2 (or 2). For example a value of 10 would still be represented by 4 4 2. However a value of 14 would be represented by 4 -2 2 instead of 4 4 4 2. Similarly a value of 18 would be represented by 4 -2 4 2 and a value of 22 would be represented by 4 -2 -2 2. For this particular scheme, a run of length n would be represented by $n - \lfloor \frac{n-2}{2} \rfloor$ codewords. When the size of the lossless encoder output is increased, the number of special symbols available also increases and the coding of the runs can be performed more efficiently.

These special sequences can also be used to signal a change of rate for applications in which the available channel capacity changes with time. The actual change can be accomodated by changing the stepsize and reducing the lossless encoder codebook size by the same amount. Several of the systems proposed above were simulated. The results of these simulations are presented in the next section.

Results

Four systems of the type described in the previous section have been simulated. Two of the systems simulated use a one tap fixed predictor, while the other two use a one pole four zero predictor with the zeros being adaptive. One of the systems in each case contains the lossless encoder followed by a runlength encoder while the other contains only the lossless encoder without the runlength encoder. The test images used were the USC GIRL image, and the USC COUPLE image. Both are 256 X 256 monochrome eight bit images and have been used often as test images. The objective performance measure were the Peak Signal to Noise Ratio (PSNR) and the Mean Absolute Error (MAE) which are defined as follows:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{<(s(k)) - \bar{s}(k)^2>}$$

$$\mathrm{MAE} = <|s(k) - \bar{s}(k)|>$$

where $< \cdot >$ denotes the average value.

Several initial test runs were performed using different number of levels, different values of x_L and different values of Δ to get a feel for the optimum values of the various parameters (Given x_L and Δ, x_H is automatically determined.). We found that an appropriate way of selecting the value of x_L was using the relationship

$$x_L = -\lfloor \frac{N-1}{2} \rfloor \Delta$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x, and N is the size of the alphabet of the lossless coder. This provides a symmetric codebook when the alphabet size is odd, and a codebook skewed to the positive side when the alphabet size is even. The zero value is always in the codebook.

As the alphabet size is usually not a power of two, the binary code for the output alphabet will be a variable length code. The use of variable length codes always bring up issues of robustness with respect to changing input statistics. With this in mind, the rate was calculated in two different ways. The first was to find the output entropy, and scale it up by the ratio of symbols transmitted to the number of pixels encoded. We call this rate the entropy rate, which is the minimum rate obtainable if we assume the output of the lossless encoder to be memoryless. While this assumption is not necessarily true, the entropy rate gives us an idea about the best we can do with a particular system. We will treat it as the lower bound on the obtainable rate. We also calculated the rate using a predetermined variable length code. This code was designed with no prior knowledge of the probabilities of the different letters. The only assumption was that the letters representing the inner levels of the quantizer were always more likely than the letters representing the outer levels of the quantizer. The code tree used is shown in Figure 3. Obviously, this will become highly inefficient in the case of small alphabet size and small Δ, as in this case, the outer levels x_L and x_H will occur quite frequently. This rate can be viewed as an upper bound on the achievable rate.

The results for the system with a one tap predictor and without the runlength encoder are shown in Tables 1 and 2. Table 1 contains the results for the COUPLE image, while Table 2 contains the results for the GIRL image. In the table R_L denotes the entropy rate while R_V is the rate obtained using the Huffman code of Figure 3. Recall that for image compression schemes, systems with PSNR values of greater than 35 dB are perceptually almost identical. As can be seen from the PSNR values in the tables there is very little degradation with rate, and in fact if we use the 35 dB criterion there is almost no degradation in image quality until the rate drops below two bits per pixel. This can be verified by the reconstructed images shown in Figure 4. Each picture in Figure 4 consists of the original image, the reconstructed image and the error image magnified 10 fold. In each of the pictures, it is extremely difficult to tell the source or original image from the reconstructed or output image. In fact, in the case of the image coded at rates above two bits per pixel it is well nigh impossible. This subjective observation is supported by the error images in each case which are uniform in texture throughout without any of the standard edge artifacts which can be usually seen in the error images for most compression schemes.

We can see from the results that if the value of Δ and hence x_L is fixed, the size of the codebook has no effect in on the performance measures. This is because the only effect of reducing the codebook size under these conditions is to increase the number of symbols transmitted. While this has the effect of increasing the rate, because of the way the system is constructed, it does not influence the resulting distortion. The drop in rate for the same distortion as the alphabet size increases can be clearly seen from the results in Tables 1 and 2.

