Designing JPEG Quantization Matrix Using Rate-Distortion Approach and Human Visual System Model

W.C.Fong, S.C.Chan and K.L.Ho
Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong.

Abstract

JPEG is an international standard for still image compression [1] . The JPEG baseline algorithm allows users to supply the custom quantization table and Huffman table to control the compression ratio and the quality of the encoded image. Methods for determining the quantization matrix are usually based on i) rate-distortion theory [2,8] and ii) spatial masking effects of the human visual system [9]. In [2], Wu and Gersho proposed a recursive algorithm for generating picture-adaptive quantization tables based on rate-distortion approach but the complexity of the encoding algorithm is rather high. In this paper, we propose improvements to the Wu-Gersho's algorithm and a new bit allocation algorithm. Simulation results show that our new algorithm is superior to the Wu-Gersho's algorithm in terms of speed and peak signal to noise ratio (PSNR). Moreover, by incorporating the Human Visual System (HVS), our proposed coder can encode images with better visual quality.

1. Introduction

JPEG is an international standard for still image compression [1]. The baseline JPEG algorithm allows users to supply custom quantization table and Huffman table to control the compression ratio and the quality of the encoded image. The quantization table given in the JPEG recommendation [1] is often used. Methods for determining the quantization matrix are usually based on i) rate-distortion theory [2,8] and ii) spatial masking effects of the human visual system (HVS) [9]. In [2], Wu and Gersho proposed a recursive algorithm for generating picture-adaptive quantization tables based on rate-distortion approach. The complexity of the encoding algorithm is rather high because it has to calculate the change of distortion and bits for every possible changes of stepsizes in the most effective subband at each iteration. It had also been pointed out in [2] that better result can be obtained by updating the efficiency λ_k at each iteration. However, the arithmetic complexity is extremely high. It is the purposes of this paper to reduce the arithmetic complexity of these bit-allocation algorithms and to incorporate the HVS to improve the visual quality of the encoded images.

2. The Wu-Gersho's Algorithm

The baseline JPEG coder is a transform coder consisting of: (8×8) DCT transformation, quantization, and runlength Huffman coding. The input image is grouped into (8×8) blocks and transformed by the DCT. The transformed coefficients are uniformly quantized by step sizes specified in the quantization matrix. The DC coefficients are differentially coded and the AC coefficients are ordered into the "zig-zag" sequence. Each nonzero AC coefficient is represented by its category and its value within that category. The category and the run of zero values preceding it is jointly entropy coded using the given Huffman table.

In the Wu and Gersho's algorithm, the quantizer step sizes $\{Q_k; k = 0,\ldots,63\}$ is adjusted to minimize the overall distortion:

$$D(\mathbf{Q}) = \sum_{n=1}^{N} \sum_{k=0}^{63} D_{n,k}(Q_k) \tag{1}$$

subjected to the bit rate constrain:

$$R(\mathbf{Q}) = \sum_{n=1}^{N} R_n(Q_0,\ldots,Q_{63}) \leq B \tag{2}$$

where $D_{n,k}(Q_k)$ is the distortion in the k-th DCT coefficient of the n-th block if it is quantized with step size Q_k, $R_n(Q_0,\cdots,Q_{63})$ is the number of bits generated in coding the n-th block with the quantization table $\{Q_0,\cdots,Q_{63}\}$, and $\mathbf{Q} = \{Q_0,\ldots,Q_{63}\}$ is the vector of quantization stepsizes. Starting from an initial quantization table of large step sizes, the algorithm decreases the step size in one entry of the quantization table at a time until a target bit rate is reached. In each iteration, the problem to be solved is:

$$\max_k \max_q \frac{-\Delta D(\mathbf{Q})|_{Q_k \to q_k}}{\Delta R(\mathbf{Q})|_{Q_k \to q_k}} \tag{3}$$

where $\Delta D(\mathbf{Q})|_{Q_k \to q_k}$ and $\Delta R(\mathbf{Q})|_{Q_k \to k}$ are respectively the change in distortion and the change in overall bit rate when the k-th entry of the quantization table, Q_k, is replaced by q_k. The maximization process can be split into two parts. The first part computes the efficiency:

$$\lambda_k = \max_q \frac{-\Delta D(\mathbf{Q})|_{Q_k \to q_k}}{\Delta R(\mathbf{Q})|_{Q_k \to q_k}} \tag{4}$$

for all values of k and the second part solves:

$$\lambda = \max_k \lambda_k \tag{5}$$

The algorithm is summarized as follows:

