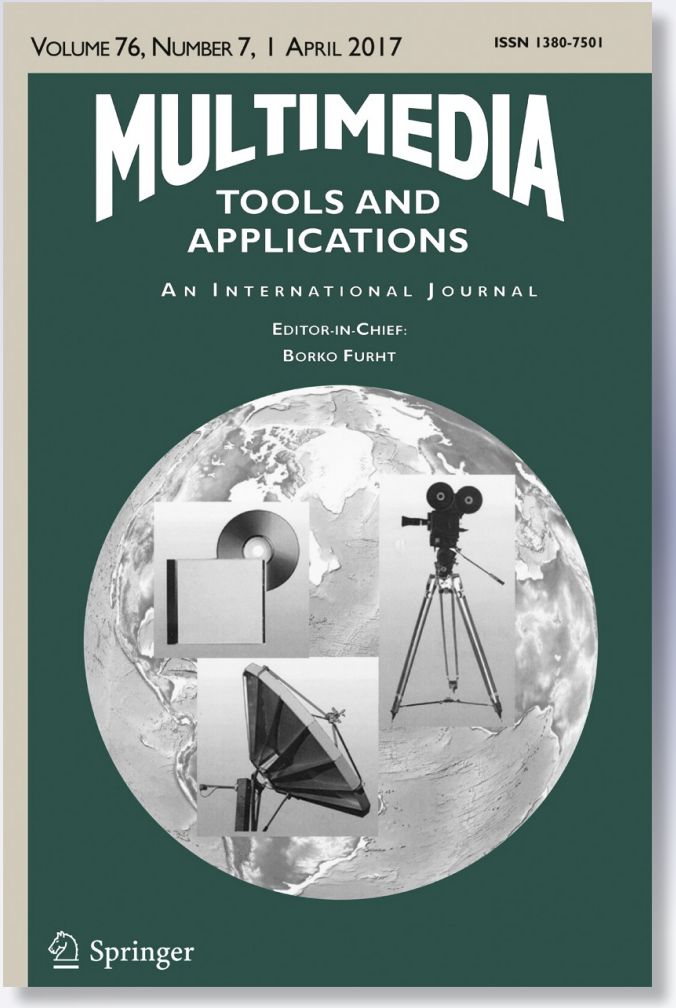
High throughput entropy decoder design for H.265/HEVC

Jung-Ah Choi & Yo-Sung Ho

Multimedia Tools and Applications
An International Journal

ISSN 1380-7501
Volume 76
Number 7

Multimed Tools Appl (2017)
76:9877-9890
DOI 10.1007/s11042-016-3583-z



Springer

Multimed Tools Appl (2017) 76:9877–9890
DOI 10.1007/s11042-016-3583-z

High throughput entropy decoder design for H.265/HEVC

Jung-Ah Choi[1] · **Yo-Sung Ho**[1]

Received: 14 October 2015 / Revised: 20 April 2016 / Accepted: 3 May 2016 /
Published online: 3 June 2016

Abstract The H.265/high efficiency video coding (HEVC) standard employs context-based adaptive binary arithmetic coding (CABAC) as a single entropy coding method. Despite the high coding efficiency of CABAC, its context modeling process limits the throughput and prevents effective parallelization, particularly at the decoder. In this paper, we design a new joint rate distortion-decoding complexity (RDDC) cost function to improve the throughput performance at the H.265/HEVC decoder side. The proposed RDDC cost function includes the complexity term based on a newly designed complexity model using the linear regression. Experimental results show that the proposed method reduces the number of context-coded bins up to 19.8 % without significant coding loss.

Keywords H.265/high efficiency video coding (HEVC) · Throughput improvement · Complexity estimation · Linear regression · Entropy coding · CABAC

1 Introduction

In order to develop a new international video coding standard, ISO/IEC MPEG and ITU-T VCEG formed a joint collaborative team on video coding (JCT-VC). JCT-VC decided to name the outcome of the development as H.265/high efficiency video coding (HEVC) [3, 7]. The first version of H.265/HEVC was completed and released in 2013. Still many experts are working on H.265/HEVC extension and optimization. It is reported that compared with the previous standard, H.264/AVC, H.265/HEVC delivers much increased coding efficiency, as high as 50 % [8].

In H.265/HEVC, context-based adaptive binary arithmetic coding (CABAC) [11] is employed as a unique entropy coder. CABAC is one of the efficient coding tools in H.265/

✉ Jung-Ah Choi
jachoi@gist.ac.kr

✉ Yo-Sung Ho
hoyo@gist.ac.kr

[1] Gwangju Institute of Science and Technology (GIST), 123 Cheomdangwagi-ro, Buk-gu, Gwangju 61005, Republic of Korea

HEVC owing to its context modeling process. However, the high data dependency problem occurs due to consistent sequential processing. This serial nature of CABAC causes throughput limitation which leads to a difficult design for parallel processing. Therefore, CABAC is regarded as the main bottleneck for real-time applications of H.265/HEVC.

