DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

Fast JPEG rate control

Valery Anisimovskiy*, Sergey Zavalishin*, Ilya Kurilin*
****Samsung R&D Institute Russia, 12, Dvintsev str., Moscow, Russia 127018***

Abstract

We propose a new rate control method for JPEG image compression. Similarly to the vast majority of JPEG rate control approaches, our method solves the task of JPEG rate control by generating custom JPEG quantization tables (QTs). The method includes adaptive bit count predictor training stage, optionally followed by rate-distortion optimization (RDO) stage. The training of the adaptive bit count predictor is based on a linear prediction model using either coefficient-wise average entropy or ρ-parameter. The trained predictor is subsequently used by the RDO stage to estimate JPEG bit count resulting from the application of particular QT, thereby substantially speeding up the RDO algorithm. For RDO stage we use one of the two algorithms: Wu-Gersho algorithm or RD-OPT algorithm. The resulting hybrid design combines strong points of each of the utilized approaches, while mitigating its shortcomings, thereby providing a good trade-off between computational complexity, rate control accuracy and reconstructed image quality.

Introduction

The Joint Photographic Experts Group (JPEG) image compression algorithm [1] remains to be one of the most popular and widespread image compression formats. However, many JPEG implementations lack a rate control (RC) capability, i.e., producing an output file having size of (or sufficiently close to) the user-defined value. The need for RC-enabled JPEG compression frequently arises in many areas, for example, when a picture taken by a camera is compressed into JPEG file and is intended for storing on a flash card of limited storage capacity or for transferring through a network channel of limited bandwidth capacity.

Various methods of JPEG RC have been proposed previously, but most of them suffer from either prohibitively high computational complexity or poor RC accuracy.

Our method leverages strong points of the most efficient JPEG RC approaches – parametric modeling, local search and dynamic programming – while trying to mitigating their shortcomings.

Keywords: JPEG coding, image compression, rate control, dynamic programming, rate-distortion optimization

Problem formulation

Primary means of rate control, provided by the JPEG image compression standard, is setting quantization tables (QTs) which are used by the JPEG compressor to quantize coefficients of Forward Discrete Cosine Transform (FDCT) applied to color planes of the input image.

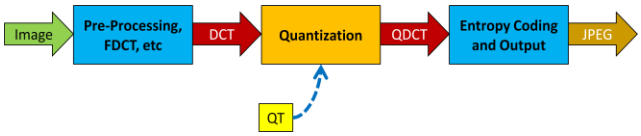

Figure 1. JPEG encoder pipeline overview

JPEG encoding pipeline is depicted in Figure 1: the input image is optionally pre-processed (e.g. by RGB-to-YCbCr color space conversion), transformed by FDCT to frequency domain, the resulting DCT coefficients are quantized using QTs to produce quantized DCT coefficients, which are entropy coded and output to JPEG file.

The JPEG standard allows setting up to three different QTs (one for each color component, i.e. Y, Cb and Cr), each QT being an 8x8 matrix of integer values ranging from 1 to 255 (the range restricted by baseline JPEG). By varying QT, one is able to set a trade-off between output bit rate and reconstructed image quality: larger quantizers in the QT lead to more coarse quantization, thus smaller quantized DCT coefficients (and smaller bit rate) at the cost of worse image quality (due to higher distortion of coarse quantization). So, the task of RC is usually formulated as Rate-Distortion Optimization (RDO) problem: find QT (or a set of QTs in case of multi-component image) causing minimum quantization distortion while keeping bit rate less than or equal to the user-configured target value (which is a constrained optimization problem). Since the entire configuration space for single QT contains $255^{64} \approx 10^{153}$ different QTs, full search solution is unacceptable for any practical RC implementation.

Related works

Several approaches to RDO problem have been developed, out of which the most interesting ones could be roughly divided into four classes: global search, parametric QT modeling, local search and dynamic programming.

Under the global search approach, the global search over the entire configuration space is performed using, e.g., genetic algorithm [2] or simulated annealing [3]. Despite the fact that the globally optimal solution (or sub-optimal one sufficiently close to it) is usually found by this approach, the computational complexity is typically prohibitively high.

For the parametric QT modeling, a simple parametric model is adopted for QT so that the number of model parameters is significantly less than the number of quantizers in the QT [4][5][6][7][8][9]. Then, RD-optimal parameter values are determined using some global or local search technique. Essentially, this approach reduces the dimensionality of the search space thereby significantly simplifying the problem and reducing computational complexity. The search in the reduced dimensionality space may be done off-line using, e.g., simulated annealing [4] in which case the algorithm is not adaptive to the input image. To make the algorithm image-adaptive without

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

suffering from prohibitive computational complexity (due to usage of complex multi-dimensional search techniques), the common solution is to adopt single-parameter model. The simplest way to do so is to define QT as the scaled version of the pre-defined default QT, (e.g. as in Annex K in [1]). The scaling factor value corresponding to the required bit rate may be found using predictive operation (via scaling-factor-to-bitrate mapping derived from some efficient heuristics [5][6]) or reactive operation [7] (iterative scaling factor correction using output bit rate from each entropy encoding iteration). Additionally, the default QT can be derived based on human visual system (HVS) properties or specific application requirements, like in [6], [8] and [9]. The strong advantage of this approach is low computational complexity, while the drawback is worse quality (resulting from poor image adaptability).