Table 3 shows the decrease in rate when a simple runlength coder is used. The runlength coder encodes long strings of x_L and x_H using the special sequences mentioned previously. As can be seen from the results the improvement provided by the current runlength encoding scheme is significant only for small alphabets and small values of Δ. This is because it is under these conditions that most of the long strings of x_L and x_H are generated. However we are not as yet using many of the special sequences in the larger alphabet codebooks, so there is certainly room for improvement.

The one tap predictor was replaced with an adaptive ARMA predictor with a fixed pole and four adaptive zeros. The fixed pole was at a lag of 257 (pixel above) while the zeros were at lags of one, two, three and four. The adaption was performed using a sample LMS algorithm as follows. Let $\mathbf{B}_k$ be the vector of predictor coefficients at time k. The adaption algorithm was

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mu \epsilon_q(k) \mathbf{E}_k$$

where μ is the adaption stepsize and

$$\mathbf{E}_k = (\epsilon_q(k-1), \epsilon_q(k-2), \epsilon_q(k-3), \epsilon_q(k-4))^T .$$

The results from using this predictor are shown in Tables 4, 5 and 6. While there is some improvement in all cases, the results for the COUPLE image show a greater improvement than the results for the GIRL image. This can be explained by noting that the COUPLE image contains many more edges than the GIRL image. As the ARMA predictor tends to improve predictor performance when edges are encountered, the improvement in performance occurs in the image with more edges.

Conclusion

We have demonstrated a simple image coding scheme which is very easy to implement in realtime and has excellent edge preservation properties over a wide range of rates.

This system would be especially useful in transmitting images over channels where the available bandwidth may be vary. The edge preserving quality is especially useful in the encoding of scientific and medical images.

References

[1] N. S. Jayant and P. Noll, *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[2] A. K. Jain, "Image Data Compression: A Review," *Proc. IEEE*, vol. 69, pp. 349-389, Mar. 1981.

[3] C. C. Cutler, "Differential Quantization for Communication Signals," U.S. Patent 2 605 361, July 29, 1952.

[4] J. D. Gibson, "Backward Adaptive Prediction as Spectral Analysis Within a Closed Loop," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1166-1174, Oct. 1985.

[5] K. Sayood and S. M. Schekall, "Use of ARMA Predictors in the Differential Encoding of Images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-36, pp. 1791-1795, Nov. 1988.

[6] S. M. Schekall and K. Sayood, "An Edge Preserving DPCM Scheme for Image Coding," *Proc. 31st Midwest Symposium on Circuits and Systems*, St. Louis, pp. 904-907, Aug. 1988.

