IET Image Processing

Review Article

High-speed implementation of rate-distortion optimised quantisation for H.265/HEVC

ISSN 1751-9659
Received on 12th March 2014
Accepted on 14th December 2014
doi: 10.1049/iet-ipr.2014.0849
www.ietdl.org

Jing He[1], Fuzheng Yang[1], Yicong Zhou[2] ✉

[1]*State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, People's Republic of China*
[2]*Department of Computer and Information Science, University of Macau, Taipa, Macau, People's Republic of China*
✉ *E-mail: yicongzhou@umac.ma*

Abstract: Rate-distortion optimised quantisation (RDOQ) is generally utilised in video coding for achieving higher coding efficiency. To determine the optimal quantised level for each transform coefficient, it requires considerable complexity in practice to calculate rate-distortion (RD) costs from multiple candidates of quantised levels. This study proposes a fast RDOQ algorithm with low computational complexity for the latest video coding standard, the high-efficiency video coding. Instead of calculating the RD costs from two candidates of quantised levels separately, their RD cost differences were derived. A bit-rate estimation method is then used to accurately calculate the length of coding bits in the RD cost function, avoiding the high-computation-cost process of context-adaptive binary arithmetic coding. Experimental results show that, with a negligible degradation of coding performance, the proposed algorithm is faster than RDOQ by 74.6% in average.

1 Introduction

H.265/high-efficiency video coding (HEVC) [1, 2] is the latest video coding standard developed by the Joint Collaborative Team on Video Coding (JCT-VC) of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). Compared with the previous video coding standard H.264/AVC [3, 4], it can be achieved a bit-rate reduction of nearly 50% but it has much higher computational complexity. This is because it utilises many new techniques such as the flexible coding structures, including the coding unit (CU), prediction unit (PU) and transform unit (TU). Therefore fast coding schemes of H.265/HEVC are required in its practical applications, especially in the real-time applications.

H.265/HEVC is a successor to the H.264/AVC standard and it still utilises the well-known hybrid video coding structure that applies quantisation to transform coefficients of the residual signal after the prediction process. Similar to H.264/AVC, H.265/HEVC only specifies the reconstruction levels and the dequantisation process, and allows the coders to design the quantiser. As a lossy compression technique, quantisation directly determines the coding distortion and influences the coding bit-rate. Therefore the choice of a quantiser greatly impacts the coding performance of the encoder. This leads to a demand of designing high-efficient quantisation schemes for video coders.

Conventionally, many coders including the H.265/HEVC test model HM [5] still utilise the uniform scalar quantiser with a fixed rounding offset for coding with low complexity. Although this kind of quantisers has low complexity and is easy to realise, it is actually not optimal because both distortion and coding bits should be considered in designing a variable-rate quantiser, and each quantised result influences the source distribution of the entropy coding process. Therefore many efforts have been devoted to the quantisation process for higher coding performance. In [6], a rate-distortion optimised quantisation (RDOQ) scheme was presented for H.263+ [7], and similar approaches were also proposed for H.264/AVC in [8]. In those methods, a trellis is constructed to model the entropy coding process. Each node of the trellis represents a quantised level of a transform coefficient, and different paths through the trellis represent different combinations of the quantised transform coefficients. Integrating the rate-distortion (RD) optimisation principle into the quantisation process, the optimal quantisation results can be achieved by searching for the path that has the minimum RD cost.

However, the searching process within a full trellis requires a very high computational complexity and thus it is difficult to implement in practice. To avoid this, a simplified RDOQ algorithm was proposed in [9] and was implemented in both the H.264/AVC test model JM [10, 11] and the H.265/HEVC test model HM [5]. Using the RD-optimised criterion, this algorithm investigates all coefficients in the reverse scanning order to determine both the position of the last non-zero coefficient and the quantised levels of other transform coefficients within a block. Although the simplified RDOQ offers higher coding efficiency than the conventional hard-decision quantisation (HDQ), it still requires high computational complexity because of the calculation of both the distortions and the coding bits for multiple candidates of the quantised levels. To solve this problem, a fast RDOQ scheme was proposed in [12] for H.264/AVC. It uses a rate model for the context adaptive variable length coding (CAVLC) to roughly estimate the coding bits in the RD cost function. In [13], a more accurate bit-rate estimation method was introduced by utilising a model to estimate the coding bits of the run-level pairs in a quantised coefficient block in the CAVLC coding process. This model was defined as 'rate-X' where X refers to a transform coefficient block.

H.265/HEVC specifies only one entropy coding method, CABAC [14], while H.264/AVC defines two. Therefore the above fast RDOQ algorithms based on CAVLC cannot be directly used in H.265/HEVC encoders. In this paper, a high-speed algorithm of RDOQ for H.265/HEVC is proposed. Firstly, a bit-rate estimation method for CABAC in H.265/HEVC is constructed by estimating the length of coding bits in the RD cost function. The optimal levels of all coefficients in a transform unit (TU) are determined based on the difference of the RD costs between two candidate levels. Experimental results show that the proposed algorithm can reduce the total quantisation time by 74.6% in average with a negligible degradation of coding efficiency.

The rest of this paper is organised as follows. The framework of RDOQ and its implementation algorithm are introduced in Section 2. Section 3 describes the proposed fast RDOQ algorithm and bit-rate estimation scheme in detail. Simulation results are provided in Section 4. Finally, this paper is concluded in Section 5.

2 Framework of RDOQ

As the quantisation process directly influences the coding distortion and bit-rate, it greatly impacts the coding efficiency. For a given TU $\boldsymbol{X}$, the optimised quantised block $\boldsymbol{Z}^*$ is obtained by minimising the coding distortion D subject to a given bit budget constraint $R \le R_c$, that is

$$\boldsymbol{Z}^* = \arg\min_{\boldsymbol{Z}^*} D(\boldsymbol{X}, \boldsymbol{Z}^*), \quad \text{s.t.} \quad R(\boldsymbol{Z}^*) \le R_c \tag{1}$$

where $D(\boldsymbol{X}, \boldsymbol{Z}^*)$ represents the coding distortion when $\boldsymbol{X}$ is quantised to $\boldsymbol{Z}^*$, and $R(\boldsymbol{Z}^*)$ indicates the number of coding bits for the quantised block $\boldsymbol{Z}^*$. In this paper, the coding distortion is measured by the sum of squared difference (SSD). Including the Lagrangian multiplier λ [15–17], the optimisation problem in (1) can be converted into the following non-constrained form

$$\boldsymbol{Z}^* = \arg\min_{\boldsymbol{Z}^*} \{J_\lambda(\boldsymbol{Z}^*)\} \tag{2}$$

where

$$J_\lambda(\boldsymbol{Z}^*) = D(\boldsymbol{X}, \boldsymbol{Z}^*) + \lambda R(\boldsymbol{Z}^*)$$

Assume that there are M coefficients in $\boldsymbol{X}$ and each coefficient contains N candidates of quantised levels, then N^M quantised blocks can be formed by different combinations of all candidates. Therefore the whole optimisation process can be converted into choosing the optimised quantised block $\boldsymbol{Z}^*$ from all N^M candidates of quantised blocks, which can be quite complex and time consuming.

The simplified RDOQ algorithm utilised by HM in [9] aims at achieving a good coding performance without searching all candidate blocks by five following steps. Instead of modifying the quantisation parameters (QPs), the RDOQ algorithm chooses the best quantised level based on the RDO criterion under a given QP.

2.1 Determine the candidates of quantised levels for each coefficient

For each DCT coefficient $c_i (i = 0, 1, \ldots, M-1)$, two available candidates of quantised levels $l_{i,\text{floor}}$ and $l_{i,\text{round}}$ can be calculated as follows

$$\begin{aligned} l_{i,\text{round}} &= \left\lfloor \frac{|c_i|}{Q_{\text{step}}} + 0.5 \right\rfloor \\ l_{i,\text{floor}} &= \begin{cases} l_{i,\text{round}} - 1, & l_{i,\text{round}} > 0 \\ 0, & l_{i,\text{round}} = 0 \end{cases} \end{aligned} \tag{3}$$

where Q_{step} refers to the quantisation step derived by the QP directly, and $\lfloor x \rfloor$ is the floor function to obtain the maximum integer no larger than x. Note that only the absolute values are considered here because the sign of a quantised level is the same as its corresponding coefficient.