According to the local search approach, the QT search is performed in high-dimensional search space using local search techniques, i.e. starting at some initial QT (which is usually chosen to be the worst-quality lowest-bitrate QT = { 255, 255, …, 255 }), local search proceeds by modifying QT by small steps directed to optimum [10][11][12][13]. One of the most successful algorithm of this kind was proposed in [10] and is referred to as Wu-Gersho algorithm hereinafter. To improve convergence of this algorithm, clever initial QT is proposed in [11] as well as entropy-based bit count estimation (instead of running costly entropy encoding) to decrease the algorithm complexity. In [12], a more sophisticated bit allocation scheme is used to improve quality of the Wu-Gersho algorithm along with HVS incorporation to improve perceived quality. In [13], joint optimization of run-length coding, Huffman coding and QT selection (based on Lagrangian multiplier method) is used to improve both quality and complexity of local search as compared to the Wu-Gersho algorithm. Although local search-based algorithms provide good performance in terms of both quality and complexity, the most efficient algorithms (Wu-Gersho algorithm and its derivatives, the joint optimization algorithm of [13]) still have quite high complexity since they perform many QT search steps and, in some variants, additional costly Lagrangian λ multiplier estimation.

In the dynamic programming approach, bit rate and distortion are decomposed into sums over 64 DCT sub-bands, so that the RDO problem of constrained optimization turns out to have optimal substructure. In that form, the problem is solved by dynamic programming (DP) technique, based on finding optimal path in the trellis formed by all possible quantizer values for each DCT sub-band [14] (RD-OPT algorithm). The original RD-OPT algorithm was later extended to include optimization of global DCT coefficient thresholding [15], while similar DP-based framework along with Lagrangian multiplier method was used in [16] to provide optimization of local DCT coefficient thresholding. Another extension in [17] utilized entropy-constrained vector quantization for the same problem, thereby significantly increasing computational complexity. Although complexity of DP-based algorithms is significantly less than that of the global search algorithms (and even many local search algorithms, e.g. the Wu-Gersho algorithm), the bit rate decomposition which motivates the optimal substructure of the RDO problem (which is required for the problem to be solved by DP) is in fact approximate and, as we show later, in its original form (based on coefficient-wise average entropy) has quite bad accuracy.

Description of method

Algorithm overview

Our method utilizes strong points of several previously known approaches – parametric modeling, local search and dynamic programming – while trying to mitigate their shortcomings. Since the weakest point of the most efficient approaches is either high computational complexity caused by many costly entropy coding runs or poor RC accuracy caused by poor accuracy of bit count estimation, the key component for creating fast and accurate RC is a low complexity bit count estimator providing good accuracy. To this end, we've developed the bit count predictor module, which allows low complexity estimation of bit count resulting from application of particular QT without running actual entropy encoding or quantization.

Our RC method provides three modes of operation depending on the quality/performance trade-off preference of the user: fast, LS (local search) and DP (dynamic programming). All three modes utilize the same adaptive bit count predictor module which is trained to adapt to particular input image prior to running chosen RC mode.

The libjpeg-turbo library was chosen as a basic JPEG implementation due to its good SIMD-based performance optimization and modular source code structure making RC implementation easier.

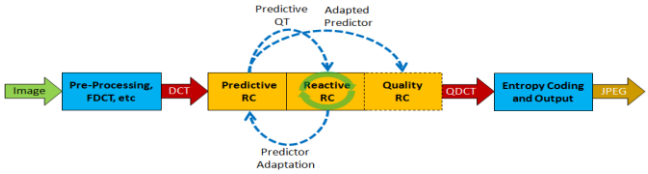

Figure 2. JPEG RC architecture overview

Our method operates using the following stages (see Figure 2):

1. Perform necessary JPEG pre-processing (e.g. RGB-to-YCbCr conversion (if required), sample offsetting, etc) implemented in libjpeg-turbo library.
2. Perform FDCT and gather DCT coefficients histograms.
3. Run bit count predictor adaptation loop by alternating predictive and reactive RC operations.
4. If the LS or DP RC mode is requested, run the respective RDO algorithm using the adaptive bit count predictor trained earlier, otherwise, if fast RC mode is configured, proceed to the next stage.
5. Quantize the DCT data using the QT output by the configured RC algorithm, then run entropy encoding on the quantized data and output encoded JPEG file.

These stages will be considered in more detail in the next sections.

Adaptive bit count predictor

The basic idea of the bit count predictor is the fast estimation of the output bit count resulting from application of the JPEG algorithm to the input image without running the entire JPEG pipeline, namely, without costly entropy coding stage (and even without quantization). The estimation is based on image statistics which could be easily gathered once per image without significant computational overhead, so that the estimation itself (which may be run many times during RC operation) has low complexity and would not restrict iterations count for sophisticated RC algorithms.

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

Some of the most computationally efficient image statistics which could be used for the task are coefficient-wise average entropy H_{avg} [18] and ρ-parameter (i.e. fraction of zero-valued quantized DCT coefficients) [19][20].

To minimize overhead of the most costly part of H_{avg} and ρ calculation, DCT analysis stage is usually run after FDCT stage [14]. DCT analysis consists of gathering DCT coefficients histograms, which are used later to easily calculate either H_{avg} or ρ with negligible computational overhead. We use the same approach as in [14] to gather histograms of DCT coefficients multiplied by 2 and rounded to integer values (which is effectively the transformation of real-valued DCT coefficients to fixed-point representation with q-scale of 1), separately for each color plane (Y, Cb and Cr). Hereinafter, $C_i(s)$ will be used to denote a histogram of i-th DCT sub-band values (i = 0,…,63). Since the same process is used for each color plane, we omit color plane index (unless explicitly given).

