# A CONTEXT-BASED ERROR DETECTION STRATEGY INTO H.264/AVC CABAC

*Yang Li, Hongkai Xiong, Li Song, Songyu Yu*

Department of Electrical Engineering, Shanghai Jiao Tong University, P.R.China

## ABSTRACT

Various error control schemes have been addressed in wireless video stream transmission. By combining an adaptive binary arithmetic coding technique with context modeling, CABAC as a normative part of H.264/AVC has achieved a high degree of adaptation and redundancy reduction. However, error propagation still remains a problem because of the property of arithmetic coding. The presented scheme compares the various error detection methods, and proposes an efficient error detection technique based on CABAC semantics, which is achieved by inserting detective markers denoting by syntax elements. The misdetection probability versus stream size expansion can be easily handled. In addition, placements of markers can vary with regard to specific video content, thus efficiency within this scheme is enhanced. Comparison with other detection scheme is also presented.

## 1. INTRODUCTION

Of all modalities desirable for future wireless multimedia systems, motion video is the most demanding in terms of bit-rate and transmission constraints. In wireless channel, transmission errors range from single bit errors to burst errors or even intermittent loss of connection. Moreover, the compressed video signal is extremely vulnerable against these errors, because low bit-rate video coding schemes rely on inter-frame coding for high coding efficiency.

H.264/AVC, the advanced video coding standard, aims at providing enhanced compression performance and provision of a "network-friendly" video representation. Though high compression efficiency H.264/AVC offers, CABAC, one of the entropy methods, is vulnerable to errors because of the arithmetic coding engine it adopts. Therefore, error control method must be utilized to tackle these problems.

Error detection schemes analyzed in this paper are designed to suppress the error propagation caused by arithmetic coding. Boyd *et al.* [1] proposed continuous error detection into arithmetic coding (so-called "forbidden" symbol scheme), whose tradeoff between file redundancy and detective ability can be managed by a single reduction factor. The simplicity of implementation has induced much work on it [2]. Another error detection scheme widely applied is in the form of markers placed into the source encoder, which will be examined closely below. This technique is performed by placing the source symbol into blocks then inserting a marker to check errors. The redundancy needed for recovery is introduced in the form of markers before it is compressed by the entropy coder, as show in *Fig* 1. At the receiver side, a detection decoder is along with the entropy decoder for quick error detection. This scheme is suitable for transmitting long files over low bit error rate (BER) channels, and it provides assurance of reconstructing the original data, free from catastrophic errors. Several marker strategies are examined in [3].

This paper focuses on incorporating context-based error detection scheme by markers into the present advanced coding standard, H.264/AVC. We deduce the misdetection probability of markers in CABAC, and relative video stream size expansion. Marker strategy based on specific characteristic of video stream is also analyzed. Section II introduces necessary background information and incorporation of error detection by markers into CABAC. In Section III, experimental results are given to demonstrate detection ability of marker strategy.

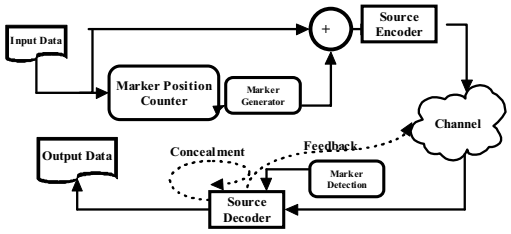


*Fig*.1 Marker strategy for error detection

## 2. CONTEXT-BASED ERROR DETECTION IN CABAC

### 2.1. CABAC Framework

Context-Based Adaptive Binary Arithmetic Coding (CABAC) [4] is one of the two entropy coding methods of the ITU/ISO/IEC standard for video coding, H.264/AVC. The encoding process consists of, at most, three elementary steps (*Fig*. 2): Binarization, Modeling and Binary arithmetic coding.

This work is supported by the grant of NSF No.60502033 and Shanghai NSF No.04ZR14082.

CABAC adopts arithmetic coding as its coding engine, one of the most important properties of which is the possibility to utilize a clean interface modeling and coding such that in the modeling stage, a model probability distribution is assigned to the given symbol. In CABAC, each probability model related to a given context index $\gamma$ is determined by a pair of two values, a 6-bit probability state index $\sigma_\gamma$ and the (binary) value $\varpi_\gamma$ of the most probable symbol (MPS).