1. Initialize the quantization table by:
$$Q_k^{(0)} = \begin{cases} 16 & for \quad k = 0, \\ Q_{\max} & for \quad k = 1,\dots,63 \end{cases}$$
2. Initialize tables of λ_k and $\tilde{Q}_k$ by searching q_k in $\{1,\dots,Q_{\max} - 1\}$ to solve (4) for k=1,...,63.

At iteration ℓ, we perform the following:

3. Search k in $\{1,\dots,63\}$ for p to solve (5).
4. Update the stepsizes by setting $Q_p^{(\ell)} = \tilde{Q}_p$.
5. Update λ_p and $\tilde{Q}_p$ by searching q_p in $\{1,\dots,Q_p^{(\ell)} - 1\}$ to solve (4).
6. Repeat Steps 3 to 5 until $R(\mathbf{Q}^{(\ell)}) \le B$.

The maximum step size, Q_{Max}, is set to 128.

It can be seen that whenever one entry in the quantization table is altered, the bits due to runlength coding will also be affected. In principle, all the 63 λ_k's have to be computed again using the new quantization step sizes to select the most efficient subband for allocating bits. However, this is extremely time consuming. Even if we update only the most efficient λ_p as in the Wu and Gersho's algorithm, the computational complexity is still very high. This is because we have to calculate $\Delta D(\mathbf{Q}^{(\ell)})\big|_{Q_p \to q_p}$ and $\Delta R(\mathbf{Q}^{(\ell)})\big|_{Q_p \to q_p}$ for $q_p = \{1,\dots,Q_p^{(\ell)} - 1\}$ at each iteration (step 5).

3. The Proposed Algorithm

Suppose that we have quantized the picture with the new stepsize $\mathbf{Q}^{(\ell)}$ and the resulting bit rate and distortion are respectively, $R(\mathbf{Q}^{(\ell)})$ and $D(\mathbf{Q}^{(\ell)})$. We want to compute:

$$\lambda_k = \max_{q_k} \frac{-\Delta D(\mathbf{Q}^{(\ell)})|_{Q_k^{(\ell)} \to q_k}}{\Delta R(\mathbf{Q}^{(\ell)})|_{Q_k^{(\ell)} \to q_k}}, q_k \text{ in } \{1,\dots,Q_k^{(\ell)} - 1\}$$

The numerator is simple to compute using:

$$-\Delta D(\mathbf{Q}^{(\ell)})\Big|_{Q_k^{(\ell)} \to q_k} = \sum_{n=1}^{N} \{D_{n,k}(Q_k^{(\ell)}) - D_{n,k}(q_k)\}$$

Calculation of $\Delta R(\mathbf{Q}^{(\ell)})|_{Q_k^{(\ell)} \to q_k}$ is somewhat complicated. Consider the k transform coefficient, $T_k(n)$, in block n. Suppose initially that the quantized value, $\hat{T}_k(n)$, is nonzero. When Q_k is changed from $Q_k^{(\ell)}$ to q_k, $T_k(n)$ will either be quantized to a zero or nonzero value depending on the value of q_k. Let the resulting quantization vector be $\hat{\mathbf{Q}}^{(\ell)}$. In the first case, the bits required to encode $T_k(n)$, $R_{\hat{\mathbf{Q}}^{(\ell)}}(n)$, can be written as:

$$R_{\hat{\mathbf{Q}}^{(\ell)}}(n) = R_{\hat{\mathbf{Q}}^{(\ell)}}^{run\text{-}cat}(n) + R_{\hat{\mathbf{Q}}^{(\ell)}}^{levels}(n) \tag{6}$$

where $R_{\hat{\mathbf{Q}}^{(\ell)}}^{run\text{-}cat}(n)$ is the length of the Huffman code to represent the zero-run and category for $\hat{T}_k(n)$ with stepsize vector $\hat{\mathbf{Q}}^{(\ell)}$, $R_{\hat{\mathbf{Q}}^{(\ell)}}^{levels}(n)$ is the number of bits needed to represent $\hat{T}_k(n)$ within that category with stepsize vector $\hat{\mathbf{Q}}^{(\ell)}$. Using the given Huffman table and the quantized transform coefficients, the change in bits required in block n can be computed as follows:

$$\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)} \to q_k}(n) = \{R_{\hat{\mathbf{Q}}^{(\ell)}}^{run-cat}(n) - R_{\mathbf{Q}^{(\ell)}}^{run-cat}(n)\}$$
$$+\{R_{\hat{\mathbf{Q}}^{(\ell)}}^{levels}(n) - R_{\mathbf{Q}^{(\ell)}}^{levels}(n)\} \tag{7}$$

If the coefficient is quantized to zero, the change in bits can also be computed similarly. To simplify the operations, we can maintain a data structure that records those nonzero transform coefficients in each block after each iteration. Finally, we have:

$$\Delta R(\mathbf{Q}^{(\ell)})|_{Q_k^{(\ell)} \to q_k} = \sum_{n=1}^{N} \Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)} \to q_k}(n) \tag{8}$$

Instead of initializing the stepsizes to their maximum values, we can initialize them to the smallest values.

When the stepsizes are increased, certain subbands will be quantized to zero and will not participate in the bit allocation. Also we can limit the search range for q_p in updating $\tilde{Q}_p$ to:

$$\left\{Q_p^{(\ell)} - W, \ldots, Q_p^{(\ell)} - 1\right\} \qquad (9)$$

Experiments show that a look ahead window W of 6 to 10 is sufficient. To reduce the arithmetic complexity further, we propose a method to approximate the calculation of λ_k. The method we used is to estimate $\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n)$. Consider the first term in Eqn (7). It is observed that the length of the Huffman code in [1] is 16 except when the coefficients are of small amplitude (low category value). Therefore we can assume that the length of the code is 16 when the coefficients are of large amplitude. On the other hand, for coefficients with small amplitude, we estimate the length of the code with a mean value. Therefore, the change in bits, $\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n)$, can solely be estimated by the coefficients in that subbands. If the Huffman table for the number of zero values and category is decoupled or separable, then the estimation will be more accurate. An example is the following Huffman table:

Bit Count	Length of Zero-run		Category	Binary Code
6	1 to 16		1	010
12	17 to 32		2	011
18	33 to 48		3	1000
....	....		4	1001
			5	0010
			6	1011
			......	......
			10	1111.

Table 1

As the codes for the zero-run and the category are separable, Eqn (7) can be rewritten as:

$$\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n) = \{R^{run}_{\hat{\mathbf{Q}}^{(\ell)}}(n) - R^{run}_{\mathbf{Q}^{(\ell)}}(n)\}$$

$$+\{R^{cat}_{\hat{\mathbf{Q}}^{(\ell)}}(n) - R^{cat}_{\mathbf{Q}^{(\ell)}}(n)\} + \{R^{levels}_{\hat{\mathbf{Q}}^{(\ell)}}(n) - R^{levels}_{\mathbf{Q}^{(\ell)}}(n)\} \quad (10)$$

where $R^{run}_{\hat{\mathbf{Q}}^{(\ell)}}(n)$ and $R^{cat}_{\hat{\mathbf{Q}}^{(\ell)}}(n)$ stand respectively for the code length to represent the zero-run and category for block n. The last two terms are functions of the quantizer levels, q_k, only. It can be seen from Table 1 that the length of the run-length code is always 6 when the number of the zeros is less than 16. Since the chance of having a long run of zeros is usually low except for end-of-block, we can assume that the length of the code is 6. Therefore, $\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n)$ can be estimated by the coefficients $T_{k-1}(n)$, $T_k(n)$ and $T_{k+1}(n)$. To further reduce the amount of computation, we can update the efficiency λ_k for part of the subbands only. As the change in λ_k will be smaller when k is farther apart from subband p (in zigzag scanning order), it makes sense to update those subbands that are adjacent to p. In this work, we update λ_{p-1}, λ_p, and λ_{p+1}. The DC subband also participates in the bit allocation and the maximum step size is 255.

