$$J_c = \left[\frac{E[z(n)^2]}{E[a(n)^2]}\right]^2 + \left[\frac{C_4[z(n)]}{C_4[a(n)]}\right]^2 - 3\left[\frac{C_4[z(n)]}{C_4[a(n)]}\right] \quad (10)$$

We consider the lattice-form IIR filter proposed in [2] which comprises N+1 sections, each of which is characterised by two identical reflection coefficients G_p, $0 \le p \le N$. The overall output $z(n)$ is weighted sum of the backward residuals as follows:

$$z(n) = \sum_{i=0}^{N} q_i(n) K_i(n) \quad (11)$$

where $\{q_i(n)\}$ and $\{K_i(n)\}$ are the backward residuals and the feedforward coefficients, respectively. The IIR lattice filter can also be described by the following section input/output equations:

$$q_m(n) = q_{m-1}(n-1) + G_m(n) f_{m-1}(n) \quad (12)$$

$$f_{m-1}(n) = f_m(n) - G_m(n) q_{m-1}(n-1) \quad (13)$$

These equations require that $f_0(n) = q_0(n)$ and the input is given by $f_N(n) = y(n)$, where $y(n)$ is the filter input. Define

$$\begin{aligned}\Theta(n) &= [\mathbf{G}(n)\ \mathbf{K}(n)]^T \\ &= [G_1(n)\ \cdots\ G_N(n)\ K_0(n)\ \cdots\ K_N(n)]^T \quad (14)\end{aligned}$$

$$\mathbf{Q}(n) = [q_0(n)\ \cdots\ q_N(n)]^T \quad (15)$$

$$\Psi(n) = [\gamma(n)\mathbf{K}(n)\ \mathbf{Q}(n)]^T \quad (16)$$

where $\gamma(n)$ is a matrix composed of the derivatives of $\mathbf{Q}(n)$ with respect to $\mathbf{G}(n)$. Applying the cost function J_c, we then obtain the lattice-form IIR blind algorithm described as

$$\begin{aligned}\Theta(n) = \Theta(n-1) - \mu\Big[&(4\gamma_1 + 36\gamma_3)E[z^2(n)] \\ &- 24\gamma_2 E[z^2(n)]\big[E[z^4(n)] - 3E^2[z^2(n)]\big]z(n) \\ &+ \big[8\gamma_2(E[z^4(n)] - 3E^2[z^2(n)]) - 12\gamma_3\big]z^3(n)\Big]\Psi(n)\end{aligned} \quad (17)$$

where

$$\gamma_{ij}(n) = \gamma_{i,j-1}(n-1) + G_j(n)\theta_{i,j-1}(n) + \delta_{ij} f_{j-1}(n) \quad (18)$$

$\gamma_{ij}(n)$ is the ijth component of $\gamma(n)$, δ_{ij} is the Kronecker delta function and

$$\theta_{i,j-1}(n) = \theta_{ij}(n) - G_j(n)\gamma_{i,j-1}(n-1) - \delta_{i,j} q_{j-1}(n-1) \quad (19)$$

These expressions require that $\theta_{i0}(n) = \gamma_{i0}(n)$ and $\theta_{iN}(n) = 0$.

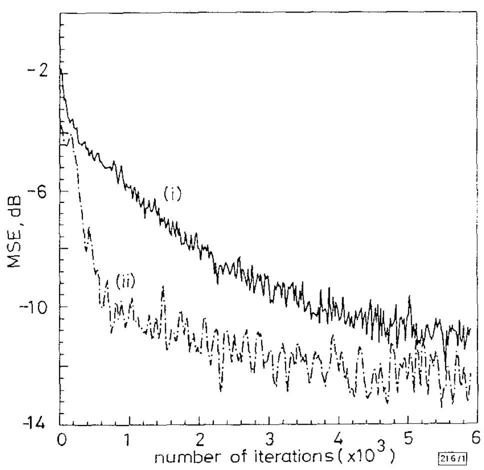

Fig. 1 *MSE performance of proposed lattice-form IIR blind equaliser and its FIR counterpart*

(i) FIR
(ii) lattice-form IIR

Simulation results: In this Section we present some Monte-Carlo simulations of the proposed blind algorithms. Binary PSK data are transmitted. The channel defined below is used in the simulations:

channel : $y(n) = a(n) + 0.9a(n-1)$

The step size μ is chosen to be 5×10^{-4}, the length N of the IIR equaliser is 6. The number of taps M for the FIR equaliser under comparison is 20. Fig. 1 shows the learning curves for the lattice-form IIR equaliser and its FIR version equaliser, respectively, for binary PSK data in the channel. These curves indicated that the proposed IIR blind algorithm not only has a faster convergence speed, but also yields a smaller steady state MSE than its FIR counterpart.