2.2 Determine the optimal quantised level for each coefficient

Following the scanning order of the transform coefficient coding process, each coefficient in a TU is determined by the RD-optimised method. The optimal quantised level for each coefficient c_i is derived by minimising the following RD cost function

$$J_\lambda(l_{i,j}) = D(c_i, l_{i,j}) + \lambda R(l_{i,j}) \tag{4}$$

where $l_{i,j}$ can be either $l_{i,\text{floor}}$ or $l_{i,\text{round}}$, $D(c_i, l_{i,j})$ indicates the distortion between c_i and its reconstructed value from $l_{i,j}$ using the SSD measure, and $R(l_{i,j})$ represents the number of coding bits of $l_{i,j}$ and it is calculated using the actual entropy coding process. After determining each coefficient, the context models are updated for each syntax element related to the transform coefficients coding process. This greatly increases the data dependency and computational complexity.

2.3 Determine the position of the last significant coefficient

The last significant coefficient denotes the last non-zero coefficient of a quantised TU in the scanning order and it is determined based on the RD cost function

$$J_\lambda(l_i^{\text{last}}) = D(\boldsymbol{X}, \boldsymbol{Z}_i) + \lambda R(\boldsymbol{Z}_i) \tag{5}$$

where $\boldsymbol{Z}_i$ represents the quantised TU with the ith coefficient l_i being the last significant coefficient, indicating that all coefficients behind l_i in the scanning order is set to zero. $R(\boldsymbol{Z}_i)$ and $D(\boldsymbol{X}, \boldsymbol{Z}_i)$ are the length of coding bits of $\boldsymbol{Z}_i$ and the coding distortion for $\boldsymbol{X}$ being quantised to $\boldsymbol{Z}_i$, respectively. Based on the quantised result of step (2), the RD costs are calculated for each case of $\boldsymbol{Z}_i$ and the ith non-zero coefficient l_i being the last non-zero coefficient. Then the one with the minimum RD cost is chosen as the optimal $\boldsymbol{Z}_i$ with the ith coefficient being the optimal last significant coefficient.

2.4 Determine all-zero coefficient groups (CGs)

In the coding process of transform coefficients in H.265/HEVC, a CG is defined as a set of 16 consecutive coefficients in a scan order [18], and it corresponds to a 4×4 subblock. One symbol *coded_sub_block_flag* (CSBF) was introduced to indicate whether there are non-zero coefficients in a CG. If the current CG contains all zeros, only CSBF bins will be transmitted.

The RDOQ algorithm in HM determines whether a CG has non-zero coefficients by comparing the following two RD costs

$$\begin{aligned} J_\lambda(\boldsymbol{Z}_{\text{CG}}^0) &= D(\boldsymbol{X}_{\text{CG}}, \boldsymbol{Z}_{\text{CG}}^0) + \lambda R(\boldsymbol{Z}_{\text{CG}}^0) \\ J_\lambda(\boldsymbol{Z}_{\text{CG}}^*) &= D(\boldsymbol{X}_{\text{CG}}, \boldsymbol{Z}_{\text{CG}}^*) + \lambda R(\boldsymbol{Z}_{\text{CG}}^*) \end{aligned} \tag{6}$$

where $D(\boldsymbol{X}_{\text{CG}}, \boldsymbol{Z}_{\text{CG}}^0)$ represents the coding distortion when all coefficients in a CG are quantised to 0, $R(\boldsymbol{Z}_{\text{CG}}^0)$ is the coding bits of CSBF, and $J_\lambda(\boldsymbol{Z}_{\text{CG}}^*)$ indicates the RD cost of the current quantised CG determined by step (2). The one with the smaller RD cost will be chosen as the optimal quantised CG.

H.265/HEVC provides a technique called sign data hiding (SDH) to reduce the coding bits of sign data [19, 20]. The last non-zero coefficients in each CG can be embedded in the parity of the sum of the CG levels using a predefined rule: even (odd) corresponds to '+' ('−'). RDOQ in HM utilises SDH after the determination of all-zero CGs. When there is a mismatch between the parity and sign, the RDOQ scheme will choose one coefficient based on the RDO criterion, and increase or decrease its quantised level by one.

2.5 Determine all-zero TUs

In order to reduce the coding bit lengths of transform coefficients, the H.265/HEVC syntax includes *coded_block_flag* (CBF) that indicates whether a transform block (TB) has non-zero coefficients. Therefore after the determination of all CGs, the RDOQ scheme will decide whether to set all coefficients to 0 based on two RD costs

$$\begin{aligned} J_\lambda(\boldsymbol{Z}^0) &= D(\boldsymbol{X}, \boldsymbol{Z}^0) + \lambda R(\boldsymbol{Z}^0) \\ J_\lambda(\boldsymbol{Z}^*) &= D(\boldsymbol{X}, \boldsymbol{Z}^*) + \lambda R(\boldsymbol{Z}^*) \end{aligned} \tag{7}$$

where $D(\boldsymbol{X}, \boldsymbol{Z}^0)$ refers to the coding distortion when all coefficients are quantised to 0, $R(\boldsymbol{Z}^0)$ is the coding bits of CBF, and $J_\lambda(\boldsymbol{Z}^*)$ represents the RD cost of the current optimal quantised result $\boldsymbol{Z}^*$. The final result of RDOQ will be $\boldsymbol{Z}^*$ if $J_\lambda(\boldsymbol{Z}^*) < J_\lambda(\boldsymbol{Z}^0)$ or vice versa.

Fig. 1 illustrates the architecture of RDOQ in HM. The simplified RDOQ succeeds in avoiding searching a huge number of candidate blocks by reducing both the candidate number of quantised levels for each coefficient and the number of candidate blocks. However, it still requires calculating the coding distortions and the number of coding bits for candidates of each coefficient or each CG in a TU. This introduces a large amount of computations and it is quite time consuming. Moreover, the calculation of coding bits involves the highly sequential process of CABAC. It also contains a complicated scheme of updating context for each coefficient, making it difficult for parallel processing in hardware implementations. Therefore it is essential to design a simple and fast algorithm for RDOQ to be practically used in H.265/HEVC encoders.

3 Proposed fast implementation algorithm of RDOQ

Similar to the framework of RDOQ in HM, the proposed fast RDOQ method is implemented in three main steps. The first step determines the optimal levels for each coefficient in a transform coefficient block and the second step focuses on the optimal position of the last non-zero coefficient. Finally, whether the coefficients in a CG or TU should all be quantised to zero is determined in the third step. To accelerate the algorithm speed, rather than calculating the entire RD costs for all candidates, we calculate and estimate only the differences of coding distortions and coding bits between two candidate levels, respectively. Furthermore, a fast bit-rate estimation model for HEVC–CABAC is proposed to evaluate the differences of coding bits, avoiding the complicated entropy coding process. The following subsections describe the algorithm in detail.

3.1 Determination of the optimal quantised level

It can be seen from the above description of RDOQ that the optimal quantised level l_i^* for each coefficient c_i depends mainly on the comparison of two RD costs $J_\lambda(l_{i,\text{floor}})$ and $J_\lambda(l_{i,\text{round}})$. To accelerate the comparison process, ΔJ_i is introduced to measure the difference between $J_\lambda(l_{i,\text{floor}})$ and $J_\lambda(l_{i,\text{round}})$ as follows

$$\Delta J_i = J_\lambda(l_{i,\text{floor}}) - J_\lambda(l_{i,\text{round}}) = \Delta D_i - \lambda \Delta R_i \tag{8}$$

where ΔD_i and ΔR_i represent the difference of the quantisation errors and the length of coding bits, respectively. λ here is the same as that in HM-RDOQ. The optimal quantised level of c_i is $l_{i,\text{floor}}$ if ΔJ_i is less than zero; otherwise, it is $l_{i,\text{round}}$. Therefore the optimal quantised level can be determined based on the value of ΔJ_i, which is related to the values of ΔD_i and ΔR_i.

