Optimize x265 Rate Control: An Exploration of Lookahead in Frame Bit Allocation and Slice Type Decision

Zhenyu Liu, *Member, IEEE*, Libo Wang, Xiaobo Li, and Xiangyang Ji, *Member, IEEE*

***Abstract*— To improve the rate-distortion (R-D) quality, x265 rate-control makes a variety of vital decisions—such as scene cut detection, slice type decision, and coding-unit quantization parameter (QP) offsets—leveraging on lookahead to evaluate the information propagation through the current and the near future consecutive frames. However, as the frame base QP that dominates the bit amount allocated to one frame was only determined by the long-term complexity history in the original algorithm, the frame bit allocation became insensitive to the recent scene changes with the growth of coding time. In addition, the specified threshold in slice type decision, which was compared to the estimated frame coding costs to detect the B-type slice, did not consider the impacts of quantization. As mentioned earlier, the irrational elements degraded the rate accuracy and the R-D performance of x265. In this paper, the frame base QP is determined with not only the coding complexity history but also the complexity changes and the data dependencies between the current and the near future pictures by exploring lookahead. Moreover, the quantization scale is introduced to the threshold specification in slice type decision, which identifies more pictures as B-type properly when increasing QP. The proposed algorithms were conducted in x265 version 2.4. Experiments revealed that under the default preset (–preset medium), 0.617 dB on average and up to 1.705 dB Bjøntegaard-Delta quality gains were achieved, while saving the encoding time by 1.13% and improving the rate accuracy by 4.2% on average.**

***Index Terms*— HEVC, x265, rate control, CU-tree, MB-tree, slice type decision.**

I. Introduction

AS THE essential technique in video coding, rate-control plays the vital role in the band-limited practical communication systems, which are subject to the discrepancy between the time-varying picture complexity and the constant band limitation. If we apply an constant quantization scale (Q), the required rate bandwidth might overflow or underflow the limitation seriously. The function of rate-control is to find the Q for each coding unit (CU) that minimizes the Rate-Distortion (R-D) cost under the specified rate constrains [1]. The rate-control algorithm can be formulated as

$$\min_{Q_i} \sum_i (D_i + \lambda R_i) \quad \text{s.t.} \sum_i R_i = Rc, \tag{1}$$

where, Q_i stands for the quantization scale of the ith CU; D_i and R_i are the corresponding distortion and rate, respectively; Rc represents the rate constraint. Although rate-control is not developed as a normative tool in any video coding standard, it has prominent impacts to the coding quality in the specified bandwidth limitation scenario. Every video coding standard reference software [2]–[6] recommends its own rate-control algorithm, and all commodity hardwired and software encoders [7]–[11] strive for promoting the performance of rate-control module.

The traditional rate-control algorithms in reference software endeavored to improve the accuracy of rate model parameterized by Q. In specific, H.263 developed its rate model on the assumption of Gaussian distributed prediction residues and L2-norm distortion [3], [12], [13]. The quadratic models of MPEG-4 [14] and H.264/AVC [4] both stemmed from the Laplacian distributed residues and L1-norm distortion [15]. Ding, et.al., noticed that the rate estimation deviated from the theoretical model at low bit rates, therefore, they proposed the rule of thumb power function model [16]. Mallat and Falzon [17] derived the theoretical power function model for low bit rates, i.e., $D(R) = CR^{1-2\gamma}$ (where $\gamma \approx 1$ and $C > 0$), which is more accurate at low rates ($R <$ 1bit/pixel) than the previous high-resolution quantizations based model, i.e., $D(R) = C2^{-2R}$(where $C > 0$). The ρ-domain model [18] explored the linear relation between the percentage of zeros in quantized transform coefficients (that is denominated as ρ) and the rate R. Specifically, $R = \theta(1-\rho)$, where θ is a constant.

The reference software of the state-of-the-art video coding standards, High Efficient Video Coding (HEVC) [19], [20], explored the power function rate model [17] to develop the relationship between the target rate and the associated QP [5], [6], [21], which was denominated as λ-domain model. 0.56% rate accuracy and 1.08dB R-D performance improvements were achieved by λ-domain model as compared to the Q-domain counterpart [4] adopted by HM-8.0 [6]. Gao, et.al., improved the λ-domain method by building multiple CU-level

Manuscript received April 13, 2018; revised August 23, 2018, October 26, 2018 and December 11, 2018; accepted December 12, 2018. Date of publication December 17, 2018; date of current version March 6, 2019. This work was supported in part by the National Natural Science Foundation of China under Grants 61620106005 and in part by the Alibaba Group through the Alibaba Research Fellowship Program. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Zhengguo Li. *(Corresponding author: Xiangyang Ji.)*

Z. Liu is with RIIT, TNList, Tsinghua University, Beijing 100084, China (e-mail: liuzhenyu73@tsinghua.edu.cn).

L. Wang and X. Li are with the Alibaba Group Taobao Co., Ltd., Hangzhou 311121, China.

X. Ji is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: xyji@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TIP.2018.2887200

models. The model selection of a CU lies on the support-vector-machine and Nash-bargaining-solution [22].

The aforementioned rate-control algorithms neglected the impact of data dependency in motion estimation. In specific, improving the quality of one picture block would reduce the distortions of all predicted blocks, which applied this block as the reference. In practice, the optimal bit allocation in a spatial-temporal dependent coding environment is more efficient to coding quality improvement [23]–[27]. The early algorithms exploited the data dependency in frame-level, and applied the Viterbi or other fast search algorithms to find the optimal Q configurations [23]–[26]. Gao, et.al., developed the skip-mode CU percentage based quality dependency factor, which evaluated the degree of quality dependency between the reference frame and the predicted ones, to optimize the bit allocation for key frames [27]. The CU-level data dependency analysis scheme came from the MacroBlock-tree (MB-tree) technique [28] in H.264/AVC [29] software encoder x264 [7]. In literature [28], the MB-level information propagation in consecutive pictures and the overall distortions when altering the Q of one MB were analyzed by lookahead module, and then the MB-level quantization parameter (QP) offsets were determined. This technique was inherited by x265 [8], a software encoder for the up-to-date standards HEVC [20], and was denominated as coding-unit-tree (CU-tree). The R-D efficiency comparisons between x265 version 2.4 and HM16.17 under the configurations of constant quantization and rate-control are shown in Fig.1. Under the constant quantization, HM16.17 outperformed x265 because HM16.17 applied the more efficient hierarchical prediction structure with 5 dyadic hierarchy stages and the bi-prediction Inter-coded key pictures [30], [31]. In contrast, with the rate-control enabled, as the data dependencies were leveraged by CU-tree based rate-control of x265, its coding efficiency was superior to that of HM16.17. However, Fig.1 and the tests in Section IV demonstrated that the rate accuracy of x265 was inferior to HM16.17.

As compared to the next generation standards being brought up, such as AV1 and JEM, HEVC provides the good coding efficiency at the relatively low computational complexity. For example, JEM achieved an averaged 30% rate saving as compared to HM at the cost of 10.5x encoding time [32]. Consequently, x265 is still playing an important role in the industrial community as a high performance software encoder. In the original x265, each frame has its dedicated frame-level base QP. The CU-tree is adopted to estimate the CU QP offsets by analyzing the information propagation in successive frames. For one CU, its coding QP value is the sum of the frame-level base QP and its specified QP offset. In the original algorithm, the frame-level base QP value only depends on the complexity sum of the previously encoded frames, which cannot respond to the current scene change sensitively. To overcome this hindrance, we apply the lookahead module to estimate the full resolution coding overheads and the degree of information dependency in the future pictures, and then improve the performance of frame-level base QP definition. Additionally, we refine the performance of slice type decision by considering the impacts of quantization in lookahead to detect the B-type slice properly. Experiments revealed that an averaged 0.617dB quality gain was obtained by our proposals.

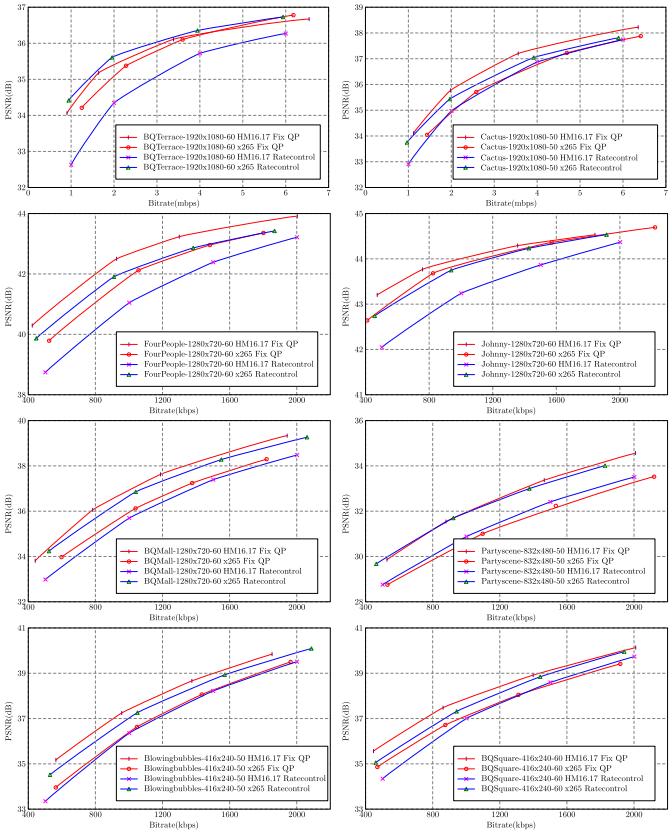


Fig. 1. Coding quality comparisons between x265 version 2.4 and HM16.17 under the conditions of constant quantization and rate-control enabled (The tests of x265 applied the configurations of "–preset placebo". The tests of HM16.17 were carried out under the configurations of "encoder_randomaccess_main.cfg". The target rates for HD1080p sequences include 1mb/s, 2mb/s, 4mb/s and 6mb/s. The target rates for other sequences are composed of 500kbps, 1mb/s, 1.5mb/s and 2mb/s).

The rest of this paper is organized as follows: Section II introduces the background of CU-tree technique and the associated terminology, and presents the proposed frame-level base QP derivation methods. Section III describes the optimized slice type decision algorithm. The experiments are illustrated in Section IV. Conclusions are drawn in Section V.

II. Frame-Level Base QP Derivation

x265 adopted CU-tree technique to derive the CU-level QP offsets. Section II-A explains the principle and describes the lookahead procedure adopted by the CU-tree information propagation analysis. The terminologies required in the following sections are introduced in Section II-A as well. Thereafter, the analytical frame-level base QP algorithms and the improved I-frame bits amortization leveraging on lookahead are proposed in Sections II-B, II-C and II-D, respectively.