© IEE 1996
1 February 1996
Electronics Letters Online No: 19961263

Fang-Biau Ueng (*Department of Communication Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China*)

References

1 SHALVI, O., and WEINSTEIN, E.: 'New criteria for blind deconvolution of nonminimum phase systems (channel)', *IEEE Trans. Info. Theory*, 1990, pp. 312–321

2 SHYNK, J.J.: 'On lattice-form algorithms for adaptive IIR filtering'. Proc. Int. Conf. Acoust., Speech, Signal Processing, 1998, New York, pp. 1554–1557

EBTC: An economical method for searching the threshold of BTC compression

Ching-Yung Yang and Ja-Chen Lin

Indexing terms: Data compression, Image processing, Image coding

An economical method for obtaining a nearly-optimal threshold for the block truncation coding (BTC) image compression algorithm is presented. Simulation results show that the PSNR performance of the proposed economic BTC (EBTC) method is very close to that of the optimised BTC (OBTC) algorithm, but EBTC only takes about half the computation time required by the OBTC algorithm.

Introduction: Block truncation coding (BTC) [1, 2] is a simple and fast compression technique for digitised images. To reduce the mean square error (MSE) further, Kamel *et al.* [3] presented an algorithm which used a (partially) optimal threshold to quantise the block. However, as was pointed out in [4], the MSE generated by the method of Kamel *et al.* did not obtain a minimum value because the quantised error was neglected. Chen and Liu [4] suggested the optimised BTC (OBTC) algorithm to minimise the MSE, using a new threshold value searching policy. The PSNR performance of the OBTC algorithm is slightly better than that of [3], and the computation speed is also faster than that of [3], although the computation cost of [4] is still a heavy burden. In this Letter we develop an economical BTC (EBTC) algorithm whose PSNR is high (MSE is minimised) but the computation time is reduced significantly. Experiments showed that there is an effective tradeoff between the PSNR and computation time. We note that the nearly-optimal threshold is obtained by only searching a small portion of the input data for each block.

EBTC algorithm: Partition the image into blocks of size $n \times n$. For each block, let $G = \{g_i | i = 1, 2, ..., |G|\}$ be the $|G| = n^2$ given grey values to be split into two classes $H = \{g_i \ge Q\}$ and $L = \{g_i < Q\}$ where Q is a threshold value to be determined. The MSE of the block is defined by

$$MSE = \sum_{g_i \in H} (g_i - \bar{h})^2 + \sum_{g_i \in L} (g_i - \bar{l})^2 \quad (1)$$

where $\bar{h}$ and $\bar{l}$ are the average grey values of H and L, respectively. We try to obtain a Q for which the MSE is small. The procedure is as follows: The centroid O of G is evaluated by $O = (\Sigma_{i=1}^{|G|} g_i)/|G|$. The radius weighted mean [5] R of G is then evaluated by $R = (\Sigma_{i=1}^{|G|} r_i g_i)/\Sigma_{i=1}^{|G|} r_i$, with $r_i = |g_i - \bar{g}|$ for all i. If O equals R, then the value of R is assigned directly to the final threshold Q (that is, the threshold is obtained quickly without any searching operation). Otherwise, the data set is divided into two (temporary) subsets, say G_1 and G_2, by using the (temporary) threshold R. Let $G_k \in$

$\{G_1, G_2\}$ be the one with fewer points, and let $\bar{g}_k$ be the centroid of G_k. The final threshold Q is then determined by searching for a split point Q that minimises the MSE of the block. However, the searching range of Q is limited only to the interval between R and $(R+\bar{g}_k)/2$.

Simulation results and conclusion: Five 512 × 512 grey scale images with 8 bit/pixel were tested on a Sun SPARC 10 workstation. The peak signal-to-noise ratio (PSNR) is used as a measurement tool to gauge the image quality. The PSNR of the image is defined as

$$PSNR = 10\log_{10}\left(255^2 \Big/ \left(\sum_j MSE_j\right)\right) \qquad (2)$$

where MSE_j is the MSE of the jth block of the image. For comparison, the modified BTC (MBTC) [2] and OBTC algorithms were also implemented in our experiments. The optimal threshold value of the OBTC algorithm is obtained by computing the formula derived in [4], and then considering the candidate thresholds according to the histogram of the grey levels in the block. The PSNR and execution time produced by these three algorithms with block sizes 4 × 4 and 8 × 8 are provided in Tables 1 and 2, respectively. The MSEs (taking the average among all the blocks) for these algorithms are also given in Table 3. (Although we did not list the corresponding results of [3], we did observe that the PSNR of [3] was a little less than, but very close to that of [4]. Also, the algorithm in [3] was observed to be slower than that in [4].)