Having calculated DCT histograms once per input image, we can easily calculate both coefficient-wise average entropy H_{avg} and ρ-parameter for any QT $\{ Q_i \mid i = 0,\ldots,63 \}$ without running actual quantization:

$$H_{avg} = \frac{\sum_{i=0}^{63} H_i}{64}, \quad H_i = -\sum_{q=q_{\min}}^{q_{\max}} p_i(q)\log_2\left(p_i(q)\right) \qquad (1)$$

$$p_i(q) = \frac{1}{N_b} \sum_{\left\{s \,\middle|\, \left\lceil \frac{s}{Q_i} \right\rfloor = q\right\}} C_i(s)$$

$$\rho = \frac{\sum_{i=0}^{63} \rho_i}{64}, \quad \rho_i = \frac{\sum_{s=-\frac{Q_i-1}{2}}^{\frac{Q_i-1}{2}} C_i(s)}{N_b} \qquad (2)$$

where N_b is the total number of 8x8 blocks in the image and the full quantizer range is defined by $q_{min} = 1$ and $q_{max} = 255$.

Differently from RC algorithms in both [19] and [20], our implementation uses either H_{avg} or ρ in the sophisticated RC algorithm (Wu-Gersho or RD-OPT) rather than simple ρ-to-quantizer mapping, due to the fact that both [19] and [20] deal with different codecs (MPEG-2, h.263, MPEG-4, JPEG XR) and concentrate on determining per-block quantization parameter rather than quantizer-per-sub-band which is the case with JPEG.

Having calculated H_{avg} or ρ-parameter, the bit count predictor estimates the predictive bit count B_p using linear model:

$$B_p(P) = \alpha P + \beta \qquad (3)$$

where prediction parameter $P = H_{avg}$ or ρ.

The parameters of linear model may be trained off-line (using large representative database of images) or on-line (for the input image being encoded) by fitting linear function to the experimental distribution of bit counts (resulting from running entire JPEG pipeline for different QTs) versus prediction parameters (H_{avg} or ρ). Although usage of off-line trained parameters does not require any additional fitting during actual image compression, thereby providing good computational performance, the accuracy of resulting RC is poor, due to the lack of adaptation to particular input image, so we use the off-line trained parameters as initializers for on-line training to improve its convergence by starting from the values which are good "on average".

To make bit count predictor adaptive for particular input image, on-line training (adaptation) is used. Its main idea is to use values of prediction parameters and bit counts resulting from several JPEG encoding runs for the same input image in linear fitting. Scaled versions of the default JPEG QT (see Annex K in [1]) are used for gathering data to be fitted. The process of the on-line training is steered so as to achieve requested target bit rate by varying scaling factor (SF) of the default JPEG QT. To find the appropriate scaling factor, the false position root-finding method is used. Figure 3 shows the flowchart of the adaptation algorithm (low importance details, like scaling factor range checking, are omitted). The algorithm consists of the two interoperating parts:

1. Predictive part, which determines the appropriate scaling factor using bit count predictor. The predictor first estimates the prediction parameter (H_{avg} or ρ) using current QT (which is the default JPEG QT scaled using current scaling factor) and DCT histograms (gathered prior to the adaptation stage) and then uses the estimated prediction parameter value in Eq. (3).
2. Reactive part, which is responsible for updating (correcting) prediction model parameters using the actual bit count available after JPEG encoding (thereby providing a feedback to the predictive part). The updating is done using MSE method: α and β are modified so as to minimize mean square error (MSE) of the prediction at each iteration (N_{iter} is the current number of the reactive loop iterations and k is the iteration index):

$$MSE = \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left(B_p(P_k) - B_a^{(k)}\right)^2 = \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left(\alpha P_k + \beta - B_a^{(k)}\right)^2 \qquad (4)$$

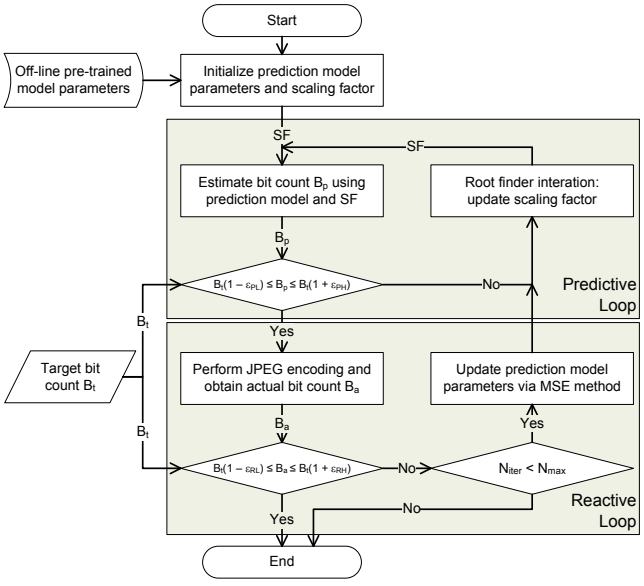

Figure 3. Flowchart of the bit count predictor adaptation process

The combination and interoperation of those two parts is crucial for attaining both acceptable complexity and good prediction accuracy: the predictive part stand-alone is fast, but

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

provides poor prediction accuracy, while stand-alone operation of the reactive part requires many costly encoding iterations to achieve good prediction accuracy resulting in high computational complexity.

To keep the computational complexity low, we limit the maximum number of entropy encoding iterations N_{max} to 4 (for the fast RC mode) or 8 (for the LS/DP RC modes). The bit count tolerances for the predictive part (ε_{PL} and ε_{PH}) and for the reactive part (ε_{RL} and ε_{RH}) are set to 0.01. The motivation behind such settings is that the fast RC mode is only aimed at achieving target bit rate while keeping complexity as low as possible, that's why it has low iterations number limit. In contrast, the LS and DP RC modes are aimed at both providing target bit rate and improving compressed image quality, thus their sophisticated RDO algorithms require much more accurate bit count predictor, which is achieved via performing more adaptation iterations. Another optimization trick we used is that the bit counting module was configured to process only fraction of MCUs of the input image rather than the entire image. For the fast RC mode the fraction is set to 50%, for the LS/DP RC modes it's set to 100% (since those modes are designed to perform quality-optimized encoding).