If we update $\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n)$ in each iteration, $R(\mathbf{Q}^{(\ell)})$ can be computed exactly and the algorithm can stop at a given compression ratio or quality (PSNR). If we choose to estimate $\Delta R_{\mathbf{Q}^{(\ell)}:Q_k^{(\ell)}\to q_k}(n)$, then the calculated bits, $\hat{R}(\mathbf{Q}^{(\ell)})$, will deviate from the true value and we can only stop at a given PSNR. Fortunately, we can requantize the picture to calculate the true bits $R(\mathbf{Q}^{(\ell)})$ when $\hat{R}(\mathbf{Q}^{(\ell)})$ reaches the given bit budget B. Since $R(\mathbf{Q}^{(\ell)})$ is larger than B, we can start the estimation again until $R(\mathbf{Q}^{(\ell)})$ become sufficiently close to B. This enables us to correct the error in the estimation. In fact, only a few iterations are needed.

4. Human Visual System

Various HVS models have been proposed in the literature [3-7]. Mannos and Sakrison [3] were the first to model the HVS as a nonlinear point transformation followed by the MTF. Nill [5] modified the HVS and applied it to the DCT domain. Ngan et al [6] used Nill's HVS model with a different MTF:

$$H(f) = (0.31 + 0.69f)\exp(-0.29f)$$

where f is the radial frequency in cycles/degree (CPD) of the visual angle subtended. Chitprasert and Rao [4] used a similar MTF:

$$H(f) = 0.246(0.1 + 0.25f)\exp(-0.25f)$$

and obtained the HVS model through the convolution-multiplication property of the DCT. Perkins and Lookabaugh [7] proposed a quadratic fit to the measured data of contrast sensitivity by Campbell and Robson and modeled the angular sensitivity as a quadratic polynomial:

$$S_\lambda(x) = -4.47x^2 + 8.86x - 4.39$$

$$S_A(\alpha) = -0.0222\alpha - 0.001\alpha^2$$

$S_\lambda(x)$ and $S_A(x)$ are respectively the normalized sensitivity in dB and the angular sensitivity in dB, x is the natural logarithm of the number of cycles per degree appearing at the retina, f, α is the deviation in degrees of the gratings orientation from the nearest horizontal or vertical axis. Here, we use the formula in [4] to compute f for the (n_1,n_2) subband: $f = \frac{\sqrt{n_1^2+n_2^2}}{2N} f_s$

where f_s is the sampling density and is taken as 64 here, N=8 is the transform size. α is simply given by:

$$\alpha = \min\{\theta, 90-\theta\},\ \theta = \tan^{-1}\left(\frac{n_2}{n_1}\right).$$

The relative weightings are then given by: $\log_{10} w(n_1,n_2) = S_\lambda(x(n_1,n_2)) + S_A(\alpha(n_1,n_2))$. In this work, we make use of the HVS models in [4] and [7] to obtain the weighting function for each subband. The distortion that we used is: $D_w(\mathbf{Q}) = \sum_{n=1}^{N}\sum_{k=0}^{63} w_k D_{n,k}(Q_k)$.

5. Simulation Results

Experiments are performed on the 512×512 greyscale images Lenna and Baboon. The algorithms that we have tested are:

1. **WG**: Modified Wu-Gersho's algorithm with $W=6$,
2. **FCH**: Proposed algorithm with $W=6$,
3. **FCH-HVS1**: Proposed algorithm ($W=6$) with HVS model in [7],
4. **FCH-HVS2**: Proposed algorithm ($W=6$) with HVS model in [4],
5. **JPEG-D**: JPEG baseline algorithm with quantization tables in [1].

Figure 1 and 2 show the PSNR of encoding the lenna and baboon images at various bit rate by the JPEG default matrix (JPEG-D), our proposed algorithm without HVS weighting (FCH), and the Wu and Gersho's algorithm (WG). It can be seen that the performance of the our algorithm is slightly better than the Wu and Gersho's algorithm. The improvement over the JPEG default quantization table ranges from 0 to 3 dB. The perceptual quality of the algorithms are similar. Figure 3 shows the Baboon image compressed to 0.35 bpp using algorithms FCH, FCH-HVS1, FCH-HVS2 and JPEG-D. It can be seen that algorithm FCH-HVS1 has best visual quality followed by algorithms FCH-HVS2, FCH and JPEG-D. The computation time of the algorithms depend on the compression required. At 0.4 bpp, the computation time of algorithm FCH requires 5 minutes on a Pentium 100 Computer which is about two times faster than the modified Wu-Gersho's algorithm (WG). It is expected that the execution time can further be reduced after careful optimization. For algorithm 2 to 4, the quantization table are estimated using table 1 while the actual encoding is performed using JPEG default Huffman table. The performance is slightly affected but the overall performance is still very good. If the custom Huffman table is required, we can first estimate the quantization table using the default Huffman table or the one in table 1 and collect the statistics of the run and category to generate the desired Huffman table.

References

[1] ISO 10918: Digital compression and coding of continuous-tone still images.

[2] S. W. Wu and A. Gersho, "Rate-constrained picture-adaptive quantization for JPEG baseline coders," IEEE ICASSP-93, vol. V, pp. 390-392.

[3] J. L. Mannos et al, "The effect of a visual fidelity criterion on the encoding of images," IEEE Trans. IT-20, pp. 525-536, July 1974.

[4] B. Chitprasert et al, "Human visual weighted progressive image transmission," IEEE Trans. COM-38, no. 7, pp. 1040-1044, July 1990.

[5] N. B. Nill, "A visual model weighted cosine transform for image compression and quality assessment," IEEE Trans. COM-33, pp. 551-557, June 1985.

[6] K. N. Ngan, et al, "Cosine transform coding incorporating human visual system model," IEEE Trans. ASSP-37, No. 11, Nov. 1989, pp. 1743-1749.

[7] M. G. Perkins et al, "A Psychophysically justified bit allocation algorithm for subband image coding systems," ICASSP-89, pp. 1815-1818.

[8] K. Ramchandran and M. Vetterli, "Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," IEEE Trans. Image Proc. vol. 3, pp. 700-704, Sept. 1994.

[9] H. A. Peterson et al, "Quantization of color image components in the DCT domain," SPIE vol. 1453, Human Vision, visual processing, and digital display II (1991) pp. 210-222.

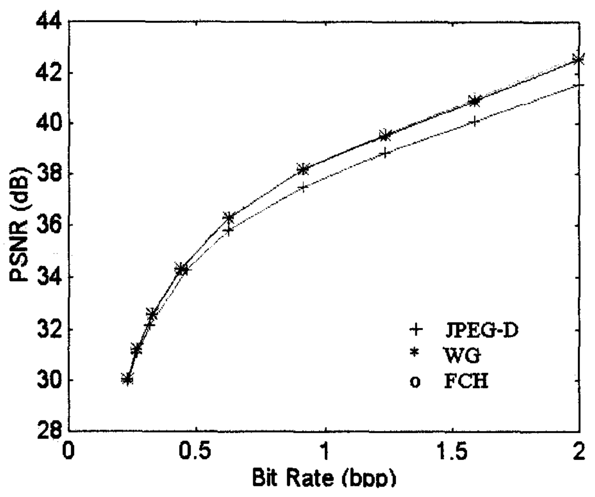

Fig. 1. Performance Comparison between **JPEG-D**, **FCH** and **WG** algorithm for lenna.

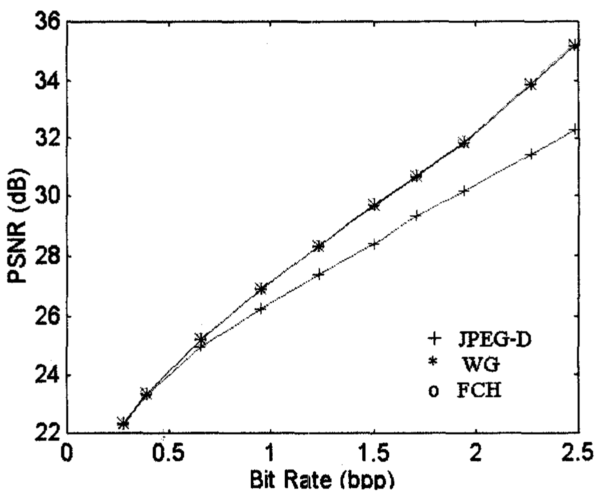

Fig.2. Performance Comparison between **JPEG-D**, **FCH** and **WG** algorithm for baboon.