A. x265 CU-Tree Introduction

The essential of CU-tree is to evaluate the amount of information of each basic picture block contributing to the

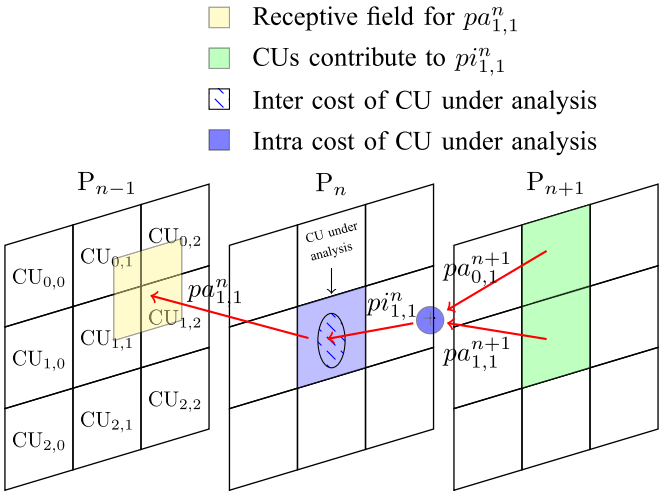


Fig. 2. Information propagation in half resolution prediction procedure (P_{n-1}, P_n and P_{n+1} are half sampled pictures of the original source ones. The circle filled with slash pattern in $CU_{1,1}$ of P_n represents the new generated information $c^n_{1,1}(1,0)$; Other grey blue area is the inherited information $c^n_{1,1}(0,0) - c^n_{1,1}(1,0)$).

predictions in future pictures. x265 and x264 both applied the lookahead scheme to implement three functions, i.e., deciding the slice types of pictures to be coded, analyzing the motion vectors of all basic picture blocks, and achieving the information propagation amounts with the basic picture block grain. The lookahead module collects a number of future pictures, which is 20 in default, and then, carries out half-resolution motion-estimation (HRME) on these sub-sampled source pictures to achieve the SATD costs without quantization, reconstruction, or entropy coding operations. For the nth half-resolution picture, with the defined slice type, the lookahead module calculates the Intra and Inter costs of all 8×8-CUs, which are expressed as $c^n_{x,y}(d_0,d_1)$, where x and y are CU indices, d_0 is the difference between the picture order count (POC) of the current CU and its list0 reference POC, and d_1 is the difference between the current POC and the POC of list1 reference. When d_0 and d_1 are both nonzero, $c^n_{x,y}(d_0,d_1)$ is a bi-prediction cost; If $d_0 > 0$ and $d_1 = 0$, $c^n_{x,y}(d_0,d_1)$ denotes the uni-prediction cost; Finally, if both d_0 and d_1 are equal to 0, $c^n_{x,y}(0,0)$ denominates the Intra prediction cost. In the following description, the uppercase character C denotes full-resolution frame-level prediction complexity, and $\dot{C}$ represents the corresponding half-resolution frame-level prediction cost. In Section II-D, the symbol $C^n(d_0,d_1)$ represents the P- or B-prediction cost of the nth frame, of which the temporal distances of list0 and list1 are d_0 and d_1, respectively.

Let us investigate the information propagation incurred by motion prediction. As illustrated in Fig. 2, it is assumed that there are three P-frames, i.e., P_{n-1}, P_n and P_{n+1}. The CU under analysis is $CU_{1,1}$ in P_n. The Inter motion prediction constructs the information propagation paths. For example, if $CU_{0,1}$ and $CU_{1,1}$ in P_{n+1} both applied $CU_{1,1}$ in P_n as the reference, these two blocks inherited some pieces of information from $CU_{1,1}$ in P_n. Obviously, besides the inherited information, one block also has its own generated information. The fundamental issue is to define the inherited information and the generated information quantitatively. In literatures [7], [8], [29], the total information contained in a CU is defined as the Intra prediction cost, i.e., $c^n_{x,y}(0,0)$. The Inter prediction cost is the difference between the current CU and its reference signals, which is attributed as the newly generated information. As Fig. 2 applies the uni-direction P-prediction, the Inter cost of CU with index (x,y) in the n-th frame is expressed as $c^n_{x,y}(1,0)$. The difference between Intra cost and Inter cost is the amount of information inherited from the references. In Fig. 2, $c^n_{x,y}(0,0) - c^n_{x,y}(1,0)$ is the piece of inherited information of $CU_{x,y}$ in P_n. When $CU_{x,y}$ in P_n is used as the reference of P_{n+1}, the propagate-in information $pi^n_{x,y}$ denotes the weighted sum of propagate-amount information (pa) of the future dependents. For instance, the $pi^n_{1,1}$ is a weighted sum of $pa^{n+1}_{0,1}$ and $pa^{n+1}_{1,1}$. The overall propagate-in information and propagate-amount information of P_n are denominated as PI^n and PA^n, respectively.

One key issue is how to derive the value of $pa^n_{x,y}$, given $pi^n_{x,y}$, $c^n_{x,y}(d_0,d_1)$ and $c^n_{x,y}(0,0)$. The propagate-amount information $pa^n_{x,y}$ should include the inherited information, i.e., $c^n_{x,y}(0,0) - c^n_{x,y}(d_0,d_1)$, and a fraction of $pi^n_{x,y}$, which is formulated as

$$pa^n_{x,y} = c^n_{x,y}(0,0) - c^n_{x,y}(d_0,d_1) + \left(1 - \frac{c^n_{x,y}(d_0,d_1)}{c^n_{x,y}(0,0)}\right) pi^n_{x,y}. \tag{2}$$

In (2), the term

$$\frac{c^n_{x,y}(d_0,d_1)}{c^n_{x,y}(0,0)} pi^n_{x,y}$$

represents the portion of $pi^n_{x,y}$ depending on the newly generated information, which does not contribute to $pa^n_{x,y}$. Another crucial issue is to deduce the variable pi with the associated pa values available. For example, as shown in Fig.2, when $pa^n_{1,1}$ is known, how to define the associated reference CUs in P_{n-1} and how much information of $pa^n_{1,1}$ contributes to the pi values of the referred CUs. It is defined that all blocks in the half resolution pictures have the uniform size $s \times s$ ($s = 8$ in x265 and x264). According to the motion vector, we can find the reference area (or the receptive field), marked as the yellow block in P_{n-1}. Because the reference pixels on P_{n-1} cover the joint areas of four adjacent 8×8-blocks, $pa^n_{1,1}$ contributes to the propagate-in information of $CU_{0,1}$, $CU_{0,2}$, $CU_{1,1}$ and $CU_{1,2}$ in P_{n-1}. In specific, if the overlap area of the receptive field and the associated CU ($CU_{x',y'}$) is denoted as $S_{(x,y)\to(x'y')}$, the portion of pa, i.e.,

$$\frac{S_{(x,y)\to(x'y')}}{s^2} pa,$$

is transferred to the pi of $CU_{x',y'}$ in the reference picture. For example, assuming the joint area of the yellow block and $CU_{0,1}$ in P_{n-1} is 3×3-pixels, $\frac{9}{64} pa^n_{1,1}$ is dispatched to $pi^{n-1}_{0,1}$. If the current CU is B-type, its pa is split equally between the two reference directions.

One prominent argument of CU-tree is that, the quantization of the current block not only degraded its own fidelity, but also produced the additive distortions to all predicted blocks

using the current block as the reference (Simply considering the stationary background blocks coded with *skip* mode). If the Q perturbation of the current CU is ΔQ, the overall distortion variations (ΔD) are composed of the change of the current CU distortions and that of $pi^n_{x,y}$ distortions. For the current CU, as the quantization is merely carried out on the prediction residues, the distortions of the generated information, i.e., $c^n_{x,y}(d_0, d_1)$, are affected. On the other hand, the inherited information, i.e., $c^n_{x,y}(0,0) - c^n_{x,y}(d_0, d_1)$, is copied from the references, therefore it is independent with ΔQ. As $pi^n_{x,y}$ is the total information copied from the current CU, its distortion is also modified by ΔQ. However, only a fraction of $pi^n_{x,y}$, i.e.,

$$\frac{c^n_{x,y}(d_0, d_1)}{c^n_{x,y}(0,0)} pi^n_{x,y},$$

is affected by ΔQ; In contrast, other portion of $pi^n_{x,y}$, i.e.,

$$\left(1 - \frac{c^n_{x,y}(d_0, d_1)}{c^n_{x,y}(0,0)}\right) pi^n_{x,y},$$

which depends on the inherited information, is not affected by ΔQ. In summary, the total information related with ΔQ is formulated as

$$c_\Delta = c^n_{x,y}(d_0, d_1) + \frac{c^n_{x,y}(d_0, d_1)}{c^n_{x,y}(0,0)} pi^n_{x,y}. \tag{3}$$

It is reasonable to assume that the overall distortion amount is proportional to c_Δ. From (3), as compared with the distortion of not referred CU ($pi^n_{x,y} = 0$), ΔD of the referred CU is scaled up by the factor r_Δ, that is written as follows.

$$\begin{aligned} r_\Delta &= \frac{c^n_{x,y}(d_0, d_1) + \frac{c^n_{x,y}(d_0,d_1)}{c^n_{x,y}(0,0)} pi^n_{x,y}}{c^n_{x,y}(d_0, d_1)} \\ &= 1 + \frac{pi^n_{x,y}}{c^n_{x,y}(0,0)} \end{aligned} \tag{4}$$

The second term of the right hand in (4) indicates the effect of the propagate-in signal $pi^n_{x,y}$. When $pi^n_{x,y}$ approaching 0, r_Δ regresses back to 1 properly.

According to the analysis of literatures [1], [23], the Lagrangian multipliers of all coding units in a stream with the optimal bit allocation, which are formulated as

$$\lambda = -\frac{\Delta D}{\Delta R} = -\frac{\Delta D/\Delta Q}{\Delta R/\Delta Q},$$

should have the same value. In other words, every bit should be assigned the same contribution to the distortion costs. In CU-tree technique, it is assumed that the term $\Delta R/\Delta Q$ is not affected by the propagate-in signal. ΔQ of the current CU will not affect the rates of its predicted CUs as well. In consequence, λ is linear with ΔD.

For the non-referred CU, of which r_Δ=1, λ is proportional to Q^2 [33], written as

$$\lambda = a \cdot Q^2. \tag{5}$$

Considering the case of the referred CU, its ΔD is dilated by a factor of r_Δ. That is

$$\widetilde{\lambda} = a \cdot r_\Delta \cdot \widetilde{Q}^2. \tag{6}$$

To the end of $\widetilde{\lambda} = \lambda$, from (5) and (6), we derive that

$$r_\Delta \cdot \widetilde{Q}^2 = Q^2. \tag{7}$$

As $r_\Delta > 1$, (7) illustrates that the heavily referred block should use smaller quantization scale. This is reasonable, because reducing the distortions of the reference CU will improve the quality of all predicted ones efficiently.

In HEVC and H.264/AVC, the relationship between Q and QP is formulated as

$$Q = b \cdot 2^{\frac{QP-12}{6}}. \tag{8}$$

Substituting (8) into (7), we deduce that the difference between $\widetilde{QP}$ and QP as

$$\Delta QP = \widetilde{QP} - QP = -S_\delta \log_2(r_\Delta), \tag{9}$$

where, S_δ is denominated as QP delta strength and its value is 3 as derived by the theoretical analysis. However, the above analysis assumed the ideal information propagation efficiency. In literature [28], the experimental results demonstrated that the optimal value of S_δ is 2, which was adopted by x265.

It should be emphasized that (9) corresponds to the ΔQP of 16×16-CU in the full resolution source picture. When the CU size is larger than 16×16, its QP offset is the average of all 16×16 sub-CUs' ΔQP values. For any CU, the applied QP value is the sum of the frame-level base QP and the specified ΔQP. In x265, the frame-level base quantization scale for the nth frame is defined as

$$Q = \frac{C^n_\Sigma}{R^n_\Sigma} \cdot \left(\frac{0.04}{f_d}\right)^{1-q_c}, \tag{10}$$

where, C^n_Σ is the accumulated complexity since the last scene cut (In x265, C^n_Σ is the variable *m_cplxrSum*, that is updated by functions *rateControlEnd* and *rateControlUpdateStats*.), R^n_Σ is the corresponding target bit number (the variable *m_wantedBitsWindow* in x265), f_d is the frame duration (For example, when the frame rate is 50, the corresponding f_d is equal to 0.02), and q_c is the quantizer curve compression factor (default value is 0.6 in x265). The drawback of the original algorithm comes from the first or the scene cut related I-frame Q value computation. In this case, C_Σ is experimentally formulated as

$$C^0_\Sigma = 0.01 \cdot \left(7 \times 10^5\right)^{q_c} \cdot s, \tag{11}$$

where, s is a scale factor (In x265, when the resolution is larger than 720p, $s = 2.5$; Otherwise, $s = 1$.). The value of R^0_Σ is the averaged bit number of one frame. The above method could make the Q value of the key I-frame even larger than those of the following P-/B-frames. For instance, when encoding the sequence of FourPeople at the constant bitrate 1mb/s, the QP of the first I-frame was 32.88. In contrast, the averaged QP of the following P-frames was merely 28.98. As we know, HEVC provides the *Skip*-Mode coding technique, with which the reference picture block is directly copied without any prediction residues. Because the I-frame in FourPeople contains the stationary background that will be coded in *Skip*-Mode, blurring the key I-frame will degrade

the quality of all following frames that apply the I-frame as the reference.