Fast RC mode

Fast RC mode is the fastest of all the modes and is aimed at reaching target file size only (without quality optimization). Its operation is identical to the on-line adaptation described in previous section. Since the on-line adaptation is steered by varying the scaling factor so as to reach the target bit count, the output of the process is the scaling factor (and respective scaled version of the default JPEG QT) providing the output file size close to the configured target value.

LS RC mode

LS RC mode is aimed at attaining the target bit rate while providing better reconstructed image quality than the fast RC mode. Quality improvement is achieved via construction of rate-distortion optimized QT based on the Wu-Gersho algorithm [10]. Unlike the fast RC mode, this RDO algorithm produces QT which is not a scaled version of the default JPEG QT. In contrast to the original algorithm described in [10], our implementation does not perform costly image encoding for each QT update, but uses the adaptive bit count predictor to estimate bit count resulting from the application of updated QT. The adaptive bit count predictor is trained for the input image by the fast RC mode operation which is run prior to running the RDO algorithm.

In our implementation of the Wu-Gersho algorithm we used sum-of-squared-errors (SSE) in DCT domain as a distortion measure. It may be easily calculated using the DCT histograms $\{ C_i(s) \mid i = 0,\ldots,63 \}$ via the following formula:

$$D = \sum_{i=0}^{63} \sum_{s=s_{\min}}^{s_{\max}} C_i(s) \left(s - \left[\frac{s}{Q_i} \right] Q_i \right)^2 \tag{5}$$

To speed up the RDO stage, sub-band-wise distortion and bit count tables are pre-calculated for relevant quantizer values. Pre-calculated sub-band-wise D(i, q) and B_p(i, q) are defined using the following formulae (i is a DCT sub-band index and q is the quantizer value):

$$D(i,q) = \sum_{s=s_{\min}}^{s_{\max}} C_i(s) \left(s - \left[\frac{s}{q} \right] q \right)^2$$
$$B_p(i,q) = \frac{\alpha P_i(q) + \beta}{64}, \quad P_i = H_i \quad or \quad \rho_i \tag{6}$$

Using those tables, the total distortion for the QT $\{ Q_i \mid i = 0,\ldots,63 \}$ can be calculated via $D = \sum_{i=0}^{63} D(i,Q_i)$ and the overall bit count prediction via $B_p = \sum_{i=0}^{63} B_p(i,Q_i)$ (see Eqs. (1), (2), (3) and (5)). So, the differences of distortion and bit count when Q_i is replaced by q are $\Delta D|_{Q_i \to q} = D(i,q) - D(i,Q_i)$ and $\Delta B|_{Q_i \to q} = B_p(i,q) - B_p(i,Q_i)$.

Another difference from the original Wu-Gersho design is the initial QT choice: we use the QT output by the preceding fast RC mode operation multiplied by 2, which significantly decreases the RDO algorithm iterations number, while almost not affecting the resulting compressed image quality.

Such a design makes our implementation of the Wu-Gersho algorithm dramatically faster than the original Wu-Gersho design (by orders of magnitude) though at the cost of deterioration of image quality, as compared to the original Wu-Gersho implementation, due to the fact that predicted bit count is used instead of the true one (i.e. the output of quantization and entropy encoding process), so the algorithm may proceed in wrong (non-optimal) direction and end up with non-optimal QT. However, the resulting quality is significantly better than the one provided by the fast RC mode.

DP RC mode

DP RC mode is again aimed at attaining the target bit rate while providing better reconstructed image quality than the fast RC mode. Quality improvement is achieved via construction of rate-distortion optimized QT based on the RD-OPT algorithm [14]. Like the LS RC mode, this RDO algorithm produces QT which is not a scaled version of the default JPEG QT. In contrast to the original algorithm described in [14] (and all derived works), our implementation does not use stand-alone coefficient-wise average entropy H_{avg} as a bit count estimator, but uses the adaptive bit count predictor (which could use either of H_{avg} or ρ as a prediction parameter). Despite that difference, the RD-OPT algorithm can be used with our bit count predictor, since the output of the predictor can be decomposed into a sum of contributions from individual coefficients:

$$B_p = \alpha P(Q_i) + \beta = \alpha \frac{\sum_{i=0}^{63} P_i(Q_i)}{64} + \beta = \sum_{i=0}^{63} B_p(i,Q_i) \tag{7}$$

where B_p(i, q) is defined by Eq. (6).

Just like the LS RC mode, pre-calculated sub-band-wise D(i, q) and B_p(i, q) are used to speed up calculations, sum-of-squared-errors (SSE) in DCT domain is used as a distortion measure and the adaptive bit count predictor is trained for the input image by the fast RC mode operation which is run prior to running the RD-OPT algorithm. For the bit count discretization we use rounding to

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

the nearest multiple of the bit count bin size $\frac{B_t}{N_B}$, where B_t is the target bit count and N_B = 1000 is the number of bit count bins (which corresponds to the MAXRATE parameter in [14]). We found that this value provides quite optimal trade-off between complexity, bit rate accuracy and image quality.

Such a design makes our implementation of the RD-OPT algorithm better than the original RD-OPT design in terms of bit count estimation accuracy, though at the cost of computational overhead due to the bit count predictor adaptation stage.

Color image processing

In general, each of the algorithms described above processes each color plane separately. However, since the target JPEG file size (set by the user) is specified for all color planes altogether, we need some means to distribute the target bit count among the color planes. For our RC method, we use a simple approach: during the on-line training (adaptation) stage the algorithm keeps track of the distribution of the total bit count (after each JPEG encoding) among color planes and after the adaptation stage is finished, uses the distribution corresponding to the iteration which provides the total output bit count most close to the target bit count. In that way, we use color planes bit distribution motivated by the relationship between the default JPEG QTs for luminance and chrominance applied to the particular input image. Since the default JPEG QTs are empirically derived based on psychovisual thresholding, our approach provides good quality of chrominance components while keeping their bit budget adequately low.

Since distortions in Cb and Cr chrominance channels are perceived approximately equally by HVS, both those planes are processed in a combined way: that is, a single set of DCT histograms is gathered for both planes, a single bit count prediction parameter is estimated for both planes, etc. and a single resulting QT is used to quantize DCT coefficients of both planes. Such an approach helps to decrease computational complexity (for color images, two effective planes are processed – luminance and combined chrominance – instead of the three) while inflicting almost no image quality degradation.

Experimental results

Performance metrics

In order to evaluate the performance of the developed RC method in terms of computational complexity, image quality and RC accuracy, we use three respective quantitative measures:

1. Relative complexity, defined as a ratio of the CPU clocks consumed by the RC-enabled encoder to that of the bare (non-RC-enabled) libjpeg-turbo encoder.
2. Peak Signal-to-Noise Ratio (PSNR) metrics, which evaluates quality degradation of the reconstructed image as compared to the original input image. To aggregate luminance PSNR (Y-PSNR) measurements for large set of images and bit rates, BD-Rate (Bjøntegaard Delta Rate) [21] is used.
3. RC accuracy is evaluated by the ratio of the actual JPEG file size (output by the RC-enabled encoder) and the configured target file size.

Test conditions

The developed RC-enabled JPEG encoder was extensively tested on a quite large and representative image database, containing 171 images from several test image databases, popular in the image compression community: EPFL JPEG XR Image Compression Database, JPEG-LS Test Suite, Kodak Lossless True Color Image Suite and USC-SIPI Image Database. The target bit rate was varied in a wide range: from 0.25 bits per pixel (bpp) to 4.0 bpp, with step size of 0.25 bpp.

For all RC configurations Huffman table optimization was enabled (since disabling it resulted in deterioration of both quality and complexity), while the anchor (baseline) for BD-Rate measurements was chosen to be the fast RC mode without Huffman table optimization using H_{avg} as the prediction parameter.

The test system had Intel Core i7-3770 CPU clocked at 3.4GHz, 16 GB of RAM and Windows 7 Enterprise x64 SP1.

Summary of results

From the results given in Table 1, one can see that our method provides quite wide range of trade-offs between complexity and quality (negative BD-Rate values mean bit rate reduction at the same quality, which is equivalent to having better quality at the same bit rate), while consistently keeping good RC accuracy (average ≈ 100% and low RMS ranging from 3 to 12%.). Complexity versus quality trade-offs are ranging from 6.5% BD-Rate reduction at the average relative complexity of 3.2 to 20.2% BD-Rate reduction at the average relative complexity of 28.1.

The fast RC mode offers the lowest complexity, while providing the smallest quality improvement, which is completely due to the usage of Huffman tables optimization, since the fast RC mode does not involve any RDO operation. The DP RC mode provides the highest quality improvement, while being the highest complexity mode. At the same time, the LS RC mode, while providing somewhat worse image quality improvement as compared to DP RC mode, offers much lower complexity.

Note that prediction parameter choice may also be used for quality/accuracy/complexity trade-off, though its effect varies depending on the RC mode. While the complexity is always significantly lower for ρ-parameter, the quality improvement for RDO-based RC modes is higher when using H_{avg}. As to RC accuracy, ρ-parameter provides better accuracy than H_{avg} for the LS RC mode, unlike the DP RC mode, where the best accuracy is achieved with H_{avg}.

Table 1. Summary of the results for the RC-enabled encoder

RC mode	Prediction parameter choice	Test results		
		Accuracy, %	Average relative complexity	Avg. BD-Rate, %
		Avg. ± RMS		
Fast	H_{avg}	99.3 ± 3.0	3.2	-6.5
	ρ	99.3 ± 3.0	2.3	-6.5
LS	H_{avg}	99.2 ± 12.3	14.5	-19.4
	ρ	99.4 ± 6.4	8.8	-15.4
DP	H_{avg}	97.3 ± 3.2	28.1	-20.2
	ρ	98.4 ± 7.2	16.3	-15.4

To compare our results with the ones of the original RD-OPT design (as described in [14]), we configured our implementation of

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

the DP RC mode so that it uses H_{avg} alone instead of the adaptive bit count predictor (just by setting $\alpha = 1$ and $\beta = 0$ and switching off the adaptation stage) and ran the same test with the same image database using different values of N_B parameter. Since [14] deals with grayscale images only, we ran both the RD-OPT and our DP RC mode on grayscale images. The results of the comparison between the RD-OPT and the DP RC mode are presented in Table 2. One can see that while for $N_B = 1000$ the original RD-OPT design is faster than our DP RC mode and provides better quality, it's worse in terms of RC accuracy (due to the lack of adaptation of the bit count predictor to the input image). Note that increasing N_B value does not help to improve quality or RC accuracy, while the complexity rises dramatically.

Note that though we use different approach to setting variable parameters of RD-OPT algorithm – we vary N_B parameter (which is equivalent to varying the MAXRATE parameter in [14]) rather than BPPSCALE as is done in [14] – it's still easy to synchronize our implementation of the RD-OPT algorithm with the original design, since MAXRATE (and thus N_B) and BPPSCALE can be related by the simple formula:

$$R_t \cdot BPPSCALE = MAXRATE \tag{8}$$

where R_t is the target bit rate value (in bpp) for the image being compressed. So, for example, for $R_t = 1.0$ bpp the values of BPPSCALE and MAXRATE are equal and the results can be directly compared.

Table 2. Results of the RD-OPT algorithm test

N_B		Test results		
		Accuracy, % Avg. ± RMS	Average relative complexity	Avg. BD-Rate, %
1000		99.7 ± 9.5	27.9	-15.6
2500		99.9 ± 9.6	46.4	-15.8
5000		100.1 ± 10.0	79.2	-15.8
7500		100.4 ± 10.3	110.0	-15.8
10000		101.0 ± 11.0	135.5	-15.9
Our DP RC mode	H_{avg}	98.9 ± 5.7	36.5	-7.8
	ρ	100.2 ± 9.8	28.4	-2.3

Computational complexity details

Figure 4 shows the split-up of RC-enabled JPEG encoder computational complexity in a form of pie charts of the constituent JPEG pipeline modules contributions. It can be easily seen that the combined share of the RC module along with the bit count predictor adaptation, being as small as 28% for the fast RC mode without Huffman table optimization, expectably grows when this optimization is used and becomes the dominant part of the total complexity for the RDO-based RC modes, taking up almost 80% for the DP RC mode with Huffman table optimization.

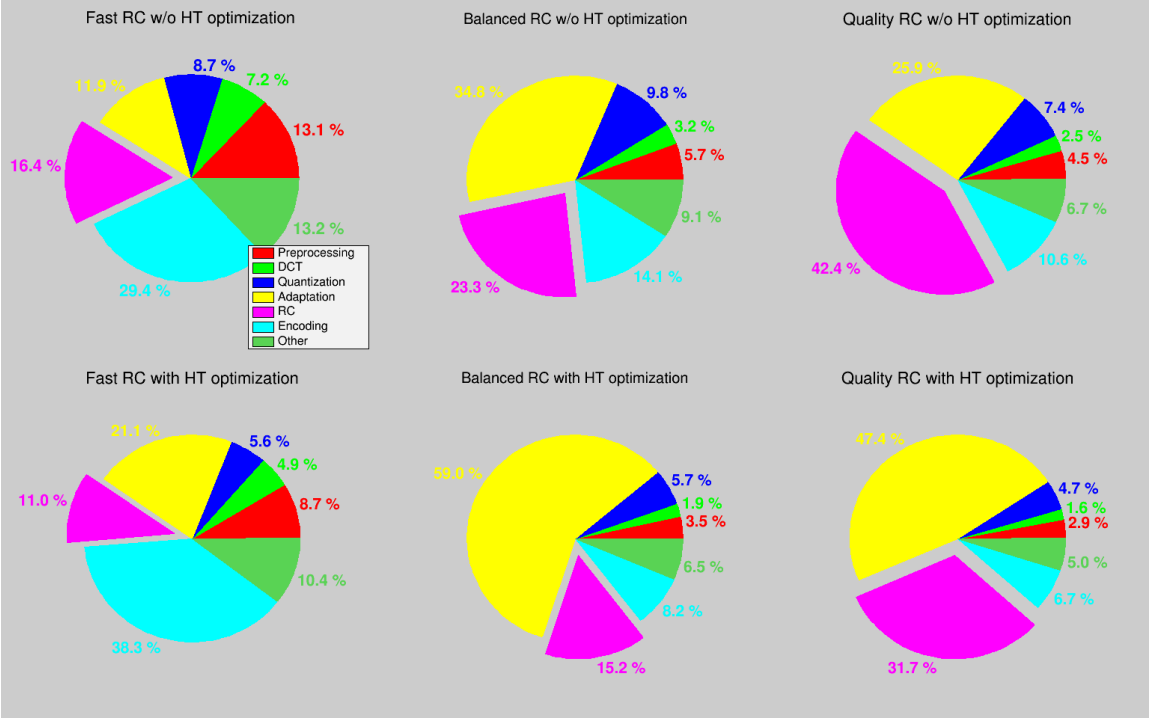

Figure 4. Module performance pie charts of the RC-enabled encoder ("Balanced RC" is the LS RC mode, "Quality RC" is the DP RC mode)

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

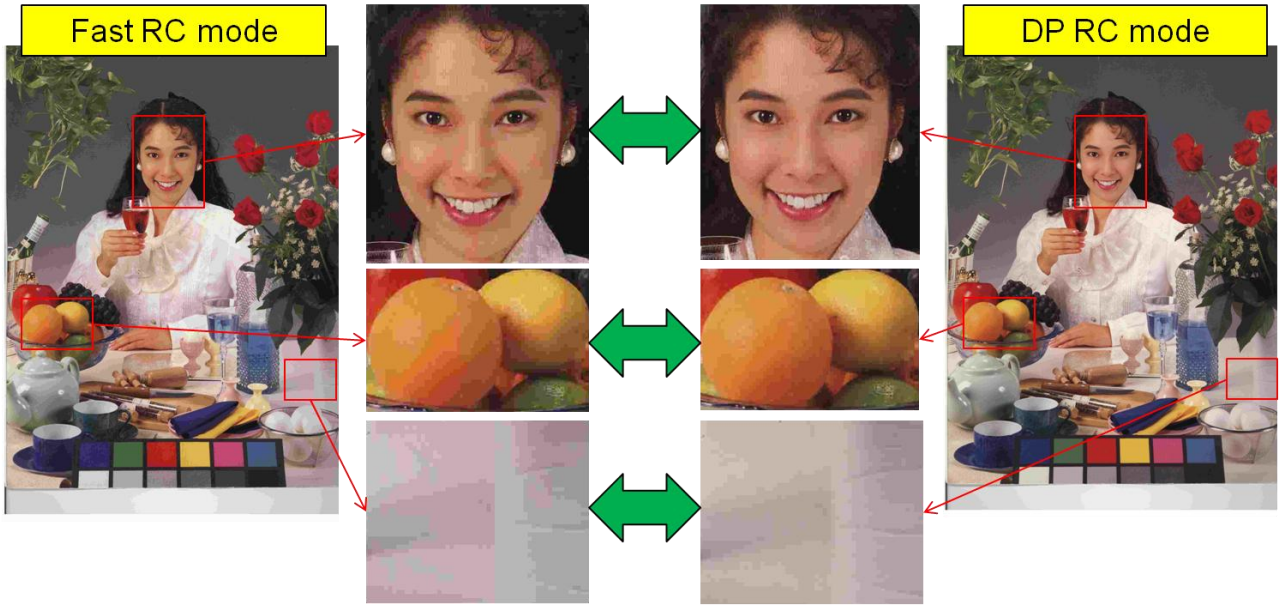

Figure 5. Visual quality comparison of the outputs of the DP RC mode and the fast RC mode (for the same JPEG file size)