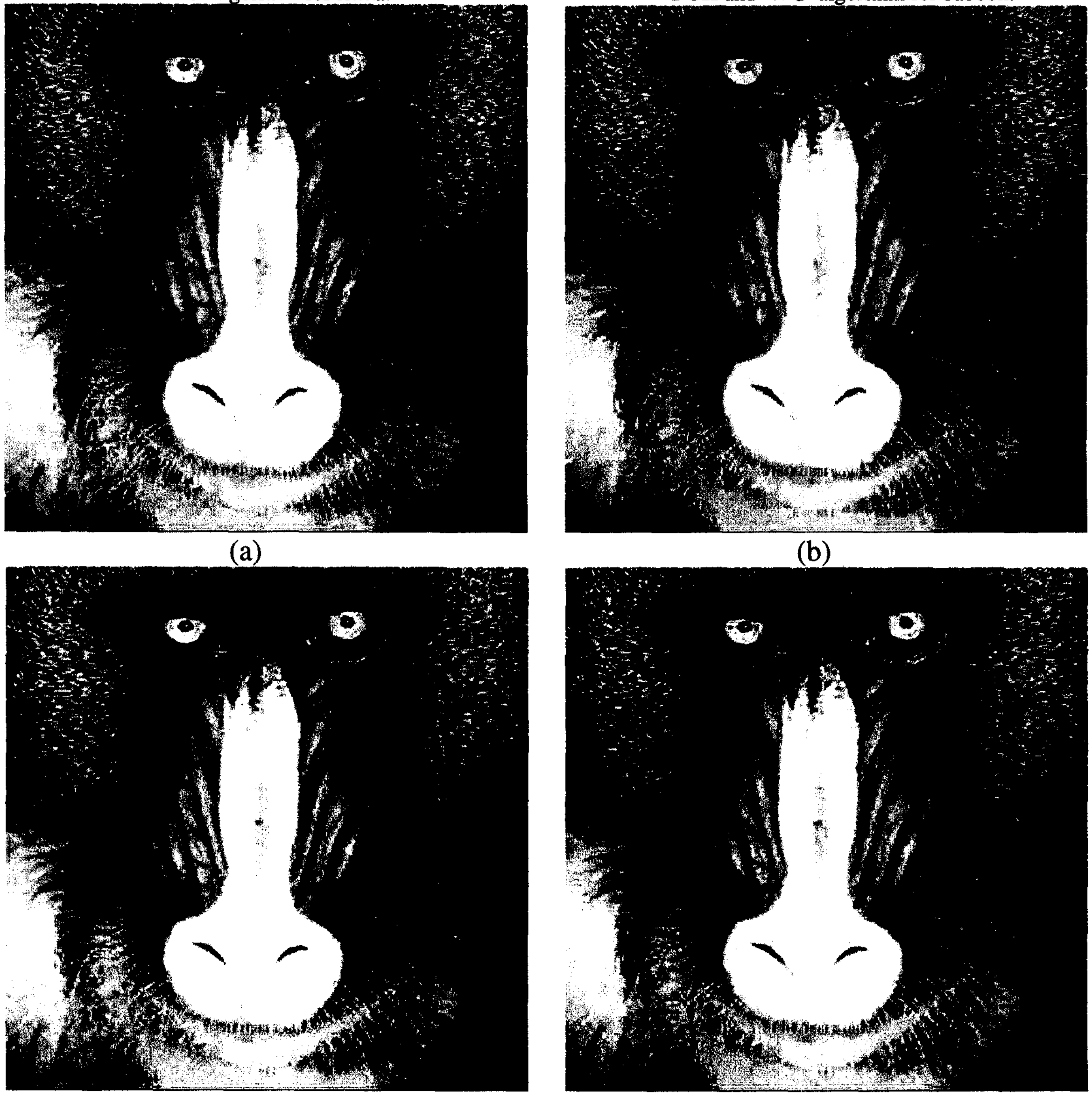

Fig.3. Baboon image compressed to 0.35 bpp using algorithm (a) **JPEG-D** (b) **FCH** (c) **FCH-HVS1** (d) **FCH-HVS2**