In the following discussions, we develop the I-frame base QP algorithm by using the HRME results in Section II-B. As we know, the bit amount of I-frame is usually much larger than that of the following P- and B-frames. How to amortise the superfluous bits of the I-frame to the following P-/B-frames is another influential issue, which is explained in Section II-C. The P-frame base QP derivation algorithm is formulated in Section II-D.

B. I-Frame Base QP Determination

As the initial key reference frame, the distortion of I-frame will be inherited by the following P- and B-frames through data dependencies produced by the motion compensation. The importance of I-frame rate-control optimization has been emphasized by the previous works. For example, the QP offsets derived through statistical methods were proposed in the literature [27] to improve the coding performance, the rate accuracy, and the quality smoothness comprehensively. Given the rate of I-frame, the literature [34] optimized the CTU-level rate allocations via Nash bargaining solution to enhance the SSIM metric. The fundamental of our I-frame base QP algorithm is to evaluate the real complexities of the full resolution I-frame and its following P- and B-frames from the HRME frame-level SATD costs. With the estimated full resolution costs and the rate constraints, the optimal I-frame base QP can be derived.

Let R_T denote the desired bit number of one frame. For example, given the target rate of 500kb/s and the 50fps frame rate, R_T is equal to 10kb. R_I is the proper bit number of I-frame and R_{BP} represents the averaged bit number of one frame in the following B-/P-frames. The ratio between R_I and R_{BP} is expressed as

$$\alpha = \frac{R_I}{R_{BP}}. \tag{12}$$

When no scene cut exists, in general, $\alpha \gg 1$. To the end of achieving the averaged R_T in $k+1$ frames, from (12), we have

$$R_T = \frac{R_I + k\frac{R_I}{\alpha}}{k+1},$$

that is

$$R_I = \frac{R_T\,(k+1)\,\alpha}{k+\alpha}. \tag{13}$$

Now, (13) describes the desired bits of the I-frame. With the SATD costs of I-frame (C_I), from $Q = s \cdot C_I / R_I$, we could estimate the value of Q. Although C_I is not available in this phase, the corresponding low-resolution I-frame cost, which is denoted as $\dot{C}_I$, has been derived in HRME. For a picture using the same Q, its original I-frame rate and the corresponding low resolution rate are written as R_I and $\dot{R}_I$, respectively. With the ratio between R_I and $\dot{R}_I$, i.e.,

$$\gamma_I = \frac{R_I}{\dot{R}_I},$$

Algorithm 1 I-Frame QP Derivation

$\acute{QP}_I = 30;$
$\acute{QP}_P = \acute{QP}_I + 1.5;$
$\acute{QP}_B = \acute{QP}_I + 3;$
$\dot{R}_I = \dot{C}_I / f_Q(\acute{QP}_I);$
$\dot{R}_P = \dot{C}_P / f_Q(\acute{QP}_P);$
$\dot{R}_B = \dot{C}_B / f_Q(\acute{QP}_B);$
while $\left|QP_I - \acute{QP}_I\right| > 0.1$ **do**
  $\alpha = \dot{R}_I / \left(a_0(\acute{QP}_I)\dot{R}_P + a_1(\acute{QP}_I)\dot{R}_B\right)$
  $R_I = \alpha R_T\,(k+1)/(k+\alpha)$
  $\dot{R}_I = R_I / \gamma_I(\acute{QP}_I)$
  $QP_I = \acute{QP}_I$
  $\tilde{Q} = s\dot{C}_I / \dot{R}_I$
  $\acute{QP}_I = \acute{QP}_I + 0.5\left(f_{QP}(\tilde{Q}) - \acute{QP}_I\right)$
  $\acute{QP}_P = \acute{QP}_I - \overline{\Delta QP}$
  $\acute{QP}_B = \acute{QP}_P + \Delta QP_{\mathrm{BP}}$
  $\dot{R}_P = \dot{C}_P / f_Q(\acute{QP}_P)$
  $\dot{R}_B = \dot{C}_B / f_Q(\acute{QP}_B)$
end while
$QP_I = \acute{QP}_I - \overline{\Delta QP}$

we have

$$Q = s\frac{C_I}{R_I} = s\frac{\dot{C}_I}{\dot{R}_I} = s\frac{\gamma_I \cdot \dot{C}_I}{R_I}. \tag{14}$$

Given the low-resolution costs, including $\dot{C}_I$, $\dot{C}_P$, and $\dot{C}_B$, and the target rate R_T, we can calculate the QP of I-frame and the key parameter α as described in Algorithm 1. $\dot{C}_P$ and $\dot{C}_B$ are the averaged costs of P-frames and B-frames in HRME, respectively. γ_I is a parameter with variable QP_I that is obtained by statistical method. In specific, by encoding the original sequences and the associated half-resolution sequence under the same constant QP, we achieved the corresponding averaged rates and then got the value of $\gamma_I(QP_I)$. The function $f_{QP}(\cdot)$ maps the quantization scale Q to its quantization parameter QP. The inverse function $f_Q(\cdot)$ maps QP to Q. The scaling factor s is related with the frame resolution. When the resolution is larger than HD720p, $s = 0.53$; Otherwise, $s = 1$. $\overline{\Delta QP}$ is the averaged CU-tree based QP offsets of the current frame. ΔQP_{BP} denotes the QP offset between B-frame and P-frame, of which the default value is 2.271 in x265.

As α is the critical parameter, we would present more details about the derivation of $a_0(\acute{QP}_I)$ and $a_1(\acute{QP}_I)$ on the basis of regression method. For any sequence, we use the constant QP configuration, in which $QP_P = QP_I + 1.5$ and $QP_B = QP_I + 3$, to obtain the target α. For each QP_I, we tested the typical video sequences to develop the overdetermined equations for parameters $a_0(QP_I)$ and $a_1(QP_I)$, which could be solved by least-squares regression. We notice that, with the increase of picture size, the correlation between timing-domain neighboring pictures decreases. For example, as illustrated by Table I, the average $\overline{\Delta QP}$ of the five 416×240 sequences was -5.22. In contrast, this metric in our HD1080p tests degraded to -3.4. The minimum information feedback occurred in the sequence BasketballDrive, of which the value of $\overline{\Delta QP}$

TABLE I
$\overline{\Delta QP}$ IN CLASS B AND CLASS D SEQUENCES

Class	Sequence	$\overline{\Delta QP}$
B	BasketballDrive	-1.3
	BQTerrace	-5.0
	Cactus	-3.7
	Kimono	-2.3
	ParkScene	-4.4
D	BasketballPass	-5.9
	BlowingBubbles	-5.5
	BQSquare	-6.9
	Flowervase	-4.7
	RaceHorses	-3.1

is merely -1.3. Therefore, based on the picture resolution, two sets of parameters were built in our algorithm. The HD1080p and the beyond ones share one set, and other smaller resolutions share the other set.

C. I-Frame Bits Amortization Algorithm

In general, as the I-frame prediction costs are larger than the P- and B-frame counterparts and the I-frame quantization scales are very possibly smaller than those of the following frames, the ratio between R_I and R_{BP}, i.e., α in (12), is much greater than 1. The complexity of the nth encoded frame, i.e., C^n, is defined as the product of the encoded bits and the frame-level base quantization scale. Recalling (10) and (11), if no special treatment is carried out, when encoding the first P-frame next to the I-frame, $C^1_{\sum}$ is equal to $C_I + 0.01 \cdot \left(7 \times 10^5\right)^{q_c} \cdot s$, and $R^1_{\sum} = 2R_T$. The abrupt increase of $C^1_{\sum}$ will dilate the Q value of this P-frame seriously. As this P frame is a key reference, its large fidelity losses will be propagated in the following frames. To ameliorate this adverse impact, x265 amortizes 77%, which is denominated as the amortize fraction (a_F), of the I-frame complexity over the next $k = 68$ frames. The amortize cost (C_a) is written as

$$C_a = \frac{a_F \cdot C_I}{k}.$$

However, the definition $a_F = 77\%$ is a rule of thumb. The accurate value of a_F should be determined by α. From (12) and (13), the amortize factor α_F in our algorithm is formulated as

$$a_F = \frac{R_I - R_T}{R_I} = \frac{k\,(\alpha - 1)}{(k+1)\,\alpha}, \tag{15}$$

where, the default value of k in x265 is 68.

Let $C^n_{\sum}$ denote the sum of complexity until the n-th coded picture. According to aforementioned I-frame QP algorithm and the proposed amortize factor (15), $C^n_{\sum}$ is recast to

$$C^n_{\sum} = \begin{cases} 0 & \text{if } n = 0 \\ (1 - a_F) C_I & \text{if } n = 1 \\ C^{n-1}_{\sum} + a_F \cdot C_I \cdot k^{-1} + C^n & \text{if } 1 < n \le k \\ C^{n-1}_{\sum} + C^n & \text{Others.} \end{cases} \tag{16}$$

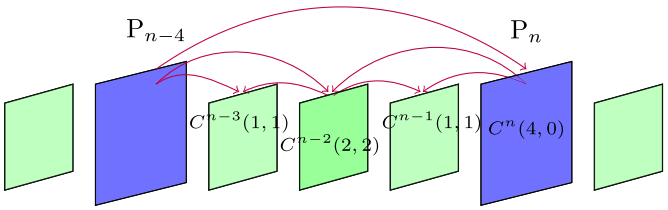


Fig. 3. C_P and C_B definition of a MiniGOP with $N_B = 3$ (MiniGOP under analyze is composed of the P-frame P_n and its leading three B-frames; $C_P = C^n(4,0)$; $C_{I|P} = C^n(0,0)$; C_B is the average of $C^{n-1}(1,1)$, $C^{n-2}(2,2)$ and $C^{n-3}(1,1)$).

D. P-Frame Base QP Derivation

In x265, the frame-level base Q is derived as (10). The main drawback is that, as $C_{\sum}$ is the accumulated complexity of all encoded pictures, the frame-level Q becomes more insensitive to the recent frame complexity changes with the increase of encoded frame number. To overcome this hindrance, the frame base QP development should consider both effects of the long-term complexity history and the lookahead costs in the current MiniGOP to be coded. In the following paragraphs, we will introduce the effects of the current short-term MiniGOP in the P-frame base quantization scale definition.