Image quality details

Figure 5 shows the visual quality comparison for example image, compressed with the fast and the DP RC modes into the JPEG files of almost the same size. The highlighted regions show that RDO-based DP RC mode provides visibly smoother gradients and more natural colors.

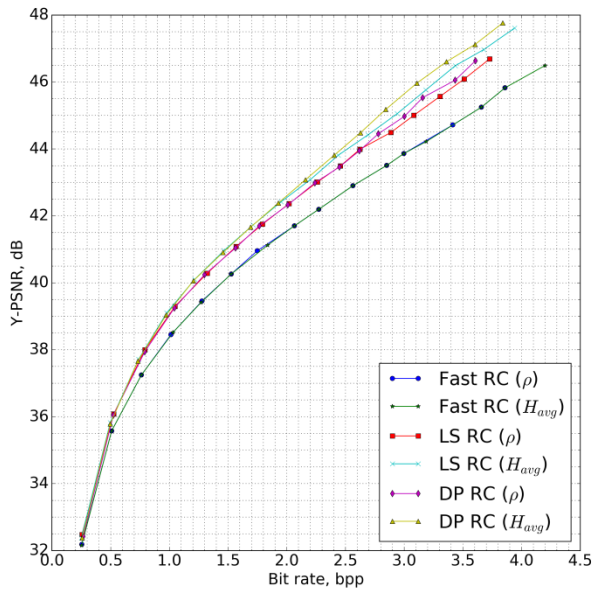

Figure 6. Y-PSNR results for the Lena color image encoded by the RC-enabled encoder

Figure 6 shows rate-distortion (RD) curves for the Y-PSNR results for all of the RC configurations of our method computed for the Lena color image (as a typical example of image used for JPEG compression). The RD curves directly reflect the results of Table 1: the fast RC mode has the worst quality (with the results of the variants using ρ-parameter and H_{avg} being almost identical), the DP RC mode provides the best quality, while the LS RC mode is in-between the fast and the DP RC modes and for both RDO-based modes the variant using H_{avg} is better than the one using ρ-parameter. The difference in Y-PSNR between different modes is small for small bit rates (i.e. bit rates below 0.5 bpp) and gradually increases with increasing bit rate up to 2 dB at 4 bpp between the DP RC mode with H_{avg} and the fast RC mode. Small quality improvement for low bit rates is quite an expectable behavior, since in both RDO-based algorithms we use the bit count predictor (instead of accurate bit counting via entropy encoding), which has quite poor prediction accuracy at low bit rates, thereby adversely affecting RDO operation and leading the process of generation of optimal QT to non-optimal solution.

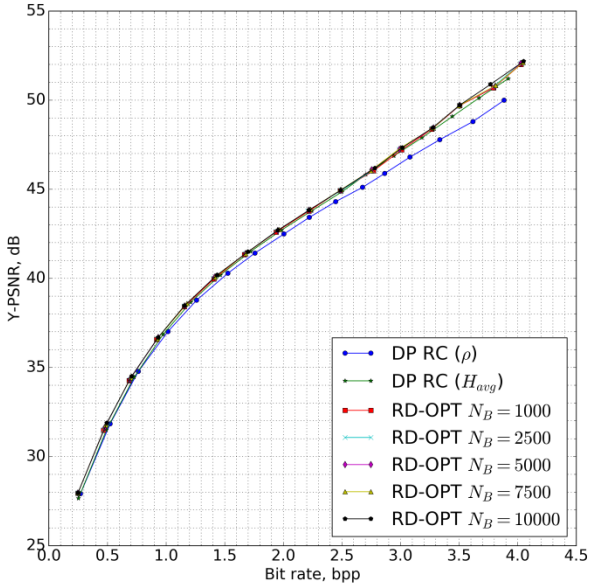

Figure 7. Y-PSNR results for the Barbara grayscale image encoded by our DP RC mode and the original RD-OPT algorithm

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

For comparison with the original RD-OPT design, Figure 7 shows the RD curves computed for the Barbara grayscale image for both the original RD-OPT algorithm (using different values of N_B parameter) and the DP RC mode of our method. One can see that the quality improvement of the original RD-OPT algorithm over our DP RC mode (using H_{avg}) is quite small and even increasing bit count discretization accuracy (i.e. N_B parameter) by an order of magnitude leads to improvement in Y-PSNR by less than 0.4 dB.

Conclusion

In the paper, we propose a novel JPEG rate control method. Our method combines strong points of several previously known approaches: fast bit count estimation using coefficient-wise average entropy and ρ-parameter, local search via Wu-Gersho algorithm, RD-OPT algorithm based on dynamic programming. We train our adaptive bit count predictor on the input image using linear prediction model and one of the prediction parameters (coefficient-wise average entropy or ρ-parameter) and subsequently use it in one of the three implemented RC modes: fast, LS or DP, the LS mode being based on the Wu-Gersho algorithm and the DP mode being based on the RD-OPT algorithm. Our implementations of both the Wu-Gersho and the RD-OPT algorithms improve the original designs either by making them significantly faster (in case of Wu-Gersho algorithm) or by making them more accurate (in case of RD-OPT algorithm). The resulting multi-mode RC method has quite good RC accuracy ranging from 3 to 12% and provides a wide range of trade-offs between complexity, accuracy and quality – from 6.5% BD-Rate reduction at the average relative complexity of 3.2 to 20.2% BD-Rate reduction at the average relative complexity of 28.1 – suitable for vast variety of applications, e.g. JPEG compression of images in digital cameras or Motion JPEG compression of video streams for transmission over a network channel having limited bandwidth capacity.

References

[1] ISO/IEC JTC1 10918-1, Information technology – digital compression and coding of continuous-tone still images: Requirements and guidelines, ITU-T Rec. T.81, 1992.

[2] L.F. Costa and A.C.P. Veiga, "Identification of the best quantization table using genetic algorithms," in IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp. 570-573, 2005.

[3] Y. Jiang and M.S. Pattichis, "JPEG image compression using quantization table optimization based on perceptual image quality assessment," in Conference Record of the 45th Asilomar Conference on Signals, Systems and Computers (ASILOMAR), pp. 225-229, 2011.

[4] B.G. Sherlock, A. Nagpal, D.M. Monro, "A model for JPEG quantization," in Proceedings of International Symposium on Speech, Image Processing and Neural Networks (ISSIPNN'94), vol. 1, pp. 176-179, 1994.

[5] M.G. Logutenko, "JPEG codec with adjustable bitrate," in Seventh International Forum on Strategic Technology (IFOST), pp. 1-4, 2012.

[6] R. de Queiroz and R. Eschbach, "A method for rate control and compression estimation in JPEG," 2001, http://image.unb.br/queiroz/papers/compression_estimation.pdf.

[7] A. Bruna, S. Smith, F. Vella, F. Naccari, "JPEG rate control algorithm for multimedia," in IEEE International Symposium on Consumer Electronics, pp. 114-117, 2004.

[8] L.-W. Chang, C.-Y. Wang, S.-M. Lee, "Designing JPEG quantization tables based on human visual system," in Proceedings of International Conference on Image Processing (ICIP 99), vol. 2, pp. 376-380, 1999.

[9] D.G.W. Onnasch, G.P.M. Prause, A. Ploger, "Quantization table design for JPEG compression of angiocardiographic images," in Computers in Cardiology, pp. 265-268, 1994.

[10] S.-W. Wu and A. Gersho, "Rate-constrained picture-adaptive quantization for JPEG baseline coders," in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-93), vol. 5, pp. 389-392, 1993.

[11] H.T. Fung and K.J. Parker, "Design of image-adaptive quantization tables for JPEG," Journal of Electronic Imaging, vol. 4(2), pp. 144-150, 1995.

[12] W.C. Fong, S.C. Chan, K.L. Ho, "Designing JPEG quantization matrix using rate-distortion approach and human visual system model," in IEEE International Conference on Communications (ICC'97), vol. 3, pp. 1659-1663, 1997.

[13] E.-H. Yang and L. Wang, "Joint optimization of run-length coding, Huffman coding, and quantization table with complete baseline JPEG decoder compatibility," IEEE Transactions on Image Processing, vol. 18, pp. 63-74, 2009.

[14] V. Ratnakar and M. Livny, "RD-OPT: an efficient algorithm for optimizing DCT quantization tables," in Proceedings of Data Compression Conference (DCC '95), pp. 332-341, 1995.

[15] V. Ratnakar and M. Livny, "An efficient algorithm for optimizing DCT quantization," IEEE Transactions on Image Processing, vol. 9(2), pp. 267-270, 2000.

[16] K. Ramchandran and M. Vetterli, "Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," IEEE Transactions on Image Processing, vol. 3(5), pp. 700-704, 1994.

[17] K. Ramchandran and M. Crouse, "JPEG optimization using an entropy-constrained quantization framework," in Proceedings of Data Compression Conference (DCC'95), pp. 342-351, 1995.

[18] V. Ratnakar, E. Feig, E. Viscito, S. Kalluri, "Runlength encoding of quantized discrete cosine transform (DCT) coefficients," in Proc. SPIE 2419, Digital Video Compression: Algorithms and Technologies, p. 398, 1995.

[19] Z. He and S.K. Mitra, "A linear source model and a unified rate control algorithm for DCT video coding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 12(11), pp. 970-982, 2002.

[20] D. Chan, J. Liang, and C. Tu, "ρ-domain rate control for JPEG XR," in Proc. 2010 IEEE Asilomar Conference on Signals, Systems, and Computers, pp. 226-230, 2010.

[21] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves (VCEG-M33)," in VCEG Meeting (ITU-T SG16 Q.6), 2001.

Author Biography

Valery Anisimovskiy received his BS (1998) and MS (2000) in applied mathematics and physics from the Moscow Institute of Physics and Technology (MIPT). Since then he worked at the Institute for Nuclear

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-255

Research of RAS (2000-2004), SPIRIT Corp (2004-2009), Luxoft (2009-2011), Huawei Russian Research Center (2011-2014). Since 2014 he is working at Samsung R&D Institute Russia, Moscow. His research interests include fields of machine learning, pattern recognition, computer vision, video coding and audio coding.

Sergey Zavalishin received his MS degree in Computer Science from Moscow Engineering Physics Institute/University (MEPhI), Russia in 2012. Currently he is a post graduate student of Ryazan State Radio Electronics University. His research interests include machine learning, computer vision and image processing.

Ilya Kurilin received his MS degree in radio engineering at Novosibirsk State Technical University (NSTU), Russia in 1999 and his PhD degree in theoretical bases of informatics at NSTU in 2006. In 2007 Dr. I. Kurilin joined Image Processing Group, Algorithm Lab at Samsung R&D Institute Russia, where he is engaged in photo/document image enhancement and image understanding projects.