In this paper, we focus on the decoding complexity reduction of H.265/HEVC CABAC. To reduce the decoding complexity of H.265/HEVC CABAC, one possible solution is that we force the H.265/HEVC encoder to generate a throughput-friendly bit stream that is easy to decode. We propose a new joint rate-distortion-decoding complexity (RDDC) cost function and we replace the H.265/HEVC rate-distortion (RD) cost function with the RDDC cost function.

There are two main contributions in this work. First, the proposed RDDC cost function considers both the rate-distortion performance and the decoding complexity constraint. Using the proposed cost function, we can balance these constraints efficiently without significant coding loss. Second, we presents the decoding complexity estimation as a machine learning problem which can be solved using regression. It is the novel approach in the throughput improvement research.

The paper is organized as follows. In Section 2, we provide the problem statement for applying H.265/HEVC CABAC to practical applications. Section 3 introduces a new cost function to trade-off rate, distortion, and decoding complexity and how to model the complexity term of this cost function. We evaluate the proposed method in Section 4. Finally, the paper is concluded in Section 5.

2 Problem statement

CABAC is a form of entropy coding based on arithmetic coding. Since CABAC provides much better compression than most other entropy encoding algorithms used in video coding, H.265/HEVC employs CABAC as a single entropy coding engine [5].

CABAC is used after the video is reduced to a series of syntax elements. Figure 1 shows a block diagram of CABAC for a single syntax element. CABAC employs context coding mode and bypass coding mode. The main difference between these two coding modes is the use of a context model. We call the bins coded by context coding as context-coded bins. In this case, context models have to be specified. Otherwise, bypass coding assumes an equivalent probability mode. Bypass bins does not require any context model.

The high coding efficiency of H.265/HEVC CABAC can be achieved by an accurate context estimate. The context modeling is highly adaptive and selects one of several hundred different contexts depending on the type of syntax element, the bin index, neighboring information, etc. Since the probabilities are non-stationary, the contexts are updated after each bin. These data dependencies result in the feedback loop in Fig. 1. Hence, in order to improve the throughput performance, the occurrence of the context-coded bins should be avoided if possible.

There were many researches to improve the throughput of quantized transform coefficients (QTCs) in CABAC entropy coder/decoder with minimal coding loss. CABAC has two coding modes, but most of the symbols are encoded by context coding mode. It is reported that the throughput bottleneck is primarily due to context selection dependencies [13]. Thus, many researchers tried to reduce the number of context-coded bins for QTC syntax elements.

Based on the existing adaptive Rice binarization of CABAC, Nguyen et al. proposed a simple entropy coding scheme [12]. Instead of using the truncated unary binarization,

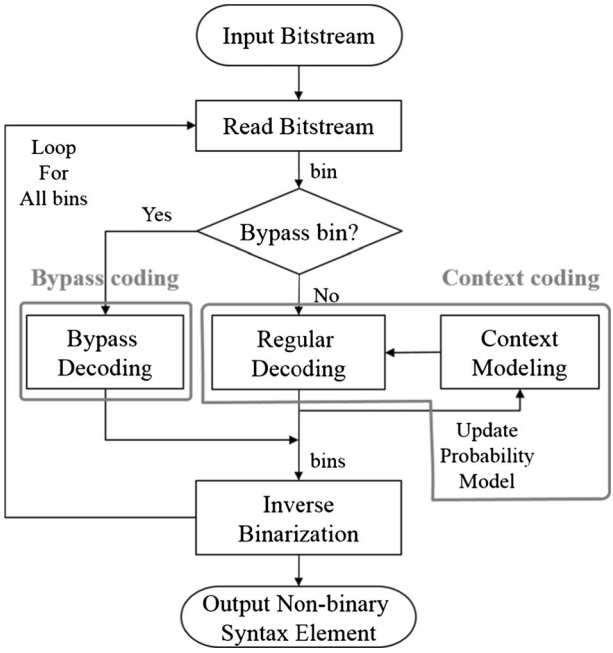


Fig. 1 CABAC decoding process for a single syntax element

Nguyen's method starts directly with the Rice binarization. Thus, all bins resulting from the binarization of a transform coefficient level are coded as bypass bins. Thus, the throughput is improved by reducing the number of context-coded bins and using bypass bins instead. However, this scheme showed significant bit-rate increase up to a 9.4 %, and it was not adopted into the final H.265/HEVC standard.

Kim et al. also proposed a new high throughput binarization method for high throughput CABAC [9]. They proposed two binarization methods: VLC table-based binarization and Golomb-Rice code-based binarization. VLC table-based binarization is for the throughput efficiency and Golomb-Rice code-based binarization is for the coding efficiency. The benefit of this method is that it can use two types of binarization methods selectively according to the target application.

Lainema's method is also about the high throughput binarization method, but it is slightly different [10]. For the improved throughput performance of entropy coding, they use CABAC bypass mode for all transform coefficient data and applies CAVLC coefficient coding engine as the binarization mode. As another option, the proposed method supports full CABAC coding engine, referred to as high efficiency binarization mode. To classify these two options (which binarization scheme is to be applied), a slice-level indication syntax element is required. Unfortunately, the Kim's method and the Lainema's method reduce the coding efficiency up to 9.5 % and 11.1 %, respectively.

Chen et al. proposed more efficient and safe throughput improvement scheme [4]. They limit the number of syntax elements which is coded as context-coded bins. In their proposed method, the syntax element *coeff_abs_level_greater1_flag* is coded only for eight starting non-zero coefficients and the syntax element *coeff_abs_level_greater2_flag* is coded only for one non-zero coefficient in 4×4 sub-block. For the remaining coefficients, an early switch to Golomb-Rice bypass mode is applied. Figure 2 describes an example of the Chen's method.

18	6	-6	-1
-12	4	-4	0
7	4	2	1
2	4	-1	0

4×4 sub-block

16	14	11	7
15	12	8	4
13	9	5	2
10	6	3	1

Entropy coding order

Coefficient	0	1	-1	0	2	4	-1	-4	4	2	-6	4	7	6	-12	18
significant_coeff_flag	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
coeff_abs_level_greater1_flag		0	0		1	1	0	1	1	1						
coeff_abs_level_greater2_flag					0											
coeff_sign_flag		0	1		0	0	1	1	0	0	1	0	0	0	1	0
coeff_abs_level_remaining						2		2	2	0	5	3	6	5	10	17

Fig. 2 An example of the Chen's binarization method adopted in the HEVC standard [4]

Chen's method shows slight loss of coding efficiency degradation, so it is adopted in the H.265/HEVC standard.

3 Proposed algorithm

3.1 Joint rate-distortion-decoding complexity optimization

The one of big changes in the H.265/HEVC standard is the introduction of larger coding block sizes: 64 × 64 pixels instead of 16 × 16 pixels in H.264/AVC. The largest coding unit is referred to as coding tree unit (CTU). The CTU of 64 × 64 pixels can be recursively partitioned into four equally sized coding units (CUs). The CU size can varies from 64 × 64 to 8 × 8, which means CTU may contain one CU or several quad-tree partitioned CUs. Each CU can be split into prediction unit (PU) and transform unit (TU). The possible PU size is from 64 × 64 to 8 × 8 and the possible TU size is from 32 × 32 to 4 × 4. The H.265/HEVC encoder checks all the possible partitions using a depth-first search procedure.

To find the most efficient partitioning, the rate-distortion optimization (RDO) process is applied. This technique is based on the estimation of rate R and distortion D for the given block. Figure 3 illustrates the flowchart of the RDO process in H.265/HEVC. Rate R is

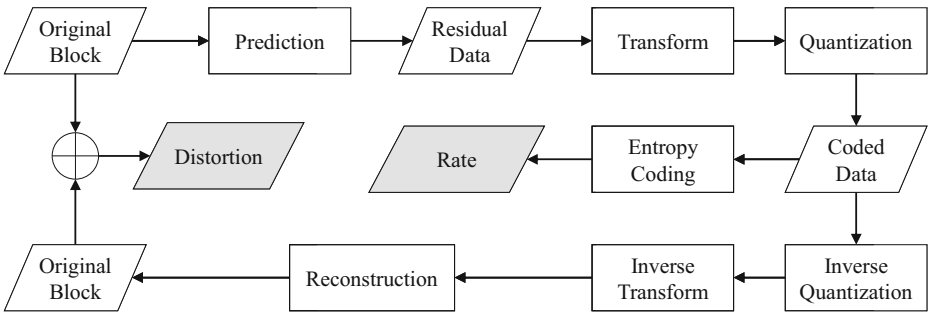


Fig. 3 The flowchart of the RDO process in H.265/HEVC

counted after the quantized transform data is entropy coded. Distortion D is calculated after the block is reconstructed and compared with the original data.

In the RDO process, the optimal partitioning set including CU, PU, and TU are chosen as the one with the minimal RDO cost. The RDO cost of the i-th depth unit is defined by

$$J_i^{R,D} = D_i + \lambda \cdot R_i \tag{1}$$

where λ is the Lagrangian multiplier and its value is a constant that is typically determined empirically. In H.265/HEVC, the encoder finds the optimal size combinations of the CU, PU, and TU, according to the "try all and find the best" strategy. After comparing all possible combinations, the one that has minimum RD cost is selected as the best mode. The RDO estimates the trade-off between rate and distortion, thus providing a criterion of choosing the best coding option.

In order to consider the decoding complexity together, we include the complexity cost in the RD cost in Eq. (1). We define a new joint rate-distortion-decoding complexity (RDDC) cost function as

$$J_i^{R,D,C} = J_i^{R,D} + \lambda' \cdot C_i \tag{2}$$

where $J_i^{R,D,C}$ is the RD cost function defined in Eq. (1), C_i is the complexity cost term of the i-th depth unit. λ' is the Lagrangian multiplier of the complexity term. Intuitionally, we can know that the CABAC decoding complexity is proportional to rate, so we decide λ' as the same with λ.

$$\lambda' = \lambda \tag{3}$$

We replace the conventional RD cost function $J_i^{R,D}$ to the joint RDDC cost function $J_i^{R,D,C}$. Since we add the decoding complexity term into the RDO process, the minimization of the cost function implies that the selected partitioning combination is the result of the trade-off among the decoder complexity, distortion, and rate. For the joint RDDC cost function, we need a quantitative model for the decoding complexity and it will be discussed in Section 3.2.

3.2 CABAC decoding complexity model

The proposed CABAC decoding complexity estimation is developed based on a linear regression model. Linear regression is an approach for modeling the relationship between a dependent variable y and independent variables denoted x_i. This model is represented by a linear combination of the independent variables x and the corresponding weights ω.

$$y = \omega_0 + \omega_1 x_1 + \cdots + \omega_m x_m = \omega_0 + \sum_{i=1}^{m} \omega_i x_i \tag{4}$$

Here, weights $\omega_0, \omega_1, \ldots, \omega_m$ were estimated using the least square method which finds the set of parameters that minimize the sum of the squared difference between the observed responses and the model.

For the proposed complexity model, y becomes the CABAC decoding complexity and x_i becomes a set of features. In order to modeling CABAC decoding complexity, we should first select the set of features available for estimating decoding complexity, especially throughput.

The QTCs account for a significant portion of the total bit-rates of a compressed video. As shown in Fig. 4, the decoding process of the QTCs are quite complex. As a result, the

```
Void TDecSbac::parseCoeffNxN()
{
  // decode the last coefficient position
  parseLastSignificantXY();
  for( Int iSubSet = iLastScanSet; iSubSet >= 0; iSubSet-- )
  {
    // decode coded_sub_block_flag
    if( iSubSet == iLastScanSet || iSubSet == 0)
      uiSigCoeffGroupFlag[ iCGBlkPos ] = 1;
    else
    {
      m_pcTDecBinIf->decodeBin( uiSigCoeffGroup, baseCoeffGroupCtx[ uiCtxSig ] );
      uiSigCoeffGroupFlag[ iCGBlkPos ] = uiSigCoeffGroup;
    }

    // decode significant_coeff_flag
    for( ; iScanPosSig >= iSubPos; iScanPosSig-- )
    {
      if( uiSigCoeffGroupFlag[ iCGBlkPos ] )
      {
        if( iScanPosSig > iSubPos || iSubSet == 0  || numNonZero )
          m_pcTDecBinIf->decodeBin( uiSig, baseCtx[ uiCtxSig ] );
        else
          uiSig = 1;
      }
    }

    // decode coeff_abs_level_greater1_flag
    for ( Int i = 0; i < numNonZero; i++) absCoeff[i] = 1;
    Int numC1Flag = min(numNonZero, C1FLAG_NUMBER);
    Int firstC2FlagIdx = -1;
    for( Int idx = 0; idx < numC1Flag; idx++ )
    {
      m_pcTDecBinIf->decodeBin( uiBin, baseCtxMod[c1] );
      if( uiBin == 1 )
      {
        c1 = 0;
        if (firstC2FlagIdx == -1)
          firstC2FlagIdx = idx;
      }
      else if( (c1 < 3) && (c1 > 0) )
      {
        c1++;
      }
      absCoeff[ idx ] = uiBin + 1;
    }

    // decode coeff_abs_level_greater2_flag
    if (c1 == 0)
      if( firstC2FlagIdx != -1)
      {
        m_pcTDecBinIf->decodeBin( uiBin, baseCtxMod[0] );
        absCoeff[firstC2FlagIdx] = uiBin + 2;
      }

    // decode coeff_sign_flag
    m_pcTDecBinIf->decodeBinsEP( coeffSigns, numNonZero );

    // decode coeff_abs_level_remaining
    if (c1 == 0 || numNonZero > C1FLAG_NUMBER)
    {
      for( Int idx = 0; idx < numNonZero; idx++ )
      {
        UInt baseLevel  = (idx < C1FLAG_NUMBER)? (2 + iFirstCoeff2) : 1;
        if( absCoeff[ idx ] == baseLevel)
        {
          xReadGoRiceExGolomb( uiLevel, uiGoRiceParam );
          absCoeff[ idx ] = uiLevel + baseLevel;
          if(absCoeff[idx]>3*(1<<uiGoRiceParam))
            uiGoRiceParam = min<UInt>(uiGoRiceParam+ 1, 4);
        }
      }
    }
  }
}
```

Fig. 4 The CABAC decoding process of quantized transform coefficients (QTCs)

compression of QTCs significantly impacts the overall CABAC complexity. Table 1 shows the syntax elements for QTCs in H.265/HEVC. There are total 10 syntax elements: *last_significant_coeff_x_prefix, last_significant_coeff_y_prefix, last_significant_coeff_x_suffix, last_significant_coeff_y_suffix, coded_sub_block_flag, sig_coeff_flag, coeff_abs_level_greater1_flag, coeff_abs_level_greater2_flag, coeff_sign_flag, and coeff_abs_level_remaining*.

Shaded table cells represent syntax elements coded as the context-coded bins. As we discussed earlier, context coded bins are mainly responsible for lower throughput of CABAC. Thus, we select the number of context-coded bins of these syntax elements as the feature of our decoding complexity model. The decoding complexity model is defined as follows.

$$C = \omega_0 + \omega_1 N_x + \omega_2 N_y + \omega_3 N_{csb} + \omega_4 N_{sig} + \omega_5 N_{gr1} + \omega_6 N_{gr2} \tag{5}$$

where N_x is the number of context coded bins of *last_significant_coeff_x_prefix*, N_y is the number of context coded bins of *last_significant_coeff_y_prefix*, N_{csb} is the number of context coded bins of *coded_sub_block_flag*, N_{sig} is the number of context coded bins of *sig_coeff_flag*, N_{gr1} is the number of context coded bins of *coeff_abs_level_greater1_flag*, N_{gr2} is the number of context coded bins of *coeff_abs_level_greater2_flag*. C is CABAC decoding complexity.

The weights for the model is obtained by off-line training with certain training set. The training is conducted by the following procedures:

1. Choose various video test sequences with different characteristics: resolution, texture complexity, motion complexity, etc. We use five different test sequences: Traffic (2560 × 1600), Kimono (1920 × 1080), BQMall (832 × 480), BlowingBubbles (416 × 240), and KristenAndSara (1280 × 720). The details for these test sequences are described in Table 2.
2. Encode the video sequences with a standard H.265/HEVC encoder using various quantization parameters (QPs): 22, 27, 32, and 37. We obtain 20 encoded bitstreams and the number of training data becomes the millions as a result (Note that millions of training samples was used to derive the weights: the number of training data = 5 test sequences × 4 QPs × number of blocks). Meanwhile, the numbers of clock-ticks are measured by the Intel VTune performance analyzer 8.0 and measured clock-ticks become the target decoding complexity C_t.

Table 1 Syntax elements of quantized transform coefficients (QTCs)

Syntax Element
last_significant_coeff_x_prefix
last_significant_coeff_y_prefix
last_significant_coeff_x_suffix
last_significant_coeff_y_suffix
coded_sub_block_flag
sig_coeff_flag
coeff_abs_level_greater1_flag
coeff_abs_level_greater2_flag
coeff_sign_flag
coeff_abs_level_remaining

Table 2 Details of training test sequences

Sequence	Resolution	Frame count	Frame rate
PeopleOnStreet	2560 × 1600	150	30fps
Kimono	1920 × 1080	240	24fps
BQMall	832 × 480	600	60fps
BlowingBubbles	416 × 240	500	50fps
KristenAndSara	1280 × 720	600	60fps

3. From the encoded bitstreams, we extract the number of context coded bins of each QTC syntax element and these values become the independent variables N_x, N_y, N_{csb}, N_{sig}, N_{gr1}, N_{gr2} in Eq. (5).
4. Using the collected data at Step 2 and Step 3, we find the weights of the decoding complexity model in Eq. (5). The weights were estimated based on the multiple linear regression. It finds the weights that minimizes the sum of the squared difference e between the observed decoding complexity C_t at Step 2 and the complexity model C.

$$e = \sum_{i=1}^{N} [C_t - C]^2 \tag{6}$$

In order to design the decoding complexity model, we conduct the above training procedure on the PC platform. The CPU was Intel(R) Xeon(R) CPU E5630 2.53 GHz CPU with 32 GB RAM and the operating system was Windows 7. To make the encoded bitstreams, we use the encoder of the H.265/HEVC reference software version 8.0 (HM 8.0) [14]. Using above procedure, we can find the best fitting of weights and the decoding complexity can be approximated as

$$C = 0.3795 + 0.469N_x + 0.4201N_y + 0.3772N_{sig} + 0.4869N_{gr1} \tag{7}$$

Comparing with Eq. (5), third and sixth terms are discarded, because found ω_3 and ω_6 were zero in our training result. It means that *coded_sub_block_flag* and *coeff_abs_level_greater2_flag* do not affect the decoding complexity much. This result is reasonable because these two syntax elements do not occupy a large amount of context coded bins. As explained above, *coded_sub_block_flag* is coded once in a TU and the maximum possible number of context coded bins caused by this syntax element is one. So, the portion of the context coded bins of this syntax element is not enough to affect the complexity. Also, *coeff_abs_level_greater2_flag* is coded only one non-zero coefficient in 4 × 4 sub-block, according to the Chen's algorithm [4] adopted in the H.265/HEVC standard.