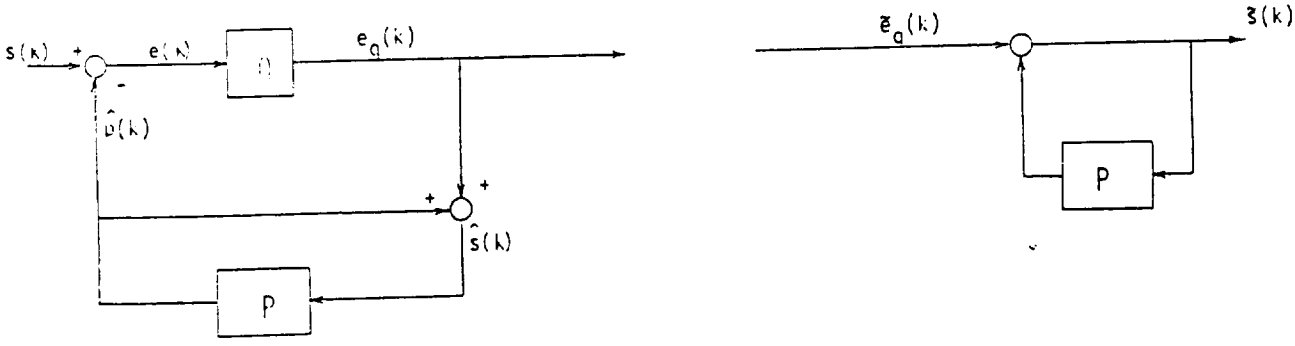

Figure 1. Block Diagram of a DPCM System

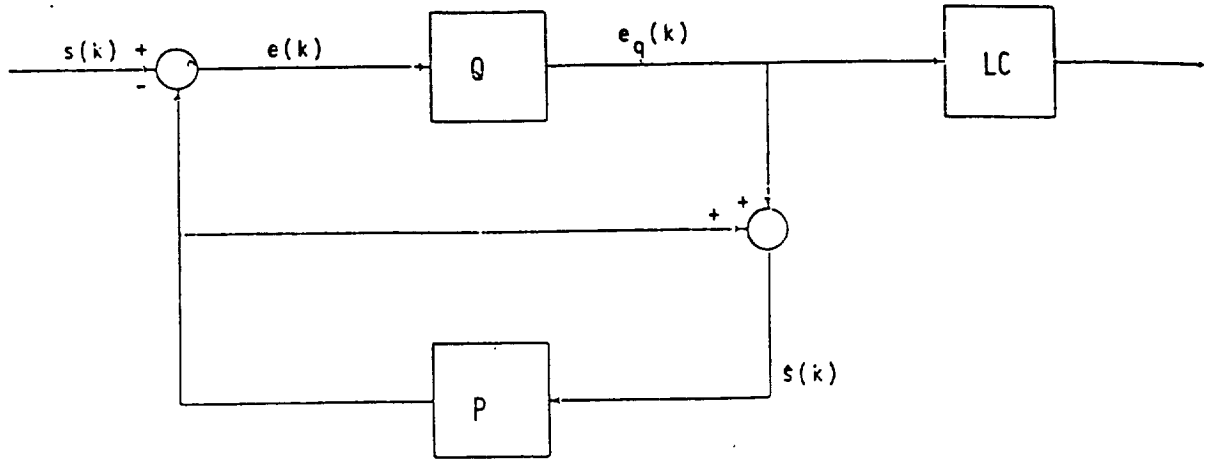

Fig. 2 Proposed Encoder Structure

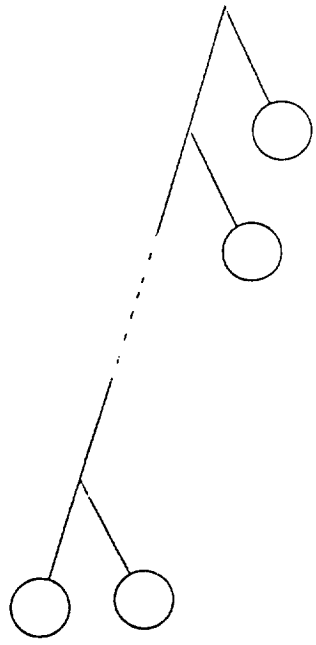

Fig. 3 Variable Length Code Tree

Delta	MAE	PSNR (dB)	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	0.5067	51.0830	6.1615	7.1418	4.9334	6.8635	4.4404	6.6884
4	1.4790	42.7898	3.8909	4.0587	3.3637	2.7982	3.1673	3.6939
6	2.4676	38.6565	2.9577	3.0137	2.6553	2.7729	2.5490	2.7023
8	3.3697	36.0009	2.4314	2.4972	2.2327	2.2756	2.1662	2.2267
12	5.1359	32.3682	1.8277	1.9800	1.7233	1.7963	1.6930	1.7669

Table 1: Performance results for the COUPLE image, alphabet size 3, 5 and 8.

Delta	MAE	PSNR (dB)	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	0.5067	51.0830	6.2821	7.8120	5.0554	7.4713	4.5635	7.1275
4	1.4790	42.7898	4.0088	4.3976	3.7414	4.0592	3.2668	3.8740
6	2.4676	38.6565	3.0819	3.2547	2.7570	2.9279	2.6468	2.8063
8	3.3697	36.0009	2.5543	2.6860	2.3272	2.3783	2.2617	2.2931
12	5.1359	32.3682	1.9426	2.1122	1.8046	1.8439	1.7786	1.8009

Table 2: Performance results for the GIRL image, alphabet size 3, 5 and 8.

Delta	Size = 3		Size = 5		Size = 8	
	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder
2	6.16	5.44	4.93	4.34	4.44	4.29
4	3.89	3.60	3.36	3.25	3.16	3.15
6	2.96	2.81	2.66	2.63	2.55	2.55
8	2.43	2.35	2.23	2.22	2.17	2.17
12	1.83	1.80	1.72	1.72	1.69	1.69

Table 3: Comparison of Entropy rates between system with Runlength (RL) Encoder and without RL Encoder for COUPLE image.

Delta	MAE	PSNR (dB)	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	1.59	46.11	4.71	5.00	3.94	4.77	3.63	4.69
4	2.00	40.71	3.02	3.04	2.70	2.82	2.50	2.76
6	2.96	37.42	2.33	2.38	2.14	2.18	2.09	2.13
8	3.86	35.11	1.94	2.05	1.81	1.87	1.79	1.83
12	5.61	31.79	1.49	1.72	1.42	1.56	1.41	1.55

Table 4: Performance results for COUPLE image with adaptive ARMA predictor.