According to the definitions of $l_{i,\text{floor}}$ and $l_{i,\text{round}}$ in (3), the difference of the quantisation errors between $l_{i,\text{floor}}$ and $l_{i,\text{round}}$ can be directly obtained as follows

$$\begin{aligned}\Delta D_i &= D(c_i, l_{i,\text{floor}}) - D(c_i, l_{i,\text{round}}) \\ &= \left[(l_{i,\text{float}} - l_{i,\text{floor}})^2 - (l_{i,\text{float}} - l_{i,\text{round}})^2\right] \times Q_{\text{step}}^2 \\ &= \left[2(l_{i,\text{float}} - l_{i,\text{floor}}) - 1\right] \times Q_{\text{step}}^2\end{aligned} \tag{9}$$

where Q_{step} refers to the quantisation step and $l_{i,\text{float}}$ denotes the value of c_i divided by Q_{step} without rounding, which is a float value. It should be noted that the coding distortions in (9) calculated in the transform domain can also represent the coding distortions in the spatial domain, because the discrete cosine transform (DCT) scheme is orthogonal [15].

On the other hand, the difference of the lengths of coding bits between $l_{i,\text{floor}}$ and $l_{i,\text{round}}$ is expressed as follows

$$\Delta R_i = R(l_{i,\text{round}}) - R(l_{i,\text{floor}}) \tag{10}$$

Compared with ΔD_i, the calculation of ΔR_i is far more complicated because of the CABAC scheme in H.265/HEVC. Thus, a method for estimating ΔR_i without using CABAC is put forward to accelerate the calculation of ΔR_i.

In the HEVC–CABAC scheme, there are five syntax elements related to the transform coefficient coding process, which are given as follows:

(1) *significant_coeff_flag:* indicating whether a coefficient is non-zero.
(2) *coeff_abs_level_greater1_flag:* indicating whether the absolute value of a coefficient is larger than 1.
(3) *coeff_abs_level_greater2_flag:* indicating whether the absolute value of a coefficient is larger than 2.
(4) *coeff_sign_flag:* indicating the sign of a non-zero coefficient.
(5) *coeff_abs_level_remaining:* indicating the remaining level of a coefficient.

Five symbols R_{sig}, R_{gt1}, R_{gt2}, R_{sgn} and R_{rem} are introduced to represent the number of coding bits related to the above five syntax elements, respectively. Thus ΔR_i can be expressed in the following form

$$\Delta R_i = \begin{cases} R_{\text{sig}}^{(1)} + R_{\text{gt1}}^{(0)} + R_{\text{sgn}} - R_{\text{sig}}^{(0)} & l_{i,\text{floor}} = 0 \\ R_{\text{gt1}}^{(1)} + R_{\text{gt2}}^{(0)} - R_{\text{gt1}}^{(0)} & l_{i,\text{floor}} = 1 \\ R_{\text{gt2}}^{(1)} + R_{\text{rem}}^{(3)} - R_{\text{gt2}}^{(0)} & l_{i,\text{floor}} = 2 \\ R_{\text{rem}}^{(l)} - R_{\text{rem}}^{(l-1)} & \text{otherwise} \end{cases} \tag{11}$$

where symbols $R_{\text{sig}}^{(0)}$, $R_{\text{sig}}^{(1)}$, $R_{\text{gt1}}^{(0)}$, $R_{\text{gt1}}^{(1)}$, $R_{\text{gt2}}^{(0)}$ and $R_{\text{gt2}}^{(1)}$ represent the estimated coding bits when the syntax element *significant_coeff_flag*, *coeff_abs_level_greater1_flag* and *coeff_abs_level_greater2_flag* is '0' or '1', respectively, and $R_{\text{rem}}^{(l)}$ refers to the estimated bits of *coeff_abs_level_remaining* when the coefficient level is equal to l. The detail estimation method is

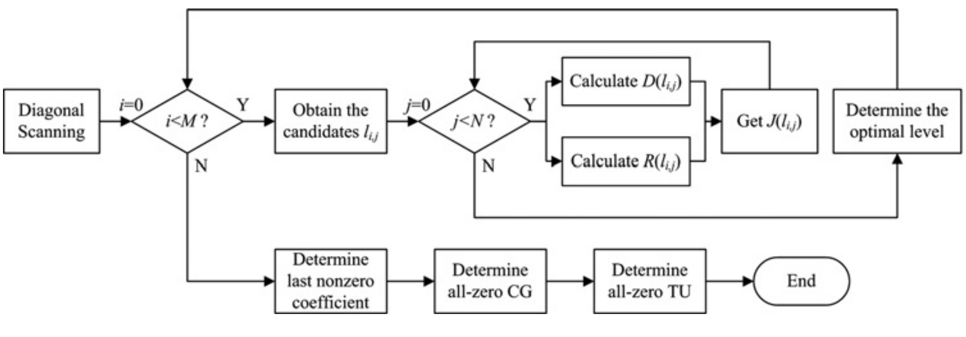

Fig. 1 *Framework of RDOQ in HM*

introduced in Section 3.4. Note that the syntax element *coeff_abs_level_greater2_flag* exists only for the first coefficient in a CG with magnitude greater than 1 in the reverse scanning order, and *coeff_abs_level_greater1_flag* only exists for the first eight coefficients in a CG in the reverse scanning order when there are more than eight coefficients with magnitudes greater than 1. Thus, when the syntax element *coeff_abs_level_greater2_flag* is not existed, ΔR_i is calculated using (12). Furthermore, the formula of ΔR_i changes into (13) if *coeff_abs_level_greater1_flag* is not existed, either.

$$\Delta R_i = \begin{cases} R_{\text{sig}}^{(1)} + R_{\text{gt1}}^{(0)} + R_{\text{sgn}} - R_{\text{sig}}^{(0)} & l_{i,\text{floor}} = 0 \\ R_{\text{gt1}}^{(1)} + R_{\text{rem}}^{(2)} - R_{\text{gt1}}^{(0)} & l_{i,\text{floor}} = 1 \\ R_{\text{rem}}^{(l)} - R_{\text{rem}}^{(l-1)} & \text{otherwise} \end{cases} \quad (12)$$

$$\Delta R_i = \begin{cases} R_{\text{sig}}^{(1)} + R_{\text{sgn}} - R_{\text{sig}}^{(0)} & l_{i,\text{floor}} = 0 \\ R_{\text{rem}}^{(l)} - R_{\text{rem}}^{(l-1)} & \text{otherwise} \end{cases} \quad (13)$$

3.2 Determination of the last non-zero coefficient

Four syntax elements are related to the position of the last non-zero coefficient: *last_significant_coeff_x_prefix*, *last_significant_coeff_x_suffix*, *last_significant_coeff_y_prefix* and *last_significant_coeff_y_suffix*. The first two elements indicate the column number of the last non-zero coefficient in a TU, while the other two indicate the row number of the last non-zero coefficient. Thus two symbols $R_{\text{las}X}$ and $R_{\text{las}Y}$ are introduced to represent the coding bits of the location of the last non-zero coefficient.