The above weights were adopted in our decoding complexity model to estimate the CABAC decoding complexity. Figure 5 shows the correlation between the estimated decoding complexity based on the proposed complexity model and the actual decoding complexity for *BQMall*, one of the training video sequences. Figure 6 shows the correlation between the estimated decoding complexity based on the proposed complexity model and the actual decoding complexity for *Traffic*, a non-training video sequence. The correlation is big as the results are in the straight line. From Figs. 5 and 6, we can observe that the proposed complexity model provides good estimation results for bit streams of non-training sets as well as bit streams of training sets.

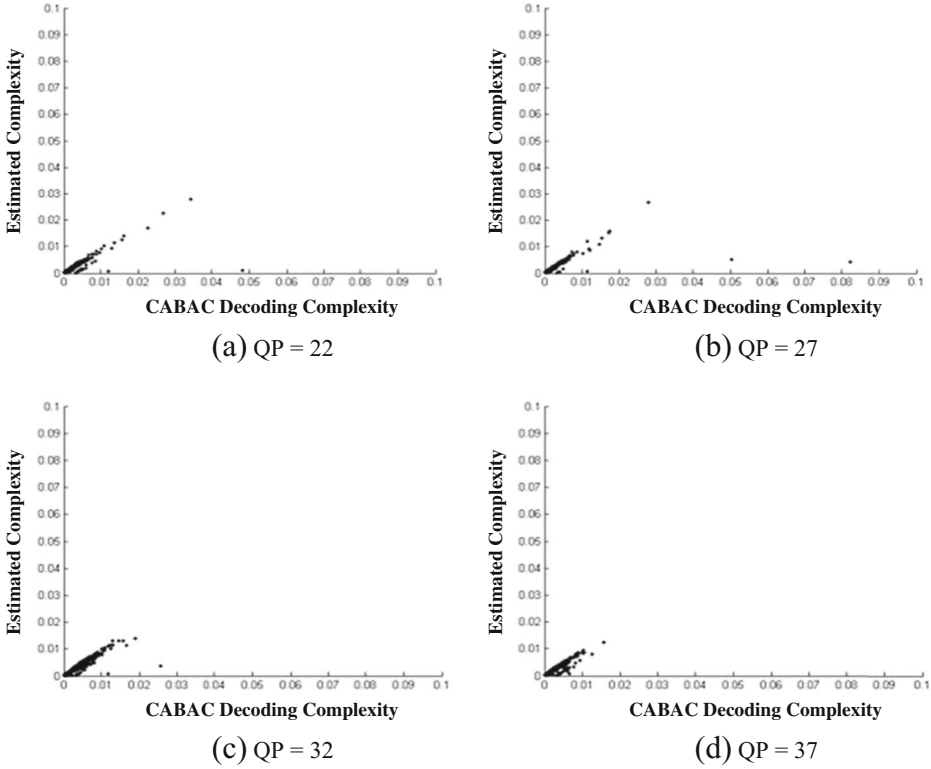


Fig. 5 The comparison between the estimated decoding complexity and the actual decoding complexity (*BQMall*)

We apply the above complexity model as the complexity term of Eq. (2). Since there are high correlation between the CABAC decoding complexity and our proposed complexity model, the complexity term of the proposed joint RDDC cost function fan reflect the actual decoding complexity of CABAC.

4 Experimental results

In order to verify the efficiency of the proposed method, we conducted experiments to verify the proposed CABAC decoding complexity model and its decoding complexity control scheme on the PC platform. The CPU was Intel(R) Xeon(R) CPU E5630 2.53 GHz CPU with 32 GB RAM and the operating system was Windows 7. We implemented the proposed method in the H.265/HEVC reference software version 8.0 (HM 8.0) [6]. The anchor data for the performance comparison is generated by HM 8.0.

The H.265/HEVC encoder equipped with the proposed RDDC cost function can generate bit streams to meet different decoding complexity constraints. To validate this claim, we compared the proposed method to H.265/HEVC transform coefficient coding with respect to context-coded bin usage and coding efficiency. We tested 13 test sequences provided by JCT-VC. These sequences differ broadly from one another in terms of frame