Table 1: PSNR and execution time for different algorithms when the block size is 4 × 4

Images	MBTC [2]		OBTC [4]		EBTC	
	PSNR	time	PSNR	time	PSNR	time
		s		s		s
Lena	33.85	0.48	34.47	4.09	34.44	2.09
Jet	32.65	0.48	33.44	3.93	33.41	2.18
Pepper	34.06	0.48	34.74	4.17	34.71	2.14
Scene	30.53	0.48	31.17	4.43	31.15	2.51
Monkey	27.66	0.48	28.10	4.88	28.08	3.14

Table 2: PSNR and execution time for different algorithms when the block size is 8 × 8

Images	MBTC [2]		OBTC [4]		EBTC	
	PSNR	time	PSNR	time	PSNR	time
		s		s		s
Lena	30.60	0.41	31.29	4.78	31.28	2.46
Jet	29.55	0.41	30.43	4.47	30.41	2.61
Pepper	30.36	0.41	31.28	4.89	31.26	2.50
Scene	27.49	0.41	28.24	5.52	28.23	3.08
Monkey	25.87	0.41	26.27	6.95	26.26	3.68

Table 3: MSE for different algorithms with block sizes 4 × 4 and 8 × 8

Images	MBTC [2]		OBTC [4]		EBTC	
	4 × 4	8 × 8	4 × 4	8 × 8	4 × 4	8 × 8
Lena	26.80	56.66	23.23	48.34	23.38	48.48
Jet	35.29	72.17	29.48	58.92	29.66	59.15
Pepper	25.55	59.80	21.85	48.48	22.00	48.62
Scene	57.52	116.04	49.62	97.45	49.90	97.68
Monkey	111.43	168.16	100.63	153.43	101.25	153.85

From these Tables, it is observed that the PSNR performance of the proposed EBTC method is better than that of the MBTC algorithm by a factor of 0.40–0.81dB. Conversely, the PSNR of the proposed method is very close to that of the OBTC algorithm, but the processing speed of the proposed method is nearly twice that of the OBTC algorithm. From Table 3, it can also be seen that the MSE produced by the proposed EBTC method is close to that produced by the OBTC algorithm Moreover, we also found that for ~73% of the total number of encoded blocks, the MSEs generated by our method within these blocks are optimal instead of nearly-optimal. We therefore suggest that the PSNR provided by the proposed EBTC method is higher than the PSNRs provided by most of the reported BTC algorithms, and is very close to that of the OBTC algorithm. In addition, the proposed method takes only ~50% of the computation time required by the OBTC algorithm

Acknowledgments: This project was supported by the National Science Council, Republic of China, under grant NSC85-2213-E009-111.

© IEE 1996
1 July 1996
Electronics Letters Online No: 19961267

Ching-Yung Yang and Ja-Chen Lin (*Institute of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 300, Republic of China*)

References

1 DELP, E.J., and MITCHELL, O.R.: 'Image compression using block truncation coding', *IEEE Trans.*, 1979, **COM-27**, (1), pp. 1335–1342

2 UDPIKAR, V.R., and RAINA, J.P.: 'Modified algorithm for block truncation coding of monochrome images', *Electron. Lett.*, 1985, **21**, (20), pp. 900–902

3 KAMEL, M., SUN, C.T., and GUAN, L.: 'Image compression by variable block truncation coding of monochrome images', *IEEE Trans. Signal Process.*, 1991, **39**, (1), pp. 208–212

4 CHEN, L.G., and LIU, Y.C.: 'A high quality MC-OBTC codec for video signal processing', *IEEE Trans. Circuits Syst. Video Technol.*, 1995, **4**, (1), pp. 92–985

5 LIN, J.C., CHOU, S.L., and TSAI, W.H.: 'Detection of rotational symmetric shape orientations by fold-invariant shape-specific points', *Pattern Recogn.*, 1992, **25**, (5), pp. 473–482

Fast algorithm for optimal bit allocation in a rate-distortion sense

Woo Yong Lee and Jong Beom Ra

Indexing terms: Image coding, Video coding, Rate distortion theory

To improve the quality of reconstructed images for a given bit rate constraint, the assigned bits must be distributed efficiently using a set of admissible quantisers so that source distortion can be minimised. However, the optimal bit allocation scheme is usually not practical due to its large computational burden. The authors propose a new fast algorithm which needs less computing time than required by existing fast algorithms.

Introduction: Optimal bit allocation for source coding involves minimising source distortion, subject to a given bit budget. In image coding, the intention is to distribute the assigned bits efficiently among image blocks by using proper quantisers. To meet this requirement, a dynamic programming method is suggested, based on rate-distortion characteristics [1]. This optimal method, however, requires large computational complexity. Even though several fast algorithms [2–4] have been proposed to reduce the computational complexity, they are not still fast enough for practical applications related to video coding.

In this Letter, we propose a new fast algorithm for optimal bit allocation, which is based on the bi-directional prediction of the Lagrange multiplier λ for a given bit rate R_t. The proposed algorithm reduces the computational complexity substantially, compared with previous fast algorithms. It could be useful for asymmetric coding applications such as CD-ROM, video on demand (VOD) etc., where high coding performance is essential and real-time encoding is not imperative.