Delta	MAE	PSNR (dB)	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	1.07	45.99	5.66	6.33	4.59	6.06	4.18	5.92
4	2.06	40.55	3.60	3.69	3.15	3.42	2.99	3.32
6	3.06	37.15	2.78	2.82	2.51	2.56	2.42	2.48
8	4.04	34.75	2.31	2.38	2.12	2.14	2.07	2.09
12	6.08	31.23	1.79	1.95	1.66	1.73	1.65	1.70

Table 5: Performance results for GIRL image with adaptive ARMA predictor.

Delta	Size = 3 Without RL Encoder	Size = 3 With RL Encoder	Size = 5 Without RL Encoder	Size = 5 With RL Encoder	Size = 7 Without RL Encoder	Size = 7 With RL Encoder
2	4.71	4.25	3.94	3.70	3.63	3.57
4	3.02	2.86	2.70	2.67	2.60	2.59
6	2.33	2.26	2.14	2.13	2.09	2.09
8	1.94	1.90	1.81	1.81	1.79	1.79
12	1.49	1.48	1.42	1.42	1.41	1.41

Table 6: Comparison of Entropy rates between systems with and without the Runlength Encoder for the COUPLE image.

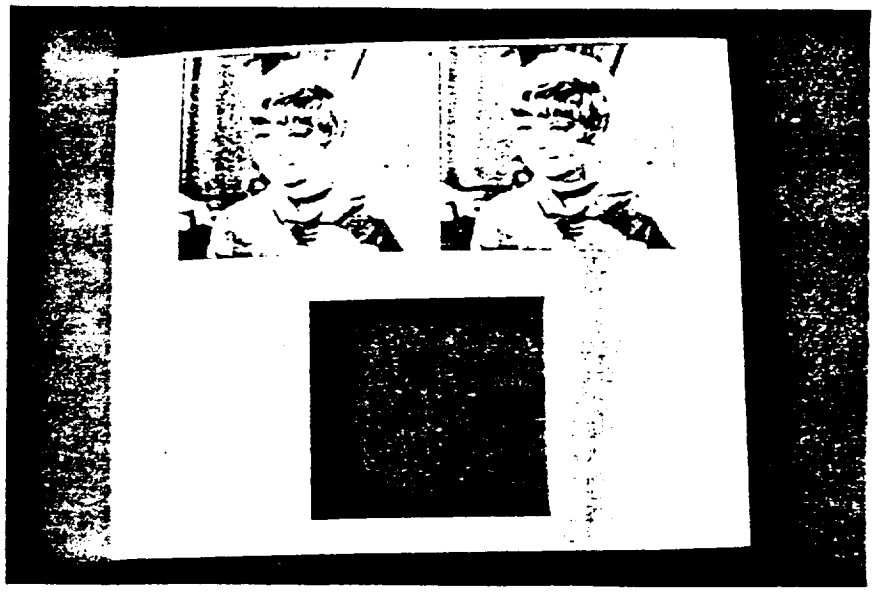

Figure 4(a). GIRL image coded at entropy rate of 1.7 bpp.

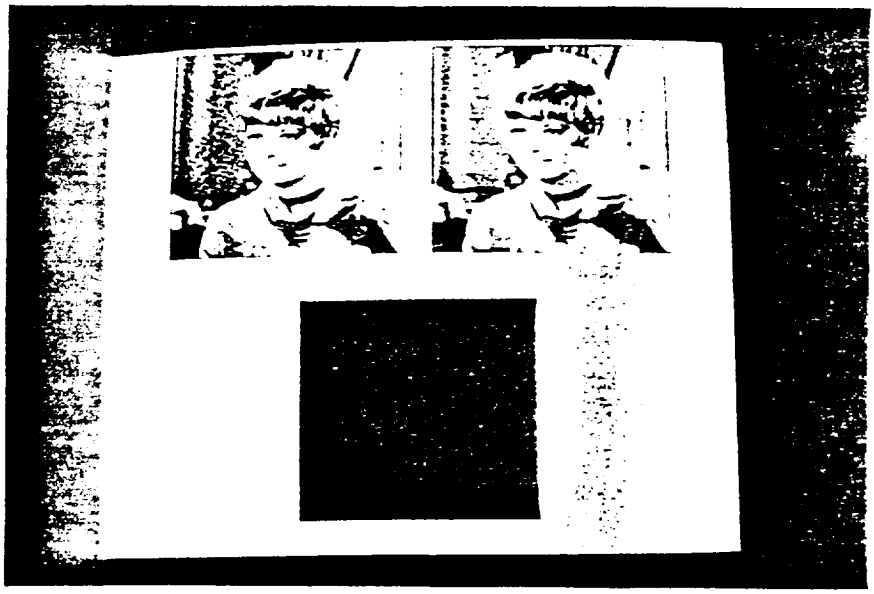

Figure 4(b). GIRL image coded at entropy rate of 1.5 bpp.

Appendix 2- Item 7