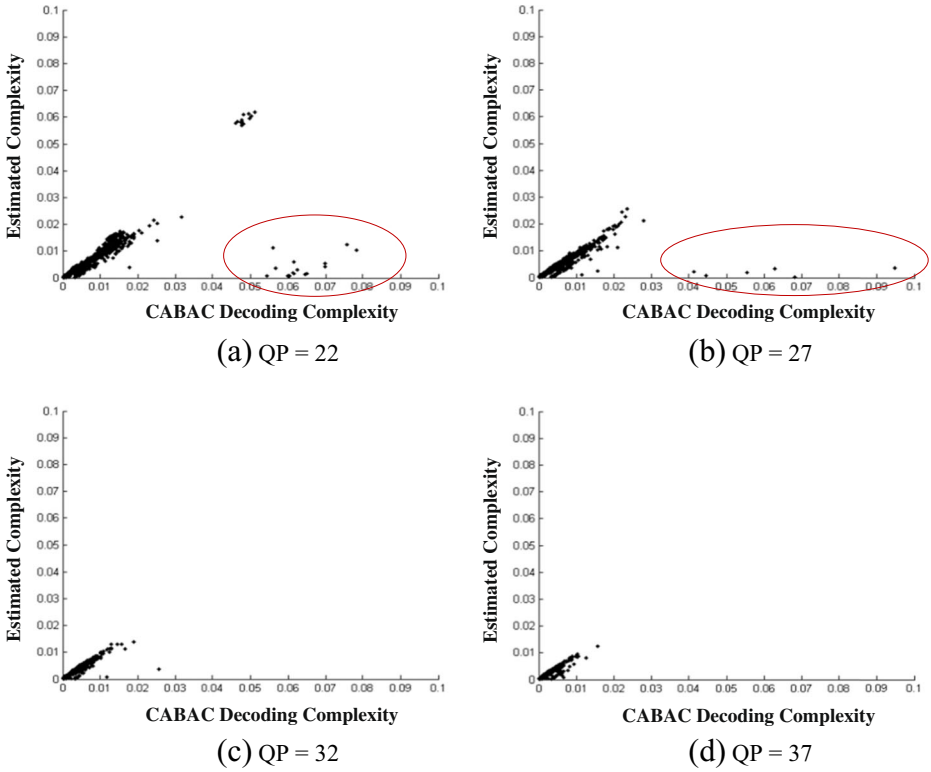


Fig. 6 The comparison between the estimated decoding complexity and the actual decoding complexity (*Traffic*)

rate, motion and texture characteristics as well as spatial resolution. Frame count, frame rate, bit depth, and spatial resolution for each video sequence are presented in the JCT-VC test conditions [2].

Table 3 shows the number of used context-coded bins of H.265/HEVC transform coefficient coding and the proposed method. The proposed method reduces the number of context-coded bins up to 19.8 %. Note that the proposed method provides better decoding complexity reduction in both lower QP value and higher QP value.

Table 4 shows the coding efficiency of the proposed method, compared with H.265/HEVC transform coefficient coding. To evaluate the coding efficiency, the Bjøntegaard delta bit-rate (BD-BR) [1] was used which is recommended by video coding standard organizations. Negative values tell how much lower the bit rate is reduced, and positive values tell how much the bit rate is increased for the same PSNR. The average coding efficiency loss of the proposed method is only 0.47 % BD-BR. Also, we checked Bjøntegaard delta PSNR (BD-PSNR) [1], which represents average difference of PSNR (dB) in same bitrate. The average BD-PSNR is only -0.025 dB and it means that the proposed method does not degrade visual quality of the coded picture block much.

In Fig. 7, we represents the comparison of decoded images for several test sequence (*Traffic*, *ParkScene*, *BasketballDrill*, and *BasketballPass*) at QP = 22. As shown in Table 3, the proposed method shows significant throughput improvement in QP = 22. Fig. X, we can

Table 3 The number of reduced context-coded bins (%)

Sequence	Resolution	QP = 22	QP = 27	QP = 32	QP = 37
Traffic	2560 × 1600	–11.5	–11.5	–11.0	–9.8
PeopleOnStreet		–12.8	–11.8	–12.1	–12.5
Kimono	1920 × 1080	–17.2	–17.1	–15.3	–11.6
ParkScene		–11.7	–10.9	–10.5	–9.9
Cactus		–12.2	–10.5	–9.9	–9.7
BasketballDrill	832 × 480	–10.8	–10.3	–11.7	–5.8
BQMall		–14.5	–11.8	–9.5	–9.6
PartyScene		–17.0	–13.7	–9.3	–6.9
RaceHorses		–15.4	–11.9	–7.0	–8.1
BasketballPass	416 × 240	–10.9	–9.8	–7.6	–12.2
BQSquare		–19.8	–19.2	–13.4	–7.4
BlowingBubbles		–13.7	–8.5	–8.4	–7.4
RaceHorses		–14.0	–8.5	–7.8	–5.9
Average		–14.0	–12.0	–10.3	–9.0

observe that the quality degradation caused by the proposed method is hard to recognize. Therefore, we are convinced that the proposed joint RDDC cost function does not introduce significant visual artifacts. From these results, we can verify that the proposed method has no noticeable effect on coding efficiency.

Based on experimental results, we ascertain that the H.265/HEVC encoder with the proposed RDDC cost function can generate bit streams to reduce the decoding complexity. The generated bit streams can save a significant amount of decoding complexity without noticeable coding efficiency loss. This is particularly interesting in a mobile

Table 4 The coding efficiency change

Sequence	Resolution	BD-BR (%)	BD-PSNR (dB)
Traffic	2560 × 1600	+0.71	–0.038
PeopleOnStreet		+0.64	–0.036
Kimono	1920 × 1080	+0.30	–0.011
ParkScene		+0.88	–0.038
Cactus		+0.63	–0.023
BasketballDrill	832 × 480	+0.65	–0.031
BQMall		+0.39	–0.025
PartyScene		+0.33	–0.026
RaceHorses		+0.48	–0.031
BasketballPass	416 × 240	+0.91	–0.053
BQSquare		–0.33	+0.026
BlowingBubbles		+0.37	–0.024
RaceHorses		+0.19	–0.016
Average		+0.47	–0.025

(a) *Traffic* (left: encoded by HM 8.0, right: encoded by the proposed method)

(b) *ParkScene* (left: encoded by HM 8.0, right: encoded by the proposed method)

(c) *BasketballDrill* (left: encoded by HM 8.0, right: encoded by the proposed method)

(d) *BasketballPass* (left: encoded by HM 8.0, right: encoded by the proposed method)

Fig. 7 Decoded image quality comparison

broadcasting environment, where multiple mobile devices will get broadcast/streaming video in real time.

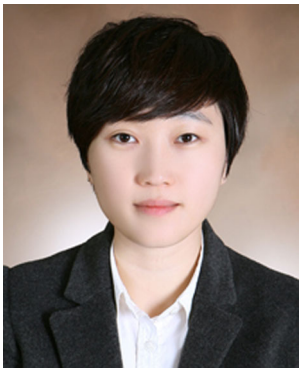
5 Conclusions

In this paper, we focused on how to realize high throughput entropy decoder for H.265/high efficiency video coding (HEVC). First, we proposed a joint rate-distortion-decoding complexity (RDDC) cost function to improve the throughput of context-based adaptive binary arithmetic coding (CABAC). The proposed cost function includes the additional complexity term that is based on our newly designed decoding complexity model. We apply this complexity model to the H.265/HEVC encoder to generate decoder-friendly bit streams. The proposed method can balance the trade-off between the rate-distortion requirement as well as the computational power of the decoding platform. Compared to H.265/HEVC CABAC, the proposed method provides fairly good throughput improvement results various video sequences without significant coding efficiency loss.

Acknowledgments This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2011-0030079).

References

1. Bjøntegaard G (2008) Improvements of the BD-PSNR model. VCEG-AI11. 35th VCEG Meet
2. Bossen F (2012) HM 8 common test conditions and software reference configurations. JCTVC-J1100. 10th JCT-VC Meet
3. Bross B, Han W, Ohm J, Sullivan G, Wiegand T (2012) High efficiency video coding (HEVC) text specification draft 8. JCTVC-J1003. 10th JCT-VC Meet
4. Chen J, Chien W, Joshi R, Sole J, Karczewicz M (2012) Non-CE1: throughput improvement on CABAC coefficients level coding. JCTVC-H0554. 8th JCT-VC Meet
5. Choi J, Ho Y (2015) Improved entropy coding for quantized transform coefficients in HEVC screen content coding. SIViP 9(5):1067–1079
6. H.265/HEVC reference software: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/
7. Ho Y, Choi J (2012) Advanced video coding techniques for smart phones. Proc Int Conf Embedded Syst Intell Technol (ICESIT)
8. Huang C, Tikekar M, Juvekar C, Sze V (2013) A 249Mpixel/s HEVC video-decoder chip for quad full HD applications. Proc 2013 I.E. Int Solid-State Circ Conf Digest Tech Papers
9. Kim S, Misra K, Kerofsky L, Segall A (2012) Non-CE1: high throughput binarization (HTB) method with modified level coding. JCTVC-H0510. 8th JCT-VC Meet
10. Lainema J, Ugur K, Hallapuro A (2012) CE1.D1: Nokia report on high throughput binarization. JCTVC-H0232. 8th JCT-VC Meet
11. Marpe D, Schwarz H, Wiegand T (2003) Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. IEEE Trans CSVT 13(7):620–636
12. Nguyen T, Marpe D, Siekmann M, Wiegand T (2012) Non-CE1: high throughput coding scheme with rice binarization. JCTVC-H0458. 8th JCT-VC Meet
13. Sze V, Chandrakasan A (2012) A highly parallel and scalable CABAC decoder for next generation video coding. IEEE J Solid State Circuits 47(1):8–22



Jung-Ah Choi received the B.S. degree in electronic engineering and avionics from Korea Aerospace University, Korea, in 2007 and the M.S. and Ph.D. degrees in information and communication engineering from Gwangju Institute of Science and Technology (GIST), Korea, in 2008 and 2014, respectively. Her research interests are digital image/video coding and signal processing.

Yo-Sung Ho received the B.S. and M.S. degrees in electronic engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, Santa Barbara, in 1990. He joined the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea, in 1983. From 1990 to 1993, he was with Philips Laboratories, Briarcliff Manor, NY, where he was involved in development of the advanced digital high-definition television system. In 1993, he rejoined the Technical Staff of ETRI and was involved in development of the Korea direct broadcast satellite digital television and high-definition television systems. Since 1995, he has been with the Gwangju Institute of Science and Technology, Gwangju, Korea, where he is currently a Professor in the Department of Information and Communications. His research interests include digital image and video coding, image analysis and image restoration, advanced coding techniques, digital video and audio broadcasting, 3-D television, and realistic broadcasting.