Let N_B denote the number of B-frames in the current MiniGOP. For example, N_B is equal to 3 in Fig.3. The P-frame SATD cost and the averaged B-frame SATD cost in the current MiniGOP are C_P and C_B, respectively. $C_{I|P}$ is the SATD cost of encoding this P frame with Intra modes. Figure 3 shows the conceptions of C_P, C_B and $C_{I|P}$. If the propagate-in information of the P-frame is PI, the short-term P-frame base quantization parameter, i.e., $\widehat{QP}$, is expressed as

$$\frac{C_P}{f_Q(\widehat{QP} + \overline{\Delta QP})} + \frac{N_B C_B}{f_Q(\widehat{QP} + \Delta QP_{\mathrm{BP}})} = (N_B + 1) R_T + \tau \frac{C_P}{C_{I|P}} PI \frac{1}{f_Q(\widehat{QP})} \tag{17}$$

In (17), the term

$$\tau \frac{C_P}{C_{I|P}} PI$$

represents the information of the current P-frame utilized by the following frames, in which τ is the utilization efficiency. From (9) and (4), substituting ΔQP, $pi^n_{x,y}$ and $c^n_{x,y}(0,0)$ with $\overline{\Delta QP}$, PI and $C_{I|P}$, respectively, we deduced

$$PI = C_{I|P}\left(2^{-\frac{\overline{\Delta QP}}{3}} - 1\right). \tag{18}$$

Substituting (18) and (8) to (17), we obtain

$$\widehat{Q} = \frac{1}{(N_B + 1)\, R_T} \times \left[C_P 2^{\frac{-\overline{\Delta QP}}{6}} - \tau C_P \left(2^{\frac{-\overline{\Delta QP}}{3}} - 1\right) + N_B C_B 2^{\frac{-\Delta QP_{\mathrm{BP}}}{6}}\right].$$

Finally, considering the impact of the quantizer curve compression factor, as shown in (10), the short-term $\widehat{Q}$ is

expressed as

$$\widehat{Q} = \frac{(0.04/f_d)^{1-q_c}}{(N_B+1)\,R_T} \times \left[C_P 2^{\frac{-\overline{\Delta QP}}{6}} - \tau C_P (2^{\frac{-\overline{\Delta QP}}{3}} - 1) + N_B C_B 2^{\frac{-\Delta QP_{\mathrm{BP}}}{6}}\right]. \tag{19}$$

The second term in the square bracket of (19) represents the partial of the P frame residues transmitted to other frames in the motion estimation procedure. The value of $-\overline{\Delta QP}$ indicates the degree of data dependencies to the current P-frame. The principle of our algorithm is that, with the increase of $-\overline{\Delta QP}$, more bits are allocated to this P-frame accordingly. The value of τ depends on the video signal properties. Therefore, τ should be updated according to the deviation of Q, i.e., ∂Q. In x265, ∂Q is defined as

$$\partial Q = \frac{(B^n - n \cdot R_T)Q}{TB}, \tag{20}$$

where, n is the encoded frame number, B^n is the actual used bit number after encoding n frames, $n \cdot R_T$ denotes the target bit number of n frames, and TB is the tolerance buffer (TB is the variable *abrBuffer* defined in x265 version 2.4.). Accordingly, the update formulation of τ is expressed as

$$\tau_{new} = \tau_{old} - \partial Q \frac{0.05(\frac{0.04}{f_d})^{1-q_c} C_P (2^{\frac{-\overline{\Delta QP}}{3}} - 1)}{(N_B+1)\,R_T}, \tag{21}$$

where, τ_{new} is clipped in the range of $[0.05, 0.8]$. The proof of τ update equation is given in Appendix A.

We notice that C_P and C_B are unavailable at present. However, the low resolution costs, i.e., $\dot{C}_P$ and $\dot{C}_B$, have been achieved from lookahead. The ratios $\theta_P = C_P/\dot{C}_P$ and $\theta_B = C_B/\dot{C}_B$ can be estimated during the encoding procedure. Consequently, (19) is recast to

$$\widehat{Q} = \frac{(0.04/f_d)^{1-q_c}}{(N_B+1)\,R_T} \left[\dot{C}_P \theta_P 2^{\frac{-\overline{\Delta QP}}{6}} - \tau \dot{C}_P \theta_P (2^{\frac{-\overline{\Delta QP}}{3}} - 1) + N_B \dot{C}_B \theta_B 2^{\frac{-\Delta QP_{\mathrm{BP}}}{6}}\right]. \tag{22}$$

The frame base Q of the current P-frame, which is the $(n+1)$-th picture to be coded, combines both impacts of the long-term (10) and the short-term (22), that is formulated as

$$Q = 0.3 \cdot \frac{C_{\sum}^n}{n \cdot R_T} \cdot \left(\frac{0.04}{f_d}\right)^{1-q_c} + 0.7 \cdot \widehat{Q}, \tag{23}$$

where, $C_{\sum}^n$ is formulated as (16). For the B frame, its base QP value definition is identical to the original x265 algorithm, which is affected by the related P-frame QP values and ΔQ_{BP}.

III. Quantization Strength Based Dynamic Slice Type Decision

The P-/B-slice type decision problem is stated as follows: Let $\phi = f^{n-1}$ denote the last nominated P-type frame, the subsequence undetermined frame list is $\mathbf{F} = \left\{f^i | n \le i \le n+N-1\right\}$, where n and i represent the frame POC indices. The default value of N is 20 and the last frame, i.e., f^{n+N-1}, is temporally attributed as P type in the slice type decision. The slice type decision is divided into several epochs. In each epoch, we decide the first MiniGOP in $\mathbf{F}$, of which the tailing P-frame is $f^{\nu-1}$, and then, $\mathbf{F}$ is updated by removing all frames belonging to this MiniGOP, i.e., $\mathbf{F} = \left\{f^i | \nu \le i \le n+N-1\right\}$ and $f^{\nu-1}$ is used as ϕ. If $\mathbf{F}$ is not empty, the next epoch will be carried out.

In each epoch, the key target is to find the first P-frame in $\mathbf{F}$. Our fast slice type decision is also based on the HRME (recalling Section II-A). Let ι_C denote the POC of the current frame under analysis in $\mathbf{F}$. $\dot{C}^{\iota_C}(d_0, d_1)$ is the frame-level HRME cost of the current frame f^{ι_C}, of which the forward and the backward POC differences are d_0 and d_1, respectively. $N_I^{\iota_C}(d_0, d_1)$ is the number of Intra-coded 8×8-CUs in HRME processing. N_{CU} is the total number of 8×8-CU in a half-resolution-frame. If any one of the following conditions is satisfied, f^{ι_C} is determined as the P-type frame:

- First, the P prediction cost of $f^{\nu+1}$ using $f^{\nu-1}$ as the reference, i.e., $\dot{C}^{\nu+1}(2,0)$ is derived. If $N_I^{\nu+1}(2,0) > N_{CU}/2$, f^{ν} and $f^{\nu+1}$ in $\mathbf{F}$ are both identified as P-frames.
- Second, if $\dot{C}^{\nu}(1,0)+\dot{C}^{\nu+1}(1,0) < \dot{C}^{\nu}(1,1)+\dot{C}^{\nu+1}(2,0)$, that is the PP type coding costs of $\{f^{\nu}, f^{\nu+1}\}$ is less than their BP type costs, then f^{ν} is assigned as P type.
- If the above two conditions are not met, f^{ν} is B type. Then, the list $\mathbf{F}$ is scanned from $f^{\nu+1}$ to $f^{\nu+B_S-1}$, where B_S denotes the maximum B frame number in a MiniGOP. In this procedure, the frame $f^{\nu-1}$ is applied as the reference to compute $\dot{C}^{\iota_C}(\iota_C - \nu + 1, 0)$, where $\nu + 1 \le \iota_C \le \nu + B_S - 1$. If the following condition is satisfied, i.e.,

$$\dot{C}^{\iota_C}(\iota_C - \nu + 1, 0) > N_{CU} \cdot T, \tag{24}$$

where the parameter T is a quantization strength related threshold, f^{ι_C} is determined as P type. The essence of this P-frame detection algorithm is that, when the prediction differences between the current frame f^{ι_C} and the reference $f^{\nu-1}$ is competitive to the quantization noises, which is a fraction of Q in L1-norm distortion, it is reasonable to assume that, f^{ι_C} could further improve its coding efficiency with the bi-direction prediction, when the quantized residues approaching all-zeros status [35]. Therefore, using B-type for f^{ι_C} is optimal.

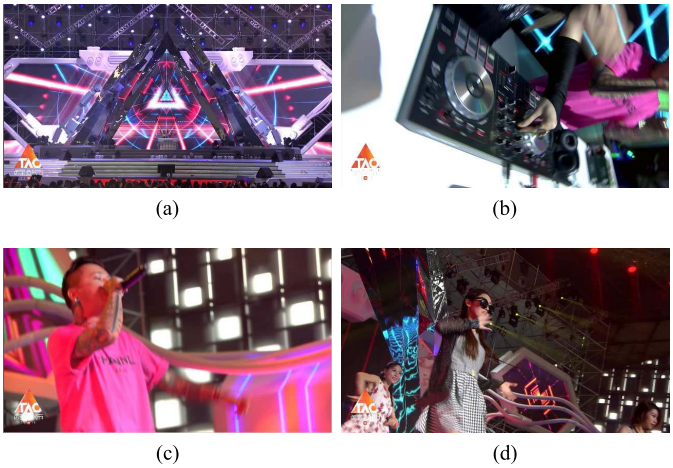
The key threshold T is defined as follows. We first define the low threshold T_L as

$$T_L = \max(300 - 50(\iota_C - \nu - 1), 30),$$

and define the high threshold as

$$T_H = 66 \cdot Q_P,$$

where Q_P is the last P-frame base Q. T_L is devised for the screen-content coding scenarios, in which the weak sampling noises lower the prediction residues. If $\dot{C}^{\nu}(1,0)$ and $\dot{C}^{\nu+1}(1,0)$ are both less than $150 \cdot N_{CU}$, we deem the current video to be a screen-content one, and then



(a) (b) (c) (d)

Fig. 4. Snapshots of the video sequence Zaowujie. (a) POC=0. (b) POC=14. (c) POC=49. (d) POC=59.

use T_L. In summary, T is defined as

$$T = \begin{cases} T_L & \dot{C}^v(1,0) < 150 \cdot N_{CU} \quad \text{and} \\ & \dot{C}^{v+1}(1,0) < 150 \cdot N_{CU} \\ T_L & T_H < 300 \\ T_H & \text{otherwise.} \end{cases}$$

If (24) is not met, f^{v+B_S} is attributed as P-type. Once the P frame is detected, its leading undetermined frames are defined as B-type. The middle position B-frame is the reference B-frame. By introducing Q in the threshold T_H, with the increase of quantization, more frames could be identified as B-frames properly.

After the above processing, the frames in the first MiniGOP are pushed out of the list **F** and the same amount of new pictures are pushed into the tail of **F**. The slice types of all frames in **F** are reset. The last defined P-type frame is used as ϕ in the new round of slice type decision epoch.

IV. Experiments

In this section, we evaluate the coding performance of the proposed rate-control algorithms in terms of coding quality, coding time, rate converge speed, rate accuracy and buffer occupancy. The proposed methods were conducted on x265 version 2.4+96-58b4fa89c42d [36]. The hardware test platform is Huawei RH5885 server, which combines Intel® Xeon™ E7-4830-v2 2.20GHz processor and 128.0GB RAM. The x265 encoder was compiled with GCC version 4.4.7 under CentOS-6.9 operation system.

Twenty-seven typical open test sequences [37], including classes A-F, were tested. In addition, to verify the adaptability, we introduce the in-house 720p sequence Zaowujie, a typical jubilee show video sequence. The snapshots of Zaowujie are shown in Fig. 4, which illustrates high dynamic range background, frequent scene cuts and complex object motions in this sequence.

We carried out the tests on single-threading and multi-threading typical configurations, which are shown in Table II. In the single-threading tests, we removed all parallel encoding tools, such as frame-level parallel and wavefront parallel processing (WPP), to eliminate the adverse effects of parallelism on the parameter estimation and to estimate the timing more accurately, as compared to the multi-threading tests. The tests with multi-threading configurations evaluated the coding quality improvements in parallel encoding applications. Bjøntegaard-Delta Bit-Rate (BDBR) and Bjøntegaard-Delta PSNR (BDPSNR) metrics [38] were employed to compare the coding qualities in this paper.

TABLE II
Coding Configurations

Setting	Parameters
Singlethreading	–frame-thread 1 –no-wpp –no-pmode –no-pme –slices 0 –lookahead-slices 0 –psnr –tune psnr –b-adapt 1 –bframes 8
Multithreading	–frame-thread 4 –psnr –tune psnr –b-adapt 1 –bframes 8

TABLE III
Coding Quality of Single-Threading Medium Configuration

Class	Sequence	BP[dB]	BR[%]	ΔT[%]
A	PeopleOnStreet	0.1029	-02.46	+1.07
	Traffic	0.6232	-21.28	+1.59
B	BasketballDrive	0.1524	-05.38	+3.36
	BQTerrace	0.5113	-29.52	+2.33
	Cactus	0.4367	-15.62	+0.23
	Kimono	0.1358	-04.61	-3.38
	ParkScene	0.6103	-19.61	+2.50
C	BasketballDrill	0.6759	-17.91	+1.25
	BQMall	0.5025	-12.51	+1.14
	Flowervase	0.3813	-12.56	+0.77
	PartyScene	0.6773	-20.21	+0.84
	RacehorsesC	0.1576	-04.90	+1.65
D	BasketballPass	0.5123	-10.62	+5.27
	BlowingBubbles	0.8624	-21.64	+5.60
	BQSquare	1.7205	-41.37	+2.29
	Flowervase	0.5058	-14.13	+0.41
	RaceHorses	0.2321	-05.37	+1.75
E	Vidyo1	0.5723	-22.76	+1.34
	Vidyo3	0.6529	-20.71	+2.53
	Vidyo4	0.6304	-22.52	+2.45
	KristenAndSara	0.7858	-28.52	-0.47
	Johnny	0.8728	-46.77	+3.23
	FourPeople	0.9236	-28.77	+2.89
	Zaowujie	0.4846	-08.42	-9.27
F	SlideEditing	0.1211	-00.48	+2.15
	ChinaSpeed	0.3779	-10.84	-2.51
	SlideShow	1.4936	-17.29	+1.59
	BasketballDrilltext	0.7094	-18.60	-0.87
	Average	0.5866	-17.32	+1.13

The coding performance results of single-threading are shown in Table III. The original x265 version 2.4 is the standard benchmark. BP and BR in Table III stand for the average PSNR difference (BDPSNR) and the average bit rates difference (BDBR) [38], respectively. ΔT represents the encoding time reduction, which is defined as

$$\Delta T = \frac{T_{\text{org}} - T_{\text{opt}}}{T_{\text{org}}} \times 100\%,$$

with T_{org} and T_{opt} denoting the original x265 encoding time and the time of x265 with our optimizations, respectively. In these tests, we fixed the single encoding thread to a specific core. This method could reduce the timing variance caused by the process migration. On average, 0.5866dB coding quality gain, or equivalently 17.32% rate reduction, was achieved by our proposals. Meanwhile, our algorithms reduced the encoding time by 1.13% on average. The time saving of our algorithms came from the proposed slice type decision scheme. As we know, the early-termination (ET) method is adopted in x265 Rate-Distortion-Optimization (RDO). When more pictures were identified as B-type properly, more RDO timing was saved by the ET method. For example, as the B-frame number of BasketballPass was increased by 52%, its encoding time was reduced by 5.27% accordingly.

For the videos with stationary background, such as BasketballDrill, FourPeople and KristenAndSara, our I-frame base QP derivation algorithm improved the quality of prominent I-frame, which was the key reference for the following pictures. For example, under 1.5mb/s rate constraints, the I-frame QP value of FourPeople derived by our algorithm is 24.6 (P-frame QP:24.82; B-frame QP:31.31). In contrast, the I-frame QP in original x265 is 32.87 (P-frame QP:26.37; B-frame QP:32.24). On the other hand, for the videos with fast scene cuts, such as Zaowujie and SlideShow, our algorithm dropped the I-frame quality adaptively. Specifically, when the Zaowujie I-frame QP was 36.66 in the original x265, our algorithm increased it to 37.34. The coding gains of Zaowujie and SlideShow mainly come from the proposed P-frame base QP calculation scheme.

The R-D performance analysis of multi-threading configurations is illustrated by Table IV. As compared to the original x265, the averaged coding quality gain of our algorithms is 0.6165dB, which is 0.02dB higher than the single-threading counterpart. This trivial improvement is caused by our amendments of the bugs in x265. For example, the flaws of skip cost estimation in HRME degraded the CU-tree accuracy especially with multi-slice parallel processing.

The coding quality statistics of our algorithms under other presets, including *veryfast*, *faster*, *fast*, *slow*, *slower*, *veryslow* and *placebo*, are illustrated in Table V and Table VI. We observed that, with the increase of encoding complexity, the coding quality gain of our algorithm boosted up accordingly. Specifically, as shown in Table V, under the *veryfast* configurations, our algorithms achieved 0.5083dB BDPSNR gain averagely. This metric was increased up to 0.6643dB under the *placebo* configurations. This phenomenon was caused by the augment of the frame number in lookahead procedure. For example, the *veryfast* specified the lookahead frame number as 10; In contrast, this parameter was increased to 60 in *placebo*. Introducing more frames in lookahead boosted the estimation accuracy of our algorithms.

It should be noted that the coding quality of SlideEditing in multi-threading *slow* configuration was lower than the original x265. This BDPSNR degradation came from three aspects. First, recalling (19), (20) and (21), the P-frame base Q is derived from the statistical information B^n, i.e., the real encoding rate. Under the multi-threading configurations, to speed up the processing, B^n was obtained from the partially encoded frames. For screen content videos, such as SlideEditing and SlideShow, because the complexity changes rapidly even in the same frame, the estimation accuracy of B^n dropped accordingly. Second, our P-frame base Q algorithm targets for not only the quality improvement, but also the specified bit rates. Compared to the original x265, the bit rate accuracy was enhanced obviously. As shown in Fig.5, when the target rate is 1mb/s, the actual rate of x265 is 549.99kb/s, and the corresponding PSNR is 45.331dB. In contrast, our algorithm improved the rate utilization by 32.4% with 5.160dB PSNR melioration. Finally, we observed that the RD-curve gap was affected by the number of the samples which were used to fit the curves. In Fig.5, "x265/Ours-8point" and "x265/Ours-4point" represent the RD-curves fitted with eight samples and four samples, respectively. The quality loss of our algorithm mainly occurred at low rates. This gap was dilated when using four samples to fit the curves, which was the case in our BDPSNR calculation.

The coding quality comparisons between the proposed algorithms with other up-to-date rate-control methods are listed in Table VII. The λ-domain rate-control algorithm implemented in HM16.17 [5], [6] was adopted as the standard benchmark. The comparisons are not fair, because x265 did not implement all encoding techniques, such as hierarchical prediction structure with multiple dyadic hierarchy stages and bi-prediction P-frame. Nevertheless, the averaged BDPSNR

TABLE IV
CODING QUALITY OF MULTI-THREADING MEDIUM CONFIGURATION

Class	Sequence	BP[dB]	BR[%]
A	PeopleOnStreet	0.1077	-02.36
	Traffic	0.6568	-22.02
B	BasketballDrive	0.1830	-06.27
	BQTerrace	0.5258	-28.76
	Cactus	0.4909	-16.57
	Kimono	0.1794	-05.98
	ParkScene	0.6257	-19.92
C	BasketballDrill	0.6896	-17.93
	BQMall	0.5091	-12.30
	Flowervase	0.5067	-16.03
	PartyScene	0.6864	-20.16
	RacehorsesC	0.1744	-05.33
D	BasketballPass	0.5806	-12.01
	BlowingBubbles	0.8623	-21.54
	BQSquare	1.7050	-40.89
	Flowervase	0.6446	-17.44
	RaceHorses	0.2345	-05.40
E	Vidyo1	0.6461	-24.22
	Vidyo3	0.7412	-21.63
	Vidyo4	0.6762	-22.97
	KristenAndSara	0.9512	-32.15
	Johnny	1.0332	-45.86
	FourPeople	0.9933	-28.99
	Zaowujie	0.4636	-07.80
F	SlideEditing	0.3654	-02.80
	ChinaSpeed	0.3784	-10.72
	SlideShow	0.9296	-11.17
	BasketballDrilltext	0.7222	-18.61
	Average	0.6165	-17.78

TABLE V
CODING QUALITY OF OTHER PRESETS WITH SINGLE-THREADING (BDPSNR:dB)

Class	Sequence	Preset						
		veryfast	faster	fast	slow	slower	veryslow	placebo
A	PeopleOnStreet	0.0562	0.0544	0.0202	0.0880	0.1434	0.1794	0.1970
	Traffic	0.6091	0.6098	0.6052	0.6349	0.6313	0.6208	0.6512
B	BasketballDrive	0.0868	0.0867	0.1178	0.1771	0.2550	0.2579	0.2634
	BQTerrace	0.4520	0.4641	0.5199	0.5890	0.5571	0.5552	0.5245
	Cactus	0.1949	0.2025	0.3614	0.5276	0.5438	0.5820	0.5692
	Kimono	0.1015	0.0919	0.1151	0.1657	0.1710	0.1742	0.1791
	ParkScene	0.3763	0.3832	0.5038	0.6743	0.6923	0.7064	0.7259
C	BasketballDrill	0.5020	0.4843	0.6329	0.8439	0.9707	1.0998	1.1151
	BQMall	0.3485	0.3402	0.4059	0.6103	0.6141	0.7823	0.8468
	Flowervase	0.3762	0.3747	0.3289	0.3861	0.3816	0.3669	0.3661
	PartyScene	0.4167	0.4130	0.5536	0.6684	0.7367	0.7561	0.7180
	RacehorsesC	0.1030	0.1031	0.1451	0.2044	0.2284	0.2442	0.2530
D	BasketballPass	0.4491	0.4542	0.4877	0.5699	0.6203	0.6308	0.6738
	BlowingBubbles	0.5597	0.5596	0.7645	0.8696	0.9004	0.9488	0.9727
	BQSquare	1.5276	1.5285	1.6398	1.7135	1.7289	1.6763	1.7095
	Flowervase	0.5262	0.5310	0.5199	0.5017	0.4939	0.4896	0.5085
	RaceHorses	0.1733	0.1825	0.1963	0.2303	0.2473	0.2754	0.2701
E	Vidyo1	0.5076	0.5194	0.5716	0.5588	0.5399	0.5509	0.5238
	Vidyo3	0.5720	0.5864	0.6351	0.6812	0.6725	0.6705	0.6461
	Vidyo4	0.5095	0.5085	0.6226	0.6108	0.5572	0.5611	0.5588
	KristenAndSara	0.7455	0.7467	0.8073	0.7450	0.7298	0.7403	0.7287
	Johnny	0.8191	0.8210	0.8974	0.8921	0.8856	0.8925	0.8381
	FourPeople	0.8889	0.8906	0.9514	0.9218	0.9025	0.8750	0.8342
	Zaowujie	0.4739	0.4812	0.4609	0.5007	0.5277	0.5213	0.5059
F	SlideEditing	0.4589	0.4252	0.4172	0.5641	0.1064	0.3098	0.4964
	ChinaSpeed	0.3345	0.3302	0.3254	0.4036	0.4524	0.4736	0.4863
	SlideShow	1.5457	1.5252	1.4944	1.4181	1.4582	1.4033	1.2757
	BasketballDrilltext	0.5177	0.5051	0.6621	0.8850	1.0172	1.1458	1.1620
	Average	0.5083	0.5073	0.5630	0.6299	0.6305	0.6604	0.6643

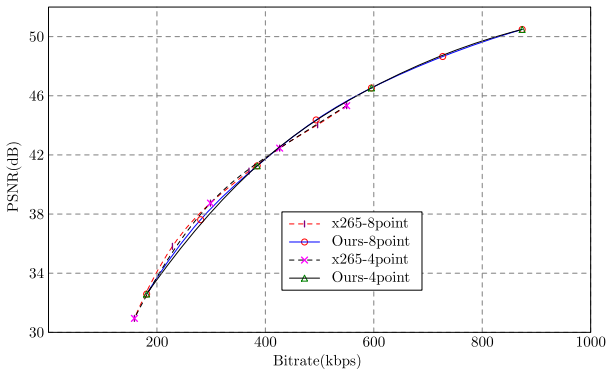


Fig. 5. SlideEditing RD-curve comparisons between the original x265 and the proposed algorithm (rate constraints: 1mb/s, 875kb/s, 750kb/s, 625kb/s, 500kb/s, 375kb/s and 250kb/s).

gain of our proposals was 0.847dB. As compared to the up-to-date machine-learning based rate-control algorithm [22], our work outperformed with 0.513dB BDPSNR improvement.

The comparisons in terms of Bjøntegaard-Delta Structural Similarity (BDSSIM) between the original x265 and ours are illustrated in Table VIII. Under different presets, 0.0083-0.0104 BDSSIM gains were achieved by our rate-control optimizations. The subjective fidelity analysis is shown in Fig.6. As compared with x265, more details were preserved by our algorithms. For example, two pictures in Fig. 6a are 650 × 620 region in the 248th picture of SlideEditing. It was observed that the slash pattern hatch in the background, which was blurred by x265, was reserved properly by our algorithms. The subjective quality gap was dilated with the decrease of target rate. In the comparisons between Fig.6d and Fig.6e, we noticed that the structural deformation of the face outlines, the compression artifacts (including the blocking and the ring noises) became more conspicuous as reducing the rate from 1mb/s to 300kb/s. Figure 6b demonstrated the advantage of our P-frame base QP algorithm. In particular, the 217th picture of SlideShow was detected as one heavily depended reference, and then its QP was reduced accordingly by (22). As compared with x265, the PSNR of the 217th picture was enhanced by 6.44dB, which contributed to the quality melioration in the following 36 predicted pictures. Another prominent issue leading to our subjective fidelity improvement is our better rate-control accuracy, as illustrated in Fig.5. This can be traced by analogy for other sequences, such as, KristenandSara and PartyScene. For example, given the 100kb/s target rate for KristenandSara, the actual rate of the original x265 was 70.43kb/s, while our algorithm achieved 101.35kb/s.

Quality consistency is one prominent metric, which is used to evaluate the impact on human visual perception. The quality inconsistency comparisons in terms of the frame-level

TABLE VI
CODING QUALITY OF OTHER PRESETS WITH MULTI-THREADING (BDPSNR:dB)

Class	Sequence	Preset						
		veryfast	faster	fast	slow	slower	veryslow	placebo
A	PeopleOnStreet	0.0647	0.0561	0.0240	0.1055	0.2067	0.1769	0.2068
	Traffic	0.5454	0.5476	0.5504	0.6889	0.5452	0.6833	0.7557
B	BasketballDrive	0.1192	0.1244	0.1472	0.2090	0.3353	0.3008	0.2887
	BQTerrace	0.5159	0.4725	0.5189	0.5298	0.6148	0.5667	0.5528
	Cactus	0.2475	0.2555	0.3908	0.5701	0.6420	0.6256	0.6147
	Kimono	0.1589	0.1580	0.1717	0.2122	0.2437	0.2138	0.2210
	ParkScene	0.3979	0.4045	0.5121	0.6924	0.7429	0.7222	0.7458
C	BasketballDrill	0.5078	0.5218	0.6512	0.8596	1.0463	1.1199	1.1266
	BQMall	0.3476	0.3410	0.4214	0.5945	0.6785	0.7714	0.8437
	Flowervase	0.5113	0.5074	0.4572	0.5047	0.5307	0.4796	0.4841
	PartyScene	0.4338	0.4469	0.5608	0.6673	0.7908	0.7485	0.7247
	RacehorsesC	0.1320	0.1371	0.1685	0.2379	0.2833	0.2627	0.2742
D	BasketballPass	0.5258	0.5318	0.5508	0.6247	0.7180	0.6994	0.7218
	BlowingBubbles	0.5872	0.5881	0.7553	0.8563	0.9694	0.9808	0.9904
	BQSquare	1.5289	1.5319	1.6121	1.7257	1.7768	1.7175	1.7263
	Flowervase	0.6610	0.6797	0.6463	0.6443	0.6454	0.6277	0.6433
	RaceHorses	0.1871	0.2005	0.2210	0.2471	0.2947	0.2946	0.2917
E	Vidyo1	0.5923	0.5784	0.6519	0.6465	0.6941	0.6202	0.6058
	Vidyo3	0.7316	0.7276	0.7663	0.7598	0.8875	0.7470	0.7028
	Vidyo4	0.5652	0.5479	0.6625	0.6923	0.7082	0.6253	0.6523
	KristenAndSara	0.9201	0.9197	0.9670	0.8851	0.9372	0.8707	0.8606
	Johnny	0.9545	0.9681	1.0246	1.0081	1.0424	0.9758	0.9205
	FourPeople	0.9724	0.9717	1.0267	0.9920	1.0356	0.9552	0.8951
	Zaowujie	0.4362	0.4336	0.4298	0.5007	0.6464	0.4985	0.4848
F	SlideEditing	0.0262	-0.0265	0.0344	-0.3361	0.6073	-0.2460	-0.0873
	ChinaSpeed	0.3386	0.3273	0.3259	0.4052	0.5169	0.4946	0.5007
	SlideShow	1.1710	1.1437	1.2899	1.2132	2.1011	1.2544	1.3116
	BasketballDrilltext	0.5289	0.5435	0.6859	0.9052	1.0971	1.1670	1.1773
	Average	0.5253	0.5229	0.5794	0.6299	0.7621	0.6770	0.6870

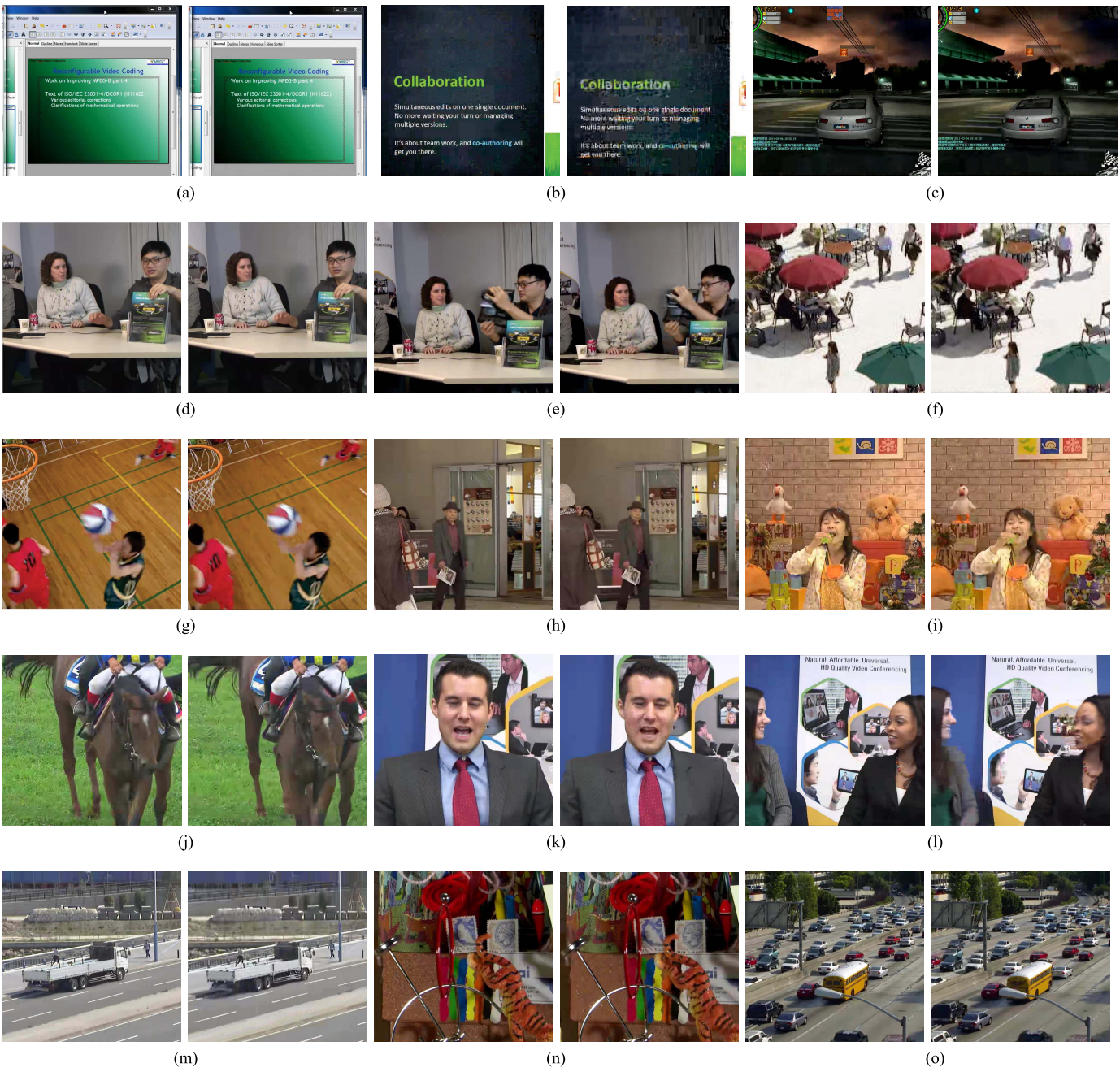
TABLE VII
CODING QUALITY COMPARISONS (BDPSNR:dB)

Class	TIP-Wang[21]	TIP-Gao[22]	x265*	Proposed*
A	-0.414	0.395	0.162	0.582
B	-0.612	0.417	0.126	0.569
C	-0.343	0.305	0.136	0.798
D	-0.073	0.341	0.118	0.937
E	0.353	0.213	0.705	1.349
Average	-0.218	0.334	0.249	0.847

* with the option "–preset placebo" and singlethreading configurations as shown in Table II

PSNR standard deviation were illustrated in Table IX. It was observed that our quality deviation of the low-resolution videos (classes C and D) was 1.503dB averagely, 0.960dB larger than the average quality fluctuations of the higher resolution counterparts (classes A, B and E). This phenomenon was caused by two reasons. The first reason is the CU-tree algorithm of x265. As shown in Table I, the amplitude of QP offsets always diminishes with the increase of frame resolution, which means the lower quality fluctuation. The second reason lies on the intrinsic scene changes of the coded videos. In specific, the maximum quality fluctuations (2.451dB in our algorithm and 1.882dB in the original x265) came from BQMall, of which the real-time frame grain PSNR statistics are shown in Fig. 7. In the beginning 150 frames, the foreground peoples with fast and complex motions made the rate-control module reducing the quality to the minimum 29.581dB in 83rd frame to reach the target rate. After that, with the shrinkage of complex motion areas, the smaller frame-level QP values were applied under the same rate constraints. Accordingly, the quality was raised to 38.138dB in the 385th frame with our algorithm.

As compared to the original x265, the frame-level PSNR standard deviation was deteriorated by 0.159dB averagely. Our slice type decision and P-frame base QP algorithms were the main reasons. Let us still use BQMall to analyze this phenomenon. With our slice type decision, the percentage of B-frame number in BQMall stream was increased by 33.8%, which increased the amount of propagate-in information of the dependency P-frames. Recalling (4) and (9), this would increase the QP offsets of P-frames consequently. On the other hand, as shown by (22), because our P-frame base QP algorithm could detect and respond to the complexity changes sensitively, our frame-level PSNR improvements were faster than the original x265. For instance, in the 297th frame, 2.4dB frame-level PSNR gain was achieved. Moreover, the following rate accuracy analysis, as illustrated by Fig.8, will reveal that these quality meliorations were achieved by meeting the given rate constraints accurately. The overall quality consistency of our algorithms is competitive to the performance of [22].



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

Fig. 6. Subjective quality analysis (In each sub-figure, the left picture is the snapshotting of the video encoded by our proposals, and the right picture is the corresponding one from the original x265). (a) SlideEditing@400kb/s (our PSNR=46.36, x265 PSNR=43.85). (b) SlideShow@200kb/s (our PSNR=35.61dB, x265 PSNR=29.17dB). (c) ChinaSpeed@300kb/s (our PSNR=32.02dB, x265 PSNR=31.49dB). (d) FourPeople@1mb/s (our PSNR=42.76dB, x265 PSNR=40.83dB). (e) FourPeople@300kb/s (our PSNR=39.14dB, x265 PSNR=37.65dB). (f) BQSquare@150kb/s (our PSNR=31.70dB, x265 PSNR=29.93dB). (g) BasketballDrill@400kb/s (our PSNR=35.36dB, x265 PSNR=33.14dB). (h) BQMall@500kb/s (our PSNR=36.80dB, x265 PSNR=33.88dB). (i) PartyScene@300kb/s (our PSNR=29.82dB, x265 PSNR=27.38dB). (j) RaceHorsesC@300kb/s (our PSNR=28.36dB, x265 PSNR=27.16dB). (k) Johnny@100kb/s (our PSNR=38.65dB, x265 PSNR=36.77dB). (l) KristenandSara@100kb/s (our PSNR=37.18dB, x265 PSNR=33.01dB). (m) BQTerrace@500kb/s (our PSNR=32.90dB, x265 PSNR=30.39dB). (n) Cactus@1mb/s (our PSNR=34.48dB, x265 PSNR=33.51dB). (o) Traffic@500kb/s (our PSNR=33.65dB, x265 PSNR=31.09dB).

The rate accuracy is another critical performance for a rate-control module. The rate accuracy comparisons are shown in Table X. Compared with the counterparts [21] and [22], our rate accuracy was 1.8-2.8% lower. This was because our CU-tree compatible rate control algorithms merely strived for the target rate by adjusting the key frame-level base QP values. In contrast, the algorithms in [21] and [22] reached the target rate by the CU-level QP adjustment. As compared to the original x265, our rate accuracy was improved by 4.2% averagely leveraging on the proposed P-frame base QP algorithm.

TABLE VIII
SUBJECT CODING QUALITY COMPARISONS IN BDSSIM

Class	Preset							
	veryfast	faster	fast	medium	slow	slower	veryslow	placebo
A	0.0057	0.0057	0.0053	0.0072	0.0070	0.0071	0.0070	0.0072
B	0.0037	0.0037	0.0048	0.0062	0.0072	0.0075	0.0075	0.0068
C	0.0161	0.0163	0.0173	0.0194	0.0186	0.0186	0.0188	0.0190
D	0.0089	0.0088	0.0103	0.0115	0.0119	0.0122	0.0126	0.0127
E	0.0032	0.0035	0.0038	0.0036	0.0041	0.0040	0.0039	0.0038
F	0.0121	0.0125	0.0125	0.0145	0.0134	0.0127	0.0122	0.0123
Average	0.0083	0.0084	0.0090	0.0104	0.0104	0.0104	0.0103	0.0103

TABLE IX
QUALITY CONSISTENCE COMPARISONS (dB)

Class	TIP-Wang [21]	TIP-Gao [22]	HM16.17	x265	Proposed
A	2.213	0.938	1.780	0.417	0.559
B	2.134	0.844	1.173	0.498	0.591
C	2.136	0.850	1.408	1.417	1.642
D	1.708	1.213	1.749	1.161	1.363
E	1.037	0.696	1.362	0.349	0.480
Average	1.846	0.908	1.495	0.768	0.927

TABLE X
BIT RATE ACCURACY COMPARISONS (%)

Class	TIP-Wang [21]	TIP-Gao [22]	HM16.17	x265	Proposed
A	89.390	97.851	99.260	96.432	98.046
B	94.990	96.931	99.911	91.210	96.798
C	89.277	99.825	99.819	93.303	95.871
D	77.631	99.630	99.660	94.050	95.460
E	96.324	99.433	99.730	88.602	98.396
Average	89.522	98.734	99.676	92.719	96.914

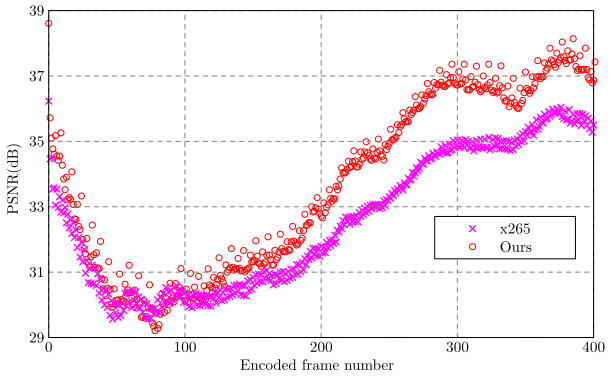


Fig. 7. Frame-level PSNR statistics of BQMall with 500kb/s target rate constraints.

In x265, after encoding n frames, the actual real-time rate R^n is defined as

$$R^n = \frac{B^n}{n \cdot f_d}.$$

The R^n curve comparisons between the original x265 and our optimizations are shown by Fig.8, with which we can

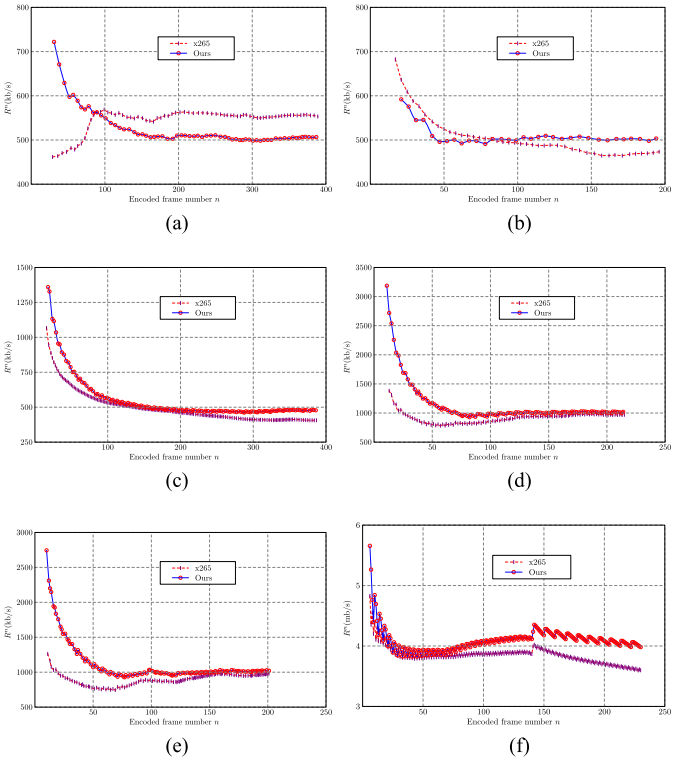


Fig. 8. Comparisons of real-time rate R^n with respect to coded frame number between our proposals and original x265 rate-control algorithm. (a) BasketballPass@500kb/s. (b) RaceHorsesD@500kb/s. (c) BQMall@500kb/s. (d) FourPeople@1000kb/s. (e) KristenAndSara@1000kb/s. (f) Kimono1@4mb/s.

evaluate the converge speed and the converge accuracy of one rate-control module. For the consistent scene sequences, such as FourPeople and KristenAndSara, our rate-control algorithms achieved the target rate faster than the original x265. For example, in the test of FourPeople, our algorithms reached the rate of 1mb/s at the 70th frame; In contrast, x265 got the target rate after the 170th frame. Our algorithms outperformed x265 when encoding the sequences being plethora of complex motions and scene changes, for example, the other four sequences in Fig.8. Let us investigate the R^n curves of BQMall. With introducing the short-term quantization $\widehat{Q}$ that is derived from lookahead, our P-frame base QP is more sensitive to the future changes of prediction costs. As shown in Fig.7, around the 170th frame, our rate-control module detected the following cost reduction, and

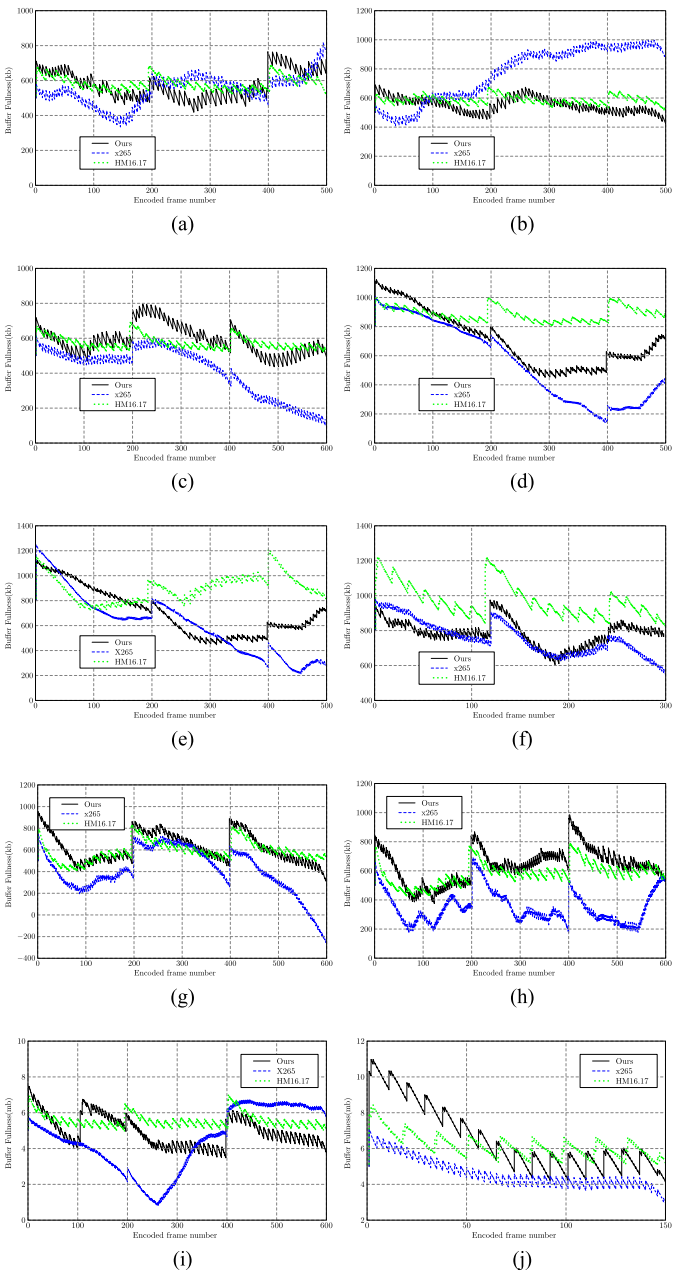


Fig. 9. Buffer occupancy comparisons. (a) BlowingBubbles@500kb/s. (b) BasketballPass@500kb/s. (c) BQSquare@500kb/s. (d) BQMall@500kb/s. (e) PartyScene@500kb/s. (f) RaceHorsesC@500kb/s. (g) FourPeople@1mb/s. (h) KristenAndSara@1mb/s. (i) BQTerrace@6mb/s. (j) Traffic@8mb/s.

then improved the quality by decreasing the frame-level base QP values. x265 also adjusted the quality under the similar trend, but its intensity was weaker than ours. In consequence, Fig.8c revealed that, as compared to x265, our rate was more accurately maintained. For Kimono1, the scene cut occurs at the 141st frame. Our algorithm converged to the target rate within 70 frames. In contrast, the rate of the original x265 deviated from the target rate by 10.0%.

In the practical video transmission system, the output buffer is required to resolve the mismatch between the actual real-time generated bit number and the bandwidth constraints. The buffer occupancy statistics of our proposals, the original x265, and HM16.17 are illustrated in Fig.9. The peak-to-valley gap of a buffer occupancy curve represents the minimum buffer size requirements that can prevent overflow and underflow cases. As compared to the original x265, for the videos with complex motions and scene changes, such as BasketballPass, BQMall, RaceHorsesC, etc., our algorithms reduced the fluctuation of buffer occupancy, which indicated less overflow or underflow cases. For the videos with stationary scene, as compared to x265, our algorithms allocated more bits to the first I-frame to improve the picture quality, which might consume more buffer resources. The worst case was Traffic, in which the required buffer size of our algorithms was increased to 7mb from the original 4mb. However, the gain was an averaged 0.6dB picture quality improvements. The metric of buffer occupancy of our proposals and the original x265 both lagged behind HM16.17 and the work of [22]. This is an intrinsic challenge of the MB-tree/CU-tree technique. As compared to [6] and [22], the CU-tree based rate-control algorithms, including ours, merely strive for the target rates by adjusting the frame-level base QP, because the CU-level QP offsets have been decided. The advantage of CU-tree is better picture quality. In contrast, its disadvantages include lower rate accuracy and larger buffer occupancy. However, Table X and Fig.9, illustrated that the proposed optimizations ameliorated the above two symptoms of the original x265 CU-tree rate-control module.

V. Conclusion

The proposed rate-control algorithms improved the coding quality and the rate accuracy of x265 leveraging on the information from lookahead procedure. The contributions of our work are summarized as follows: (1) By exploring lookahead, we estimated the full resolution frame-level coding complexities and the information propagations, with which we could derive the crucial I- and P-frames' base QP values accurately. (2) With considering the impacts of quantization in the threshold definitions, we enhanced the efficiency of slice type decision algorithm to detect more B-frames properly. As a result, the coding quality and the rate accuracy were both ameliorated. Experiments revealed that, with the default configurations (–preset medium), an averaged 0.617dB and up to 1.705dB BDPSNR quality gains were achieved by our proposals, while reducing the encoding time by 1.13%. In addition, the rate accuracy was improved by 4.2% on average with our frame-level base QP determination algorithms. Finally, it was demonstrated that our algorithms outperformed the original x265 rate-control module in terms of rate converge speed and buffer occupancy control as well.

Appendix A
Transmission Efficiency τ Update Equation

Let Q_{opt} denote the optimal quantization scale value, Q_{cur} is the current quantization scale value. The squared error between Q_{opt} and Q_{cur} is

$$e^2 = \left(Q_{opt} - Q_{cur}\right)^2.$$

Therefore, the derivative of e^2 to τ has the form of

$$\frac{\partial e^2}{\partial \tau} = \left(Q_{opt} - Q_{cur}\right)\left(\frac{0.04}{f_d}\right)^{1-q_c} \frac{C_P(2^{\frac{-\Delta QP}{3}} - 1)}{(N_B + 1)\, R_T}. \quad \text{(A.1)}$$

The decent direction is equal to $-\frac{\partial e^2}{\partial \tau}$.

Then, we have

$$\tau_{new} = \tau_{old} - \delta \frac{\partial e^2}{\partial \tau}. \quad \text{(A.2)}$$

Considering $Q_{opt} - Q_{cur}$ is proportion to ∂Q, which is formulated as (20), we define that

$$\delta\left(Q_{opt} - Q_{cur}\right) = 0.05 \partial Q. \quad \text{(A.3)}$$

Substituting (A.3) and (A.1) to (A.2), we obtain τ update scheme as

$$\tau_{new} = \tau_{old} - \partial Q \frac{0.05(0.04/f_d)^{1-q_c} C_P(2^{\frac{-\Delta QP}{3}} - 1)}{(N_B + 1)\, R_T}.$$

REFERENCES

[1] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 533–545, Sep. 1994.

[2] *MPEG-2 Video Test Model 5*, document ISO/IEC JTC1/SC29/WG11 Doc N, Test Model Editing Committee, vol. 400, 1993.

[3] *ITU-T/SG15 TMN 8*, Video Codec Test Model, Portland, OR, USA, Jun. 1997.

[4] Z. Li, F. Pan, P. Lim, G. Feng, X. Lin, and S. Rahardja, *Adaptive Basic Unit Layer rate Control for JVT*, document JVT-G012, JVT, Pattaya, Thailand, Mar. 2003.

[5] B. Li, H. Li, L. Li, and J. Zhang, *Rate Control by R-Lambda Model for HEVC*, document JCTVC-K0103, Shanghai, China, Oct. 2012.

[6] B. Li, H. Li, L. Li, and J. Zhang, "λ Domain rate control algorithm for high efficiency video coding," *IEEE Trans. Image Process.*, vol. 23, no. 9, pp. 3841–3854, Sep. 2014.

[7] VideoLAN. *x264*. Accessed: Jun. 2016. [Online]. Available: https://www.videolan.org/developers/x264.html

[8] MulticoreWare Inc. *x265 HEVC Encoder/H.265 Video Codec*. (2017). [Online]. Available: http://x265.org

[9] *NVENC-NVidia Hardware Video Encoder (NVENC)*, document NVENC_DA-06209-001_v04, NVIDIA, Jul. 2014.

[10] M. Jiang and N. Ling, "Low-delay rate control for real-time H.264/AVC video coding," *IEEE Trans. Multimedia*, vol. 8, no. 3, pp. 467–477, Jun. 2006.

[11] H.-C. Fang, T.-C. Wang, Y.-W. Chang, and L.-G. Chen, "Hardware oriented rate control algorithm and implementation for realtime video coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 2, Apr. 2003, p. II-489.

[12] J. Ribas-Corbera and S. Lei, "Rate control in DCT video coding for low-delay communications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 172–185, Feb. 1999.

[13] T. Berger, *Rate Distortion Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, 1971.

[14] H.-J. Lee, T. Chiang, and Y.-Q. Zhang, "Scalable rate control for MPEG-4 video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 878–894, Sep. 2000.

[15] T. Chiang and Y.-Q. Zhang, "A new rate control scheme using quadratic rate distortion model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 246–250, Feb. 1997.

[16] W. Ding and B. Liu, "Rate control of MPEG video coding and recording by rate-quantization modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 1, pp. 12–20, Feb. 1996.

[17] S. Mallat and F. Falzon, "Analysis of low bit rate image transform coding," *IEEE Trans. Signal Process.*, vol. 46, no. 4, pp. 1027–1042, Apr. 1998.

[18] Z. He, Y. K. Kim, and S. Mitra, "Low-delay rate control for DCT video coding via ρ-domain source modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 8, pp. 928–940, Aug. 2002.

[19] B. Bross, W.-J. Han, J.-R. Ohm, J. G. Sullivan, Y.-K. Wang, and T. Wiegand, *High Efficiency Video Coding (HEVC) Text Specification Draft 10*, document JCT-VC-L1003, Geneva, Switzerland, 2013.

[20] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[21] M. Wang, K. N. Ngan, and H. Li, "Low-delay rate control for consistent quality using distortion-based Lagrange multiplier," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 2943–2955, Jul. 2016.

[22] W. Gao, S. Kwong, and Y. Jia, "Joint machine learning and game theory for rate control in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 26, no. 12, pp. 6074–6089, Dec. 2017.

[23] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 23–50, Nov. 1998.

[24] Y. Sermadevi and S. S. Hemami, "Efficient bit allocation for dependent video coding," in *Proc. Data Compress. Conf.*, Mar. 2004, pp. 232–241.

[25] Y. Sermadevi, J. Chen, S. S. Hemami, and T. Berger, "When is bit allocation for predictive video coding easy?" in *Proc. Data Compress. Conf.*, Mar. 2005, pp. 289–298.

[26] T. Toivonen, L. Merritt, V. Ojansivu, and J. Heikkila, "A new rotation search for dependent rate-distortion optimization in video coding," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, vol. 1, Apr. 2007, pp. 1165–1168.

[27] W. Gao, S. Kwong, H. Yuan, and X. Wang, "DCT coefficient distribution modeling and quality dependency analysis based frame-level bit allocation for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 139–153, Jan. 2016.

[28] J. Garrett-Glaser, "A novel macroblock-tree algorithm for high-performance optimization of dependent video coding in H.264/AVC," Dept. Comput. Sci., Harvey Mudd College, Claremont, CA, USA, Tech. Rep., 2009.

[29] D. Marpe, T. Wiegand, and G. J. Sullivan, "The H.264/MPEG4 advanced video coding standard and its applications," *IEEE Commun. Mag.*, vol. 44, no. 8, pp. 134–143, Aug. 2006.

[30] H. Schwarz, D. Marpe, and T. Wiegand, "Analysis of hierarchical B pictures and MCTF," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2006, pp. 1929–1932.

[31] D. Marpe, H. Schwarz, and T. Wiegand. *Hierarchical Prediction Structures*. Accessed: Jul. 2017. [Online]. Available: https://www.hhi.fraunhofer.de/en/departments/vca/research-groups/image-video-coding/research-topics/hierarchical-prediction-structures.html

[32] D. Grois, T. Nguyen, and D. Marpe, "Performance comparison of AV1, JEM, VP9, and HEVC encoders," in *Applications of Digital Image Processing XL*, vol. 10396. International Society for Optics and Photonics, 2018, p. 103960L.

[33] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[34] W. Gao, S. Kwong, Y. Zhou, and H. Yuan, "SSIM-based game theory approach for rate-distortion optimized intra frame CTU-level bit allocation," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 988–999, Jun. 2016.

[35] Z. Liu, S. Guo, and D. Wang, "Binary classification based linear rate estimation model for HEVC RDO," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 3676–3680.

[36] Z. Liu. *X265 Ratecontrol Demo*. Accessed: Oct. 2018. [Online]. Available: https://github.com/hevccloud/Our265

[37] Z. Liu, L. Wang, and X. Li, "Rate control optimization of x265 using information from quarter-resolution pre-motion-estimation," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3623–3627.

[38] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33, Austin, TX, USA, Apr. 2001.

Zhenyu Liu (M'07) received the B.E., M.E., and Ph.D. degrees in electrical engineering from the Beijing Institute of Technology, China, in 1996, 1999, and 2002, respectively. From 2002 to 2004, he held a post-doctoral position at Tsinghua University, China, where he concentrated on the embedded processor architecture design. From 2004 to 2009, he was a Visiting Researcher with the Graduate School, IPS, Waseda University, Japan. He joined Tsinghua University in 2009, where he is currently an Associate Professor with RIIT, TNList. His research interests include signal processing, energy-efficient real-time video encoding, and application-specific processing.

Libo Wang received the B.E. and M.E. degrees in science fundamental mathematics from Shanghai Jiao Tong University, China, in 2001 and 2005, respectively. From 2004 to 2005, he was with the Intel China Software Center as an Intern to develop the high-performance AVS Codec on X86 architecture. From 2005 to 2016, he was with Actions Semiconductor Co., Ltd., where he was undertaking research on 4K H.264/H.265 Codec, 4K ISP, and 2D GPU on 28-nm application processor. He is currently a Senior Multimedia Algorithm Engineer with Alibaba China, Co., Ltd., China, and is also with Taobao, Co., Ltd. His research interests include energy-efficient real-time video Codec, image processing, and high-performance CNN inference framework.

Xiaobo Li received the B.E. degree from PLA Information Engineering University, in 2000, and the M.E. degree in computer science from the Software and Microelectronics School, Peking University, in 2009. He joined Alibaba Co., Ltd., China, in 2009, where he is currently a Staff Algorithm Engineer, where he is also leading the Multimedia Algorithm Team, Taobao Co., Ltd., China. His research interests include the high-performance H.264/H.265 Codec and computer vision artificial intelligence.

Xiangyang Ji (M'10) received the B.S. degree in materials science and the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. He joined Tsinghua University, Beijing, in 2008, where he is currently a Professor with the Department of Automation, School of Information Science and Technology. He has authored over 100 referred conference and journal papers. His current research interests include signal processing, image/video compressing, and intelligent imaging.