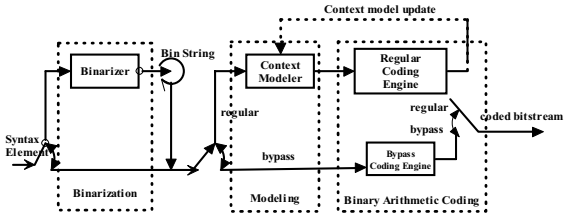


*Fig*.2 CABAC encoder block diagram

As for the probability estimation in CABAC, 64 representative probability values $p_\sigma \in [0.01875, 0.5]$ were derived for LPS by the following recursive equation:

$p_\sigma = \alpha \bullet p_{\sigma-1}$ for all $\sigma = 1,...,63$

where $\alpha = (\frac{0.01875}{0.5})^{1/63}$ and $p_0 = 0.5$

As a result of this design, each context model in CABAC can be completely determined by two parameters: its current estimation of the LPS probability, which in turn is characterized by an index $\sigma \in [0,63]$, and its value of MPS $\varpi$ being either 0 or 1. The update of probability states can be seen in Fig. 3.

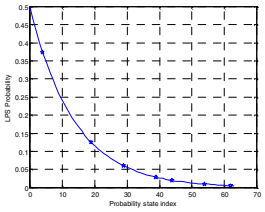


*Fig*. 3 LPS probability values

If a MPS occurs: $p_{new} = \max(\alpha \bullet p_{old}, p_{62})$

If a LPS occurs: $p_{new} = \alpha \bullet p_{old} + 1 - \alpha$

**2.2. Marker position selection**

Since syntax elements in CABAC can be distinguished by four basic types of context models, error propagation property of these elements varies. Intra prediction modes for chroma and motion vector differences are those syntax elements, which involve context templates with up to two neighboring syntax elements in the past of the current syntax elements to encode. Thus error within these elements will affect neighboring macroblocks spatially. The second type of context models is only defined for syntax element of mb_type and sub_mb_type. For this kind of context models, the values of prior coded bins are used for the choice of a model for a present bin. Error within these two elements can be categorized as temporal mistakes. All the above two types of context models are header information within the dashed rectangular in *Fig*. 4. Error in this part will induce the decoder devastation. In light of the coding mechanism of arithmetic coding, which is based on the principle of recursive interval subdivision, marker is placed at the end of the head information to detect former errors. To avoid unnecessary stream size augment, one marker is sufficient here for quick detection.

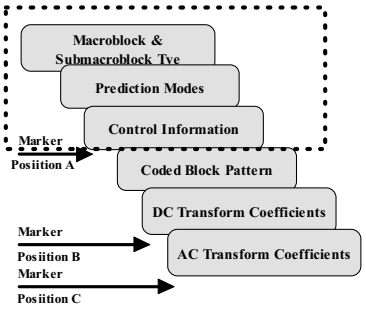


*Fig*. 4 Sketchy description on marker placement in CABAC

Types of context models applied to less important syntax elements are residual data information. A one-bit symbol coded block_flag and a binary-valued significance map are used to indicate the occurrence and the location of nonzero transform coefficients in a given block. They do not rely on past coded data, but on the position in the scanning path. Error on these bins will not propagate in terms of semantics. Context models for coding the nonzero transform coefficients are chosen based on the number of previously transmitted nonzero levels within the reverse scanning path. Modeling functions are specified that involve the evaluation of the accumulated number of encoded levels with a specific value prior to the current level bin to encode. If error within this range fails to be rectified by concealment, it will hamper the quality of streaming video at the receiver. Because various video sequences possess different characteristics, marker strategy can be more flexible here to cater for specific requirements. For sequences with abundant texture, more markers may be needed in coding AC relative coefficients to reinforce the detection scheme. Other critical bits are the EOB (End of block) symbols, loss of which will lead to catastrophic error propagation. Thus with regard to the particular video stream and specific requirement, marker in position C can be inserted closely after EOB or luma AC coefficients or both.

**2.3. Marker symbol selection**

Unlike other marker strategies [3], our scheme is strongly related to CABAC framework. Placements of markers need to be finely designed to suit this framework. To avoid overhead cost of H.264, marker is selected within syntax elements in CABAC, thus the probability of markers is on the curve of Fig. 3. Special markers other than syntax elements are discouraged in this scheme, for cost saving in terms of computation. Other marker strategy such as average marker or most frequency marker technique [3] does not utilized under CABAC framework with the same consideration. New binary context are designed for markers,

which is consistent with the CABAC model. The probabilities of Markers can vary in specific position to provide unequal error detection. For example, syntax elements in position A are more important, small probability markers are needed here to avoid misdetection. In less important positions, markers with relