Efficient SNR Scalable Coding using Context-based Adaptive Binary Arithmetic Coding

Seung-Hwan Kim and Yo-Sung Ho
Gwangju Institute of Science and Technology (GIST)
1 Oryong-Dong, Buk-Gu, Gwangju, 500-712, Korea
{kshkim, hoyo}@gist.ac.kr

Abstract: The H.264/AVC standard represents an important step in the evolution of video coding standards since it offers a significant improvement in terms of rate-distortion efficiency against the popular MPEG-2 and MPEG-4 video standard. Hence, H.264/AVC have been researched for basic coding tool in many research areas, such as MPEG3D and scalable extension works in MPEG standard. In this work, an efficient SNR scalable video coding scheme, named H.264-CABAC-BPFGS, is presented and evaluated. The proposed H.264-CABAC-BPFGS uses a H.264/AVC conformant base layer and the CABAC (context-based adaptive arithmetic coding) scheme for the enhancement layer. The H.264-CABAC-BPFGS coding scheme also retains the MPEG-4 FGS functionalities, such as adaptation to dynamic changes in network conditions, packet loss resilience, etc.

1. Introduction
The objective of video coding for Internet streaming video is changing to optimizing the video quality over a given bit rate range instead of at a given bit rate. Therefore, the bitstream should be partially decodable at any bit rate within the bit rate range to reconstruct a video signal with the optimized quality at that bit rate. [1]

The fine granularity scalability (FGS) provides the functionality that the given bitstream can be partially decodable at any bit rate. Since the prediction is always based on base layer, FGS suffers from reduced coding performance in contrast to traditional SNR scalability included in MPEG-2. However, in the traditional SNR scalability, if there is an error or packet loss in the enhancement layer, errors propagate to the end of a GOP and cause serious drifting problem in the prediction frames followed. Therefore, FGS is one of most important functionality in the SNR scalable video coding. [2]

In this paper, we implemented two kinds of FGS-based SNR scalable codecs called H.264-BPFGS and H.264-CABAC-BPFGS, respectively. In both cases, H.264/AVC is used for base layer. In H.264-BPFGS, bit plane coding method using VLC table is used to encode the enhancement layer. In H.264-CABAC-BPFGS, context-based binary adaptive arithmetic coding scheme is used for the enhancement layer.

2. Review of MPEG-4 FGS
Fig. 1 shows the encoder structure of the MPEG-4 FGS system. As shown in Fig. 1, the FGS encoder consists of two parts: base layer and enhancement layer. In the base layer, the basic information of the input signal is coded in the same way as the traditional block-based coding method. In the enhancement layer, the residual signal that is not coded in the base layer is divided into 8×8 blocks and each block is DCT transformed. All the 64 DCT coefficients in each block are bitplane coded using VLC table. [1]

3. H.264-BPFGS
MPEG-4 ASP (Advanced Simple Profile) is the MPEG-4 FGS base layer coding solution. Hence, the use of H.264/AVC encoder in the base layer is expected to bring improvements in coding efficiency. There has been some work reported regarding the combination of the MPEG-4 FGS and the H.264/AVC standards. The approach taken in the H.264-BPFGS is a direct implementation of the MPEG-4 FGS enhancement layer, without any modification, on the top of the H.264/AVC base layer. The base layer is implemented by using H.264/AVC reference software version JM 9.5. The enhancement layer provides fine granular scalability through bitplane encoding. Following steps show the procedure of bitplane coding of 64 DCT coefficients in each block:
a) Zigzag order 2-D quantized DCT array into a 1-D array
   10, 0, -6, 1, -3, 0...0 (64 DCT coefficients)
b) Take absolute values and signs of the DCT coefficients
   +, x, +, +, x,..., x (sign bits)
c) Find the largest absolute value and the minimum number of bits, N, needed to represent it in the binary format. N is the number of bitplanes.

Largest absolute value: 10 = 1010_2 (N=4)

d) Represent every one of the quantized DCT coefficients with N bits in the binary format and form N bit planes

1, 0, 0, 0, 0, 0, ..., 0 (MSB)
0, 0, 1, 0, 0, 0, ..., 0 (MSB-1)
1, 0, 1, 0, 1, 0, ..., 0 (MSB-2)
0, 0, 0, 1, 1, 0, ..., 0 (MSB-3)

e) For each bitplane, 2-D symbols are formed of three components:

a. Number of consecutive 0’s before a 1 (Run),
b. End-Of-Plane (EOP).