It can be seen from Section 2 that the optimal position of the last significant coefficient is determined by the RD cost function in (5). Assume that l_i^* and l_j^* $(0 \le i < j \le M-1)$ are two successive non-zero quantised levels in the current TU (all levels between l_i^* and l_j^* are zero), the difference between $J_\lambda(l_i^{\text{last}})$ and $J_\lambda(l_j^{\text{last}})$ can be expressed as

$$\Delta J_{i,j}^{\text{last}} = J_\lambda(l_i^{\text{last}}) - J_\lambda(l_j^{\text{last}}) = \Delta D_{i,j}^{\text{last}} - \lambda \Delta R_{i,j}^{\text{last}} \quad (14)$$

where $\Delta D_{i,j}^{\text{last}}$ indicates the difference of quantisation errors between $\boldsymbol{Z}_i$ and $\boldsymbol{Z}_j$ which can be derived as follows

$$\Delta D_{i,j}^{\text{last}} = D(\boldsymbol{X}, \boldsymbol{Z}_i) - D(\boldsymbol{X}, \boldsymbol{Z}_j) = D(c_j, 0) - D(c_j, l_j^*) \quad (15)$$

According to (15), it can be seen that $\Delta D_{i,j}^{\text{last}}$ exactly equals to the difference of the quantisation errors of c_j between two quantised levels 0 and l_j^*. Thus it can be calculated directly using the same formula in (9). On the other hand, according to the principle of the transformed coefficient coding in H.265/HEVC [18], $\Delta R_{i,j}^{\text{last}}$, which refers to the difference of coding bits between $\boldsymbol{Z}_j$ and $\boldsymbol{Z}_i$, can be expressed in the following form in (16)

$$\begin{aligned} \Delta R_{i,j}^{\text{last}} &= R(\boldsymbol{Z}_j) - R(\boldsymbol{Z}_i) \\ &= \sum_{k=i+1}^{j-1} R_{\text{sig},k}^{(0)} + R_{\text{gt1},j} + R_{\text{gt2},j} + R_{\text{sgn},j} + R_{\text{rem},j} \\ &\quad + R_{\text{las}X,j} + R_{\text{las}Y,j} - R_{\text{las}X,i} - R_{\text{las}Y,i} \end{aligned} \quad (16)$$

With the above discussions, all differences related to the RD cost between each pair of successive non-zero levels can be obtained by (14)–(16). Therefore the one with the minimum RD cost is chosen to be the last non-zero coefficient of the current TU.

3.3 Determination of the all-zero CGs and all-zero TUs

In the H.265/HEVC entropy coding process, a TU is divided into 4×4 subblocks (i.e. CGs), each CG is corresponding to a syntax element CSBF that indicates whether a CG has non-zero coefficients. Similarly, each TU also corresponds to a syntax element CBF to represent whether a TU has non-zero coefficient. Therefore two symbols R_{CG} and R_{CBF} are introduced to represent the coding bits of these two syntax elements.

It can be seen from Section 2 that whether a CG should be quantised to all zeros depends on two RD costs in (6), thus the difference between these two RD costs can be expressed as

$$\Delta J_{\text{CG}} = J_\lambda(\boldsymbol{Z}_{\text{CG}}^0) - J_\lambda(\boldsymbol{Z}_{\text{CG}}^*) = \Delta D_{\text{CG}} - \lambda \Delta R_{\text{CG}} \quad (17)$$

where ΔD_{CG} and ΔR_{CG} represent the differences of the quantisation errors and coding bits between $\boldsymbol{Z}_{\text{CG}}^0$ and $\boldsymbol{Z}_{\text{CG}}^*$, respectively. Assume that $l_1, \ldots, l_L$ are L $(0 < L \le 16)$ non-zero coefficients in a non-zero CG, then ΔD_{CG} and ΔR_{CG} can be derived as follows

$$\begin{aligned} \Delta D_{\text{CG}} &= D(\boldsymbol{X}_{\text{CG}}, \boldsymbol{Z}_{\text{CG}}^0) - D(\boldsymbol{X}_{\text{CG}}, \boldsymbol{Z}_{\text{CG}}^*) \\ &= \sum_{i=1}^{L} \left[D(c_i, 0) - D(c_i, l_i^*) \right] \end{aligned} \quad (18)$$

$$\Delta R_{\text{CG}} = R(\boldsymbol{Z}_{\text{CG}}^*) - R(\boldsymbol{Z}_{\text{CG}}^0) = R_{\text{CG}}^{(1)} - R_{\text{CG}}^{(0)} + \sum_{j=1}^{16} R(l_j^*) \quad (19)$$

According to (18) and (19), it can be seen that ΔD_{CG} equals to the sum of the L differences of quantisation errors that can be calculated directly using (9), and ΔR_{CG} can be divided into two parts: the sum of coding bits of all 16 coefficients in the current CG that can be estimated using the same method in Section 3.1, and the difference of coding bits for CSBF that can be calculated during the entropy coding process. After ΔD_{CG} and ΔR_{CG} calculated, the optimal quantised result of the current CG can be determined based on the sign of ΔJ_{CG}.

The SDH technique is utilised in the determination process of all-zero CGs and it is the same as HM does when RDOQ is not used [20]. For each level in a CG, the difference between the original coefficient and its dequantised value is calculated. To match the sign hiding rules, the value of a level with the largest difference will be increased by one or decreased by one depending on the sign of the difference.

After determining all CGs, we will check the all-zero TU based on two RD costs in (7). This is the same as the original RDOQ in HM does. In order to accelerate this process, we calculate the sum of absolute value of all quantised coefficients in a TU (l_{SUM}), and use it to estimate whether the TU is likely to be quantised to an all-zero TU. Table 1 shows probabilities of TUs that are quantised to all-zero TUs under the conditions of $l_{\text{SUM}} = 1$, $l_{\text{SUM}} = 2$ and $l_{\text{SUM}} > 2$, respectively. It can be seen from Table 1 that a TU can hardly be quantised to an all-zero TU when l_{SUM} is greater than 2. This is because of large coding distortion acquiring and limited bit-rate saving. Therefore in the proposed fast RDOQ algorithm, all-zero TU is checked only if l_{SUM} is greater than 2, otherwise this step is omitted.

Table 1 Probabilities of TUs that are quantised to all-zero TUs under the given conditions

Sequences	$l_{\text{SUM}} = 1$	$l_{\text{SUM}} = 2$	$l_{\text{SUM}} > 2$
PeopleOnStreet	0.4366	0.1762	0.0101
BQTerrace	0.4239	0.1712	0.0098
Kimono	0.4865	0.1962	0.0114
BasketballDrill	0.4553	0.1837	0.0106
PartyScene	0.2771	0.1125	0.0062
BlowingBubbles	0.3772	0.1525	0.0087
RaceHorses	0.3402	0.1377	0.0077
vidyo1	0.5053	0.2037	0.0119
ChinaSpeed	0.5288	0.1731	0.0099
SlideEditing	0.4423	0.1785	0.0103
average	0.4273	0.1685	0.0097

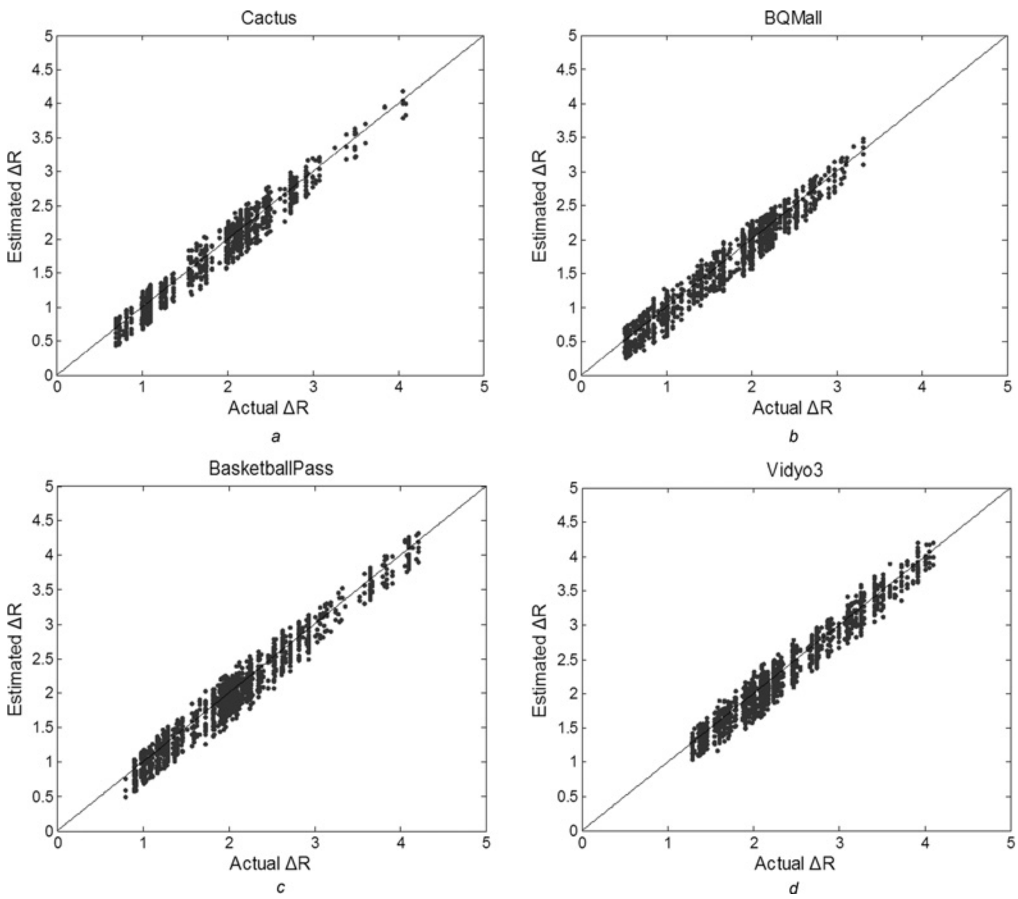

Fig. 2 *Comparison results of the actual and estimated ΔR*

a Cactus
b BQ Mall
c BasketballPass
d Vidyo3

Table 2 Performance of the HM-RDOQ and the proposed RDOQ compared with HDQ

Sequences		Y BD-rate, %					
		All intra (AI)		Random access (RA)		Low delay (LD)	
		HM-RDOQ against HDQ	Proposed RDOQ against HDQ	HM-RDOQ against HDQ	Proposed RDOQ against HDQ	HM-RDOQ against HDQ	Proposed RDOQ against HDQ
class A	traffic	−3.61	−2.76	−3.62	−2.84		
	PeopleOnStreet	−4.58	−4.11	−3.00	−3.23		
	average	−4.09	−3.44	−3.31	−3.04		
class B	BasketballDrive	−6.23	−4.65	−6.65	−4.83	−6.38	−4.82
	BQTerrace	−4.83	−3.72	−3.81	−2.74	−2.72	−1.84
	Cactus	−5.05	−4.60	−5.74	−4.89	−4.67	−3.55
	Kimono	−6.54	−5.88	−5.69	−4.73	−4.54	−3.61
	ParkScene	−4.84	−4.14	−4.32	−3.96	−2.63	−2.67
	average	−5.49	−4.60	−5.24	−4.23	−4.18	−3.30
class C	BasketballDrill	−3.67	−3.20	−3.91	−4.19	−4.68	−3.80
	BQMall	−3.90	−3.02	−4.75	−4.18	−4.16	−3.42
	PartyScene	−3.29	−2.83	−4.77	−4.44	−4.19	−4.43
	RaceHorses	−3.89	−3.20	−4.43	−3.74	−3.49	−2.61
	average	−3.68	−3.06	−4.46	−4.14	−4.13	−3.57
class D	BasketballPass	−3.95	−3.21	−3.88	−4.05	−2.81	−3.06
	BlowingBubbles	−3.31	−2.90	−4.66	−4.55	−4.30	−4.14
	BQSquare	−2.90	−2.49	−3.63	−3.88	−3.21	−3.23
	RaceHorses	−3.44	−3.03	−3.79	−3.49	−3.14	−2.74
	average	−3.40	−2.91	−4.00	−3.99	−3.36	−3.29
class E	Vidyo1	−3.12	−2.24			−1.76	−2.08
	Vidyo3	−3.79	−3.01			−1.42	−0.99
	Vidyo4	−3.69	−2.60			−2.87	−1.93
	average	−3.53	−2.62			−2.01	−1.67
class F	BasketballDrillText	−3.53	−3.02	−4.09	−4.87	−2.97	−3.82
	ChinaSpeed	−3.18	−2.46	−4.29	−3.29	−3.83	−3.19
	SlideEditing	−3.46	−2.88	−3.95	−3.22	−3.23	−2.83
	SlideShow	−3.25	−2.80	−6.43	−5.60	−5.93	−4.57
	average	−3.35	−2.79	−4.69	−4.25	−3.99	−3.60
average for all		−3.99	−3.31	−4.49	−4.04	−3.64	−3.17

Table 3 Performance of the proposed RDOQ compared with the HM-RDOQ

Sequences		Y BD-rate, %		
		All intra (AI)	Random access (RA)	Low delay (LD)
class A	Traffic	0.89	0.81	
	PeopleOnStreet	0.50	−0.24	
	average	0.69	0.28	
class B	BasketballDrive	1.68	1.98	1.70
	BQTerrace	1.14	1.14	0.95
	Cactus	0.50	0.95	1.18
	Kimono	0.77	1.05	0.99
	ParkScene	0.74	0.39	−0.04
	average	0.96	1.10	0.95
class C	BasketballDrill	0.49	−0.29	0.50
	BQMall	0.92	0.60	0.77
	PartyScene	0.48	0.36	−0.23
	RaceHorses	0.72	0.72	0.92
	average	0.65	0.35	0.49
class D	BasketballPass	0.75	−0.17	−0.24
	BlowingBubbles	0.42	0.14	0.18
	BQSquare	0.43	−0.23	−0.01
	RaceHorses	0.41	0.30	0.40
	AVERAGE	0.50	0.01	0.08
class E	Vidyo1	0.92		−0.32
	Vidyo3	0.81		0.45
	Vidyo4	1.14		0.97
	average	0.95		0.37
class F	BasketballDrillText	0.53	−0.81	−0.87
	ChinaSpeed	0.74	1.05	0.67
	SlideEditing	0.60	0.77	0.45
	SlideShow	0.47	0.88	1.42
	average	0.58	0.47	0.42
average for all		0.73	0.49	0.49

3.4 Fast bit-rate estimation for CABAC in H.265/HEVC

In CABAC in H.265/HEVC, non-binary elements are firstly binarised into a series of bins and different context models are assigned to each syntax element for estimating the probability. Finally, each bin and the corresponding context model are put into an arithmetic coding engine to generate the bit stream.

Arithmetic coding can be done using an estimated probability (regular mode), or equal probability of 0.5 (bypass mode). For the bypass coding mode, the length of coding bits can be directly derived after the binarisation process. On the other hand, when the regular coding mode is used, the length of coding bits depends mainly on the source probability distribution. The average length of coding bits for one symbol is short if the probability of which is closed to one and vice versa. Thus the length of coding bits can be estimated based on the probabilities of symbols '0' and '1' for each syntax element, and it is formulated as

$$\begin{aligned} R^{(1)} &= -\log_2 P^{(1)} \\ R^{(0)} &= -\log_2 (1 - P^{(1)}) \end{aligned} \tag{20}$$

where $P^{(1)}$ and $R^{(1)}$ denote the probability and length of coding bits of symbol '1', respectively. The calculations of $P^{(1)}$ for each syntax element are described as follows.

(1) $P^{(1)}_{\text{sig}}$: the proportions of non-zero coefficients for each position in a TU in all coded CUs of the current slice.
(2) $P^{(1)}_{\text{gt1}}$: the ratios of all coefficients that are greater than 1 and all non-zero coefficients for each position in a TU in all coded CUs of the current slice.
(3) $P^{(1)}_{\text{gt2}}$: the ratios of all coefficients that are greater than 2 and all coefficients that are greater than 1 for each position in a TU in all coded CUs of the current slice.

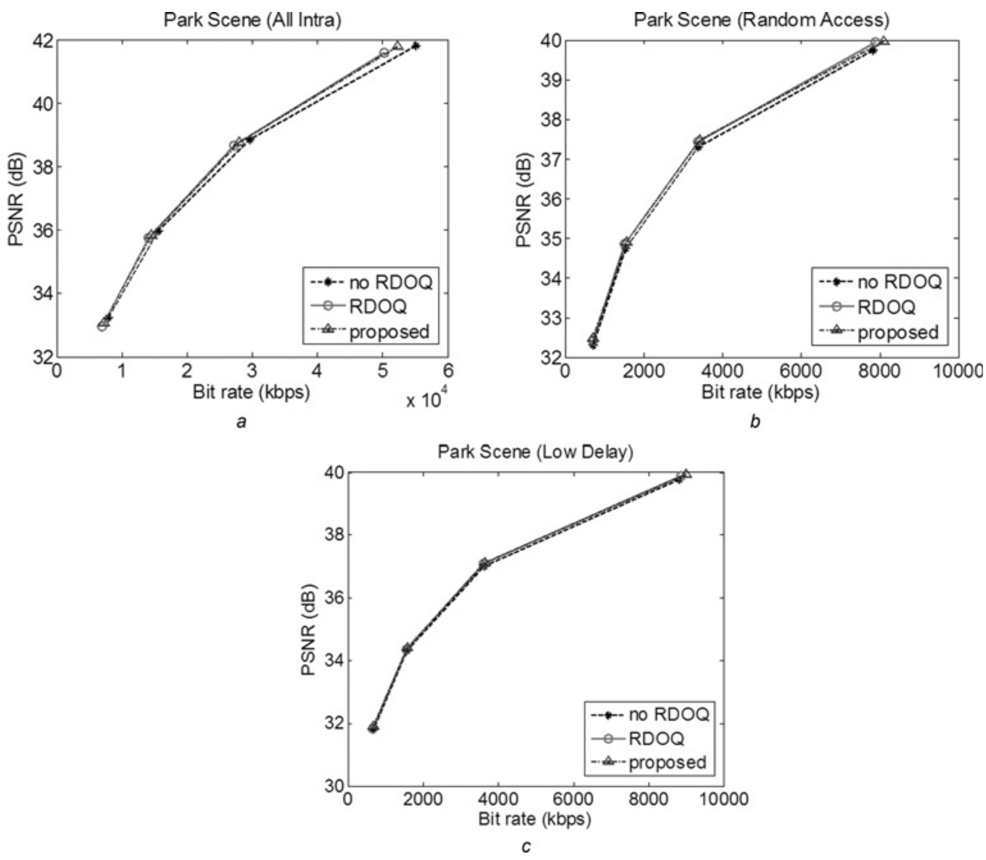

Fig. 3 *RD performance comparison for Park Scene*

a All intra
b Random access
c Low delay

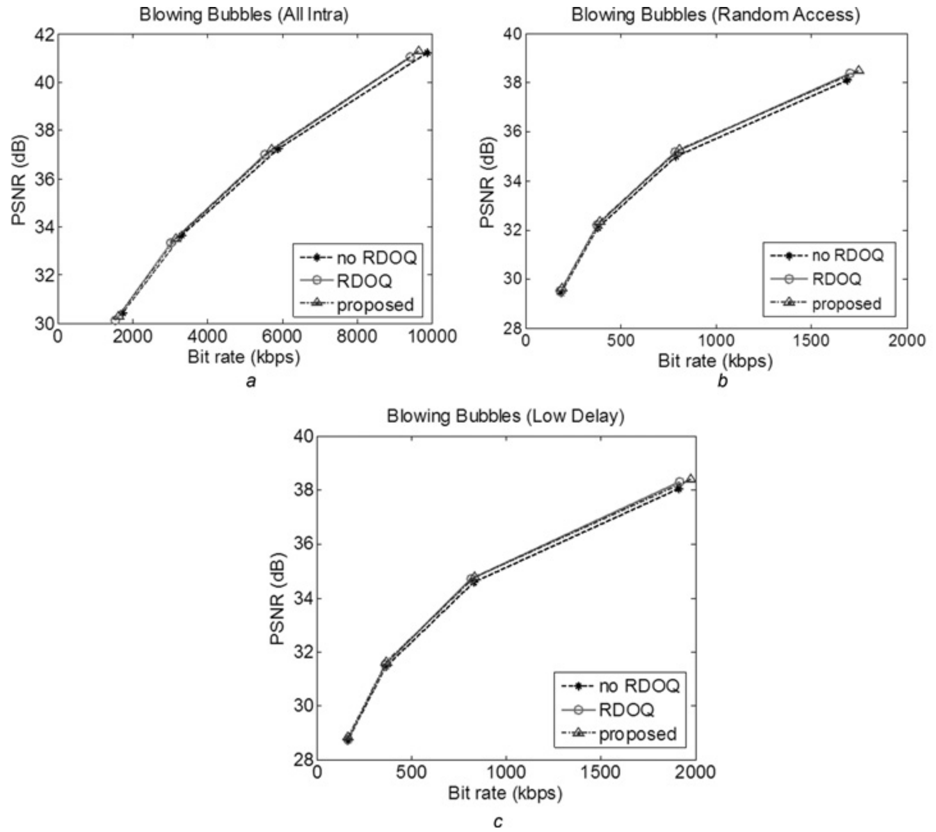

Fig. 4 *RD performance comparison for Blowing Bubbles*

a All intra
b Random access
c Low delay

(4) $P^{(1)}_{\mathrm{las}X}$ and $P^{(1)}_{\mathrm{las}Y}$: the proportion of symbol '1' in the unary prefix code after the binarisation process of the positions of all the last non-zero coefficients in the coded CUs of the current slice.

With all distribution probabilities calculated, the difference of coding bits ΔR in (11), (12), (16) and (19) can be estimated without utilising the CABAC process. Note that the distribution

Table 4 Quantisation time comparison with respect to the conventional HDQ

Sequences		All intra (AI)		Random access (RA)		Low delay (LD)	
		ΔT_{RDOQ}, %	$\Delta T_{\mathrm{proposed}}$, %	ΔT_{RDOQ}, %	$\Delta T_{\mathrm{proposed}}$, %	ΔT_{RDOQ}, %	$\Delta T_{\mathrm{proposed}}$, %
class A	Traffic	793.4	143.4	441.9	133.3		
	PeopleOnStreet	770.9	140.6	386.2	131.0		
	average	782.2	142.0	414.1	132.1		
class B	BasketballDrive	772.2	143.8	427.9	133.7	468.5	133.6
	BQTerrace	1012.5	159.0	570.7	144.4	642.3	145.1
	Cactus	904.3	153.8	482.9	137.0	568.9	141.5
	Kimono	673.6	142.9	361.7	125.7	392.8	133.4
	ParkScene	833.4	149.9	424.6	136.0	488.5	135.6
	average	839.2	149.9	453.6	135.3	512.2	137.8
class C	BasketballDrill	951.7	154.3	459.2	134.4	487.7	136.4
	BQMall	953.8	157.0	449.3	136.1	511.4	137.2
	PartyScene	1242.6	170.0	667.0	151.4	718.0	158.5
	RaceHorses	951.3	156.7	563.2	144.0	557.4	140.4
	average	1024.9	159.5	534.7	141.5	568.6	143.1
class D	BasketballPass	891.5	145.6	448.7	140.5	453.2	134.6
	BlowingBubbles	1138.4	164.0	556.2	138.5	595.9	143.9
	BQSquare	1210.6	161.7	558.9	146.4	597.6	140.6
	RaceHorses	970	157.7	585.1	143.5	604.7	146.0
	average	1052.6	157.2	537.2	142.2	562.9	141.3
class E	Vidyo1	656.9	143.0			354.2	134.9
	Vidyo3	676.7	142.8			364.3	132.5
	Vidyo4	668.3	146.6			327.2	128.2
	average	667.3	144.1			348.6	131.9
class F	BasketballDrillText	924.4	149.6	443.5	131.6	484.1	136.5
	ChinaSpeed	860.3	159.6	427.3	133.6	455.5	134.1
	SlideEditing	1086.2	165.5	354.1	126.4	376.7	127.8
	SlideShow	606.2	144.7	370.4	131.1	375.5	133.1
	AVERAGE	869.3	154.8	398.8	130.7	423.0	132.9
average for all		888.6	152.4	472.6	136.8	491.2	137.7

IET Image Process., 2015, Vol. 9, Iss. 8, pp. 652–661

probabilities of all syntax elements are updated after the encoding process of each CU for the purpose that highly accurate estimated results can be obtained by the updating process. Fig. 2 shows the comparison results of the actual and estimated ΔR in (11), (12) and (16) for four sequences: Cactus, BQMall, BasketballPass and Vidyo3. The correlation coefficients are 0.9660, 0.9611, 0.9709 and 0.9651, respectively. These demonstrate the accuracy of the proposed bit-rate estimation method. Therefore the Lagrangian multiplier λ used by HM-RDOQ also works for the proposed algorithm in (8), (14) and (17) because the actual and estimated rates are in the same range.

3.5 Proposed algorithm of RDOQ

In summary, the proposed high-speed RDOQ algorithm can be implemented as the following steps.

Step 1: Update the distribution probabilities for all syntax elements.
Step 2: For each coefficient c_i in the current TU, do the following steps.

Step 2.1: Obtain $l_{i,\text{ round}}$ and $l_{i,\text{ floor}}$ according to (3).
Step 2.2: Calculate ΔD_i directly using (9).
Step 2.3: Estimate ΔR_i according to (11) and (12).
Step 2.4: Determine the optimal level l_i^* for c_i based on the value of ΔJ_i in (8).

Step 3: Determine the position of the last non-zero coefficient for the current TU based on (14)–(16).
Step 4: For each CG in the current TU, do the following steps.

Step 4.1: Determine whether the coefficients in the current CG should be set all to zero using (17)–(19).
Step 4.2: Utilise the SDH technique to the current CG.

Step 5: Calculate l_{SUM} and determine whether the coefficients in the current TU should be set to all zero according to (7).

The proposed algorithm can effectively improve the speed of the RDOQ process by utilising the fast bit-rate estimation scheme. Furthermore, the process of the proposed RDOQ is well adapted to the parallel structure that is suitable for hardware implementations. This is because the difference of the RD costs ΔJ can be obtained independently for each coefficient.

4 Experimental results

Extensive experiments have been conducted to verify the efficiency and complexity of the proposed fast RDOQ algorithm. The proposed algorithm is implemented based on the H.265/HEVC test model HM-11.0 [5]. The experiments are conducted under the condition of the H.265/HEVC main profile defined in [21]. Six classes of 22 test sequences representing different cases and video characteristics are used. Three different GOP structures are employed. They are all intra (AI), random access (RA) and low delay (LD). The QPs are set to 22, 27, 32 and 37, respectively. There are two simulations designed: (1) the coding efficiency comparison between HM-RDOQ and the proposed RDOQ, (2) the complexity comparison of HM-RDOQ and the proposed RDOQ with respect to the traditional HDQ.

4.1 Coding efficiency evaluation

The coding efficiency results are presented as the percentage of the bit-rate savings (BD-rate) as proposed in [22] where negative numbers of BD-rate indicate performance gains and positive numbers of BD-rate indicate performance losses. Table 2 shows the results of HM-RDOQ and the proposed RDOQ compared with the conventional HDQ. It can be seen from Table 2 that the original RDOQ in HM achieves an average bit-rate reduction of 3.99, 4.49 and 3.64% for the all intra, RA and LD configurations, respectively. When using the proposed RDOQ algorithm, the results for three configurations change to 3.31, 4.04 and 3.17%, respectively. Thus the proposed algorithm achieves 87% of the coding gains achieved by HM-RDOQ, compared with HDQ.

Table 3 shows the performance of the proposed RDOQ algorithm compared with HM-RDOQ. It can be observed from Table 3 that compared with the RDOQ in HM encoder, the bit-rate increment of the proposed method is 0.73% in average for the AI configuration, and the coding efficiency loses less than 0.5% for the RA and LD configuration. Figs. 3 and 4 show the RD curves of the conventional HDQ, the original RDOQ in HM and the proposed RDOQ for the sequences ParkScene (1080 p, 24 fps) and BlowingBubbles (WQVGA, 50 fps) under the configurations of AI, RA and LD, respectively. It can be seen that the proposed RDOQ algorithm achieves almost the same coding efficiency with HM-RDOQ. Compared with the conventional HDQ, large performance gains can be obtained when using the proposed method.

4.2 Coding complexity comparison

The complexity of HM-RDOQ and the proposed algorithm are measured in terms of the relative quantisation time compared to the conventional HDQ. All results of the quantisation time are calculated as an average of four different QPs (22, 27, 32 and 37). Each sequence is encoded five times and the average time is used in order to get more accurate results. All experiments are conducted with three configurations of AI, RA and LD. The metrics of the complexity comparison is as follows

$$\Delta T_{\text{RDOQ}} = \frac{T_{\text{RDOQ}}}{T_{\text{HDQ}}} \times 100\%$$
$$\Delta T_{\text{proposed}} = \frac{T_{\text{proposed}}}{T_{\text{HDQ}}} \times 100\% \qquad (21)$$
$$\Delta T_{\text{reduction}} = \frac{T_{\text{RDOQ}} - T_{\text{proposed}}}{T_{\text{RDOQ}}} \times 100\%$$

Table 5 Quantisation time reduction of the proposed RDOQ relative to the HM-RDOQ

Sequences		$\Delta T_{\text{reduction}}$, %		
		All intra (AI)	Random access (RA)	Low delay (LD)
class A	Traffic	81.9	69.8	
	PeopleOnStreet	81.8	66.1	
	average	81.9	68.0	
class B	BasketballDrive	81.4	68.8	71.5
	BQTerrace	84.3	74.7	77.4
	Cactus	83.0	71.6	75.1
	Kimono	78.8	65.2	66.0
	ParkScene	82.0	68.0	72.2
	average	81.9	69.7	72.5
class C	BasketballDrill	83.8	70.7	72.0
	BQMall	83.5	69.7	73.2
	PartyScene	86.3	77.3	77.9
	RaceHorses	83.5	74.4	74.8
	average	84.3	73.0	74.5
class D	BasketballPass	83.7	68.7	70.3
	BlowingBubbles	85.6	75.1	75.9
	BQSquare	86.6	73.8	76.5
	RaceHorses	83.7	75.5	75.9
	average	84.9	73.3	74.6
class E	Vidyo1	78.2		61.9
	Vidyo3	78.9		63.6
	Vidyo4	78.1		60.8
	average	78.4		62.1
class F	BasketballDrillText	83.8	70.3	71.8
	ChinaSpeed	81.4	68.7	70.6
	SlideEditing	84.8	64.3	66.1
	SlideShow	76.1	64.6	64.6
	average	81.5	67.0	68.2
average for all		82.4	70.5	71.1

Table 6 Encoding time comparison with respect to HM using conventional HDQ

Sequences		All intra (AI)		Random access (RA)		Low delay (LD)	
		ΔT_1, %	ΔT_2, %	ΔT_1, %	ΔT_2, %	ΔT_1, %	ΔT_2, %
class A	traffic	133.9	102.1	110.1	101.0		
	PeopleOnStreet	133.5	102.1	108.4	100.9		
	average	133.7	102.1	109.3	101.0		
class B	BasketballDrive	131.4	102.0	109.1	100.9	107.5	100.7
	BQTerrace	142.0	102.7	112.6	101.2	110.8	100.9
	Cactus	137.5	102.5	111.2	101.1	109.8	100.9
	Kimono	127.9	102.1	107.6	100.7	105.9	100.7
	ParkScene	136.3	102.5	108.9	101.0	108.0	100.7
	average	135.0	102.4	109.9	101.0	108.4	100.8
class C	BasketballDrill	139.6	102.5	111.3	101.1	110.3	101.1
	BQMall	140.9	102.7	111.5	101.2	110.7	101.0
	PartyScene	151.6	103.2	114.7	101.3	113.1	101.2
	RaceHorses	140.7	102.7	112.2	101.2	110.2	100.9
	average	143.2	102.8	112.4	101.2	111.1	101.1
class D	BasketballPass	136.8	102.2	110.7	101.3	109.1	100.9
	BlowingBubbles	146.4	102.9	113.3	101.1	111.8	101.1
	BQSquare	151.7	103.4	112.8	101.3	112.1	101.0
	RaceHorses	143.3	102.9	113.1	101.2	111.1	101.0
	average	144.6	102.9	112.5	101.2	111.0	101.0
class E	Vidyo1	127.9	102.2			105.0	100.7
	Vidyo3	128.7	102.1			105.6	100.7
	Vidyo4	128.4	102.3			105.9	100.7
	average	128.3	102.2			105.5	100.7
class F	BasketballDrillText	140.1	102.4	111.5	101.1	110.5	101.0
	ChinaSpeed	137.9	103.0	111.0	101.1	109.2	100.9
	SlideEditing	147.5	103.2	106.3	100.7	104.9	100.5
	SlideShow	127.2	102.4	106.5	100.7	105.4	100.6
	average	138.2	102.8	108.8	100.9	107.5	100.8
average for all		137.8	102.6	110.7	101.1	108.8	100.9

where T_{HDQ}, T_{RDOQ} and T_{proposed} are the quantisation time of the conventional HDQ, HM-RDOQ and the proposed RDOQ, respectively. Tables 4 and 5 summarise the results of relative quantisation time of all sequences. Simulation results show that the RDOQ method in HM is quite time consuming compared with HDQ. The quantisation time of HM-RDOQ is nearly four times more than that of HDQ for the RA and LD configurations. The proposed RDOQ algorithm greatly reduces the quantisation time. In average, it only takes 17.6, 29.5 and 28.9% of the quantisation time of HM-RDOQ for the AI, RA and LD configurations, respectively.

Table 6 shows the results of the complexity comparison of the whole encoder using the following metrics

$$\Delta T_1 = \frac{T_{\text{RDOQ}}^{\text{all}}}{T_{\text{HDQ}}^{\text{all}}} \times 100\% \qquad \Delta T_2 = \frac{T_{\text{proposed}}^{\text{all}}}{T_{\text{HDQ}}^{\text{all}}} \times 100\% \tag{22}$$

where $T_{\text{HDQ}}^{\text{all}}$, $T_{\text{RDOQ}}^{\text{all}}$ and $T_{\text{proposed}}^{\text{all}}$ denote the whole encoding time of HM using HDQ, RDOQ and the proposed algorithm, respectively. It can be seen from the table that compared with HM-RDOQ, the proposed RDOQ increases only the encoding time by 2.6, 1.1 and 0.9% in average for the AI, RA and LD configurations, while the former increases the whole encoding time by an average of 37.8, 10.7 and 8.8%, respectively. Therefore conclusion can be drawn that the proposed RDOQ greatly reduces the coding complexity of the encoder compared with the original RDOQ in HM.

5 Conclusions

In this paper, a high-speed implementation algorithm of RDOQ has been designed to substitute the conventional RDOQ in H.265/HEVC. The proposed algorithm employs the probability distributions of all syntax elements in CABAC to form an accurate bit-rate estimation scheme. It calculates the differences with respect to the RD costs between the candidate levels without utilising the actual CABAC. The proposed algorithm has been integrated into the H.265/HEVC reference software HM. The simulation results have shown that it is able to save over 70% of the total quantisation time of the conventional RDOQ used in HM, while achieving a similar RD performance.

6 Acknowledgments

This work was supported in part by the National Science Foundation of China (grant no. 61371089) and the 111 Project (no. B08038) and in part by the Macau Science and Technology Development Fund under Grant 106/2013/A3 and by the Research Committee at University of Macau under Grants MYRG2014-00003-FST, MRG017/ZYC/2014/FST, MYRG113(Y1-L3)-FST12-ZYC and MRG001/ZYC/2013/FST.

7 References

1 H.265/HEVC: 'High efficiency video coding', ITU-T Rec. H.265 and ISO/IEC 23008-2 (HEVC), ITU-T and ISO/IEC JTC 1/SC 29/WG 11 (MPEG), 2013
2 Sullivan, G.J., Ohm, J.-R., Han, W.-J., Wiegand, T.: 'Overview of the high efficiency video coding (HEVC) standard', *IEEE Trans. Circuits Syst. Video Technol.*, 2012, **22**, (12), pp. 1649–1668
3 H.264/AVC: 'Advanced video coding for generic audio-visual services', ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC), ITU-T and ISO/IEC JTC 1, 2003
4 Ohm, J.-R., Sullivan, G.J., Schwarz, H., Tan, T.K., Wiegand, T.: 'Comparison of the coding efficiency of video coding standards – including high efficient video coding (HEVC)', *IEEE Trans. Circuits Syst. Video Technol.*, 2012, **22**, (12), pp. 1669–1684
5 HEVC Reference Software HM: https://www.hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/
6 Wen, J., Luttrell, M., Villasenor, J.: 'Trellis-based R-D optimal quantization in H.263+', *IEEE Trans. Image Process.*, 2000, **9**, (8), pp. 1431–1434
7 Cote, G., Erol, B., Gallant, M., Kossentini, F.: 'H.263+: video coding at low bit rates', *IEEE Trans. Circuits Syst. Video Technol.*, 1998, **8**, (7), pp. 849–866
8 Wang, W., Cui, H., Tang, K.: 'Rate distortion optimized quantization for H.264/AVC based on dynamic programming'. Proc. SPIE Visual Communications and Image Processing, Beijing, China, July 2005, vol. 5960, no. 4, pp. 2100–2111
9 Karczewicz, M., Ye, Y., Chong, I.: 'Rate distortion optimized quantization'. ITU-T SG16/Q6 Doc. VCEG-AH21, Antalya, Turkey, January 2008, pp. 1–3

10 H.264/AVC Reference Software JM: http://www.iphome.hhi.de/suehring/tml/download/
11 Liu, L., Yourapis, A.M.: 'Rate distortion optimized quantization in the JM reference software', Joint Video Team of ISO/IEC and ITU-T SG16/Q6 Doc. JVT-AA207, Geneva, CH, April 2008, pp. 24–29
12 Xiao, M., Wen, J., Chen, J., Tao, P.: 'An efficient algorithm for joint QP and quantization optimization for H.264/AVC'. Proc. Int. Conf. Image Processing, Hong Kong, China, 2010, pp. 1233–1236
13 Huang, T.-Y., Kao, C.-K., Chen, H.H.: 'Acceleration of rate-distortion optimized quantization for H.264/AVC'. Proc. Int. Symp. Circuits and Systems, 2013, pp. 473–476
14 Sze, V., Budagavi, M.: 'High throughput CABAC entropy coding in HEVC', *IEEE Trans. Circuits Syst. Video Technol.*, 2012, **22**, (12), pp. 1778–1791
15 Sullivan, G.J., Wiegand, T.: 'Rate-distortion optimization for video compression', *IEEE Signal Process. Mag.*, 1998, **15**, (6), pp. 74–90
16 Ortega, A., Ramchandran, K.: 'Rate-distortion methods for image and video compression', *IEEE Signal Process. Mag.*, 1998, **15**, (6), pp. 23–50
17 Li, X., Oertel, N., Hutter, A., Kaup, A.: 'Laplace distribution based Lagrangian rate distortion optimization for hybrid video coding', *IEEE Trans. Circuits Syst. Video Technol.*, 2009, **19**, (2), pp. 193–205
18 Sole, J., Joshi, R., Nguyen, N., *et al.*: 'Transform coefficient coding in HEVC', *IEEE Trans. Circuits Syst. Video Technol.*, 2012, **22**, (12), pp. 1765–1777
19 Clare, G., Henry, F., Jung, J.: 'Sign data hiding', JCTVC Doc. JCTVC-G271, Geneva, Switzerland, November 2011, pp. 1–9
20 Yu, X., Wang, J., He, D., Martin-Cocher, G., Campbell, S.: 'Multiple sign bits hiding', JCTVC Doc. JCTVC-H0481, San Jose, CA, February 2012, pp. 1–8
21 Bossen, F.: 'Common HM test conditions and software reference configurations', JCTVC Doc. JCTVC-J1100, Stockholm, SE, July 2012, pp. 1–3
22 Bjontegaard, G.: 'Calculation of average PSNR differences between RD curves', ITU-T SG16/Q6 Doc. VCEG-M33, Austin, Texas, USA, April 2001, pp. 1–4