(0, 1) (MSB)
(2, 1) (MSB-1)
(0, 0, (1, 0), (1, 1)) (MSB-2)
(3, 0, (0, 1)) (MSB-3)

f) The sign bits of all non-zero coefficients are put at the end of the codes for coding the bit planes.

(0, 1) (+) (MSB)
(2, 1) (-) (MSB-1)
(0, 0, (1, 0), (1, 1)) (-) (MSB-2)
(3, 0, (-) (0, 1)) (-) (MSB-3)

Finally, each (RUN, EOP) symbols in bit-plane is entropy coded using VLC tables used in MPEG-4 FGS to produce the output bitstream. [1]

4. H.264-CABAC-BPFGS

Huffman coding can be optimum if the symbol probability is an integer power of 1/2, which is usually not the case. Arithmetic coding encodes data by creating a code string, which represents a fractional value on the number line between 0 and 1 [5]. It encourages clear separation between the model for representing data and the encoding of information with respect to that model. Another advantage of arithmetic coding is that it dispenses with the restriction that each symbol must translate into an integral number of bits, thereby coding more efficiently. It actually achieves the theoretical entropy bound to compression efficiency for any source. Therefore, in order to enhance the coding efficiency of the enhancement layer of H.264-BPFGS, we propose H.264-CABAC-BPFGS.

Fig. 2 shows the basic structure for enhancement layer coding in H.264-CABAC-BPFGS. The encoding process consists of, at most, three elementary steps:

a) Binarization
b) Context modeling
c) Binary arithmetic coding

In the binarization step, residual image obtained from subtracting reconstructed image from original image is divided into 8x8 blocks and transformed (DCT). Those DCT coefficients are zigzag scanned and form several bit planes (same process in MPEG-4 FGS).

4.1 Context modeling

In the context modeling step, corresponding context models are selected for each element of the bin string in each block. Before explain the procedure of context modeling, we define some terminologies.

BPL (bitplane level): the index of bitplane layer of given data in each frame.

DL (data level): the index of bitplane layer of given data in each 8x8 block.

SP (scanning position): the order of zigzag scanning of given data in each 8x8 block (0–63)

For the coding of each bitplane in enhancement layer, we need to find the probability distribution of each binary symbol. Table 1 and Table 2 show the probability distribution of zero value in each plane and SP, respectively. Based on the selected BPL, DL and SP, we are try to estimate the probability of given symbol by using the neighboring symbols and upper layer symbols which are already encoded.

Table 1. Probability of zero in each bitplane

<table>
<thead>
<tr>
<th>BPL</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.983</td>
<td>0.973</td>
<td>0.964</td>
<td>0.959</td>
</tr>
<tr>
<td>1</td>
<td>0.964</td>
<td>0.914</td>
<td>0.906</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.913</td>
<td>0.837</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.838</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Probability of zero in SP

<table>
<thead>
<tr>
<th>SP</th>
<th>DL</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>i=0</td>
<td>0.452</td>
<td>0.535</td>
<td>0.589</td>
<td>0.571</td>
<td></td>
</tr>
<tr>
<td>0&lt;i&lt;3</td>
<td>0.856</td>
<td>0.662</td>
<td>0.637</td>
<td>0.625</td>
<td></td>
</tr>
<tr>
<td>3&lt;i&lt;7</td>
<td>0.899</td>
<td>0.788</td>
<td>0.694</td>
<td>0.673</td>
<td></td>
</tr>
<tr>
<td>7&lt;i&lt;14</td>
<td>0.954</td>
<td>0.865</td>
<td>0.741</td>
<td>0.726</td>
<td></td>
</tr>
<tr>
<td>14&lt;i&lt;25</td>
<td>0.991</td>
<td>0.893</td>
<td>0.776</td>
<td>0.734</td>
<td></td>
</tr>
<tr>
<td>25&lt;i&lt;53</td>
<td>0.996</td>
<td>0.961</td>
<td>0.898</td>
<td>0.856</td>
<td></td>
</tr>
<tr>
<td>53&lt;i&lt;63</td>
<td>0.999</td>
<td>0.992</td>
<td>0.971</td>
<td>0.954</td>
<td></td>
</tr>
</tbody>
</table>

Basic procedure of proposed CABAC is as follows:

a) Select the appropriate type of context model by using BPL, DL and SP.
b) Determine K.
K is an index (0 or 1) which represents whether MSB is reached or not at a given SP.

c) Determine N:
If (SP<3): N is the number of non-zero corresponding symbols in upper and left block. In contrast to all other types of context models, it depends on the context categories of different block types.
Else: N is the number of non-zero previous encoded three binary symbols (SP-1, SP-2, SP-3)
d) M: the number of coded non-zero symbols in upper DL.
From M, we exactly estimate the complexity of given block.
The range of M is from 0 to 64.

\[
\begin{array}{cccccccc}
BPL=0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
BPL=1, DL=0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
BPL=2, DL=1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
BPL=3, DL=2 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
BPL=4, DL=3 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
\end{array}
\]

Fig. 3. Context modeling by neighboring symbols

4.2 Binary arithmetic coding

In the binary arithmetic coding step, a bitstream is created from the probability of given symbol and the probability model is updated.

Binary arithmetic coding is based on the principle of recursive interval subdivision that involves the following elementary multiplication operation. Based on the probability of given symbol, the given interval is subdivided into two subintervals: one interval of range which is associated with the LPS, and the dual interval of range, which is assigned to the most probable symbol (MPS). Depending on the observed binary decision, either identified as the LPS or the MPS, the corresponding subinterval is then chosen as the new current interval.

![Flowchart of binary arithmetic coding](chart.png)

Fig. 4. Flowchart of binary arithmetic coding

5. Experimental results

In these experiments four codecs were used:

- **Codec 1**: MPEG-4 FGS codec.
- **Codec 2**: H.264-BPFGS codec.
- **Codec 3**: JSVM codec (version 2.0).
- **Codec 4**: H.264-CABAC-BPFGS.

**Codec 2 and Codec 4** were developed by authors, and both are based on the H.264/AVC Joint Model 9.5. In the experiments, two sets of experiments were performed: base layer and enhancement layer.

1) **Base Layer Test**: The main goal of this test is to evaluate the coding efficiency gain when the base layer uses H.264/AVC standard (in comparison with the Advanced Simple Profile used in the MPEG-4 FGS standard). In this test, the enhancement layer coding schemes of both codecs are the same, conformant with the MPEG-4 FGS standard.

2) **Enhancement Layer Test**: The main goal of this test is to compare performance of the H.264-CABAC-BPFGS enhancement layer coding scheme in comparison with JSVM and H.264-BPFGS. In these test conditions, a large number of sequences with different motion and texture characteristics were chosen to represent various types of content. For these tests, scalable bitstream was truncated and decoded at several points in the specified bitrate.

3) **Base Layer Test Result**: Figure 5 shows that for all combinations, the coding efficiency is higher when using the H.264/AVC standard in the base layer. As we expect, the use of the H.264/AVC encoder for the base layer allows a higher encoding efficiency for all scenarios with PSNR gains between about 2.5 and 4 dB.

![Graph showing PSNR gains](graph.png)

Fig. 5. MPEG-4 FGS and H.264 BPFGS

4) **Enhancement Layer Test Result**: For the enhancement layer test, we verify that the H.264-CABAC-BPFGS solution proposed in this paper has a coding efficiency superior to JSVM and MPEG-4 FGS. The main reason for the efficiency in enhancement coding is obtained from the accurate probability estimation scheme from the proposed context model which use the geometric information of given symbol such as BPL, DL, SP, and the correlation of neighboring symbol and upper bitplane layer. In addition, we define M which represents the number of coded non-zero symbols in upper DL. Hence, we can exactly estimate the complexity of the current block. Consequently, we obtain 15% bit rate
block. Consequently, we obtain 15% bit rate reduction for coding the same number of biplane in enhancement layer compare to MPEG-4 FGS.

Fig. 6. The efficiency of H.264-CABAC-BPFGS

Fig. 7. The efficiency of H.264-CABAC-BPFGS

6. Conclusions
In this paper a new solution for fine grain scalable SNR video coding scheme, namely H.264-CABAC-BPFGS is proposed. In H.264-CABAC-BPFGS, we adopt H.264/AVC standard for the base layer coding and propose newly designed binary arithmetic coding for the biplane coding in the enhancement. The experimental results show that the efficiency of the H.264/AVC for the base layer coding. We also verified that the proposed binary arithmetic coding scheme for enhancement layer shows a higher coding efficiency than the coding schemes for enhancement layer coding in JSVM and MPEG-4 FGS.

Acknowledgements
This work was supported by the Ministry of Information and Communication (MIC) through the Realistic Broadcasting Research Center (RBRC) at Gwangju Institute of Science and Technology (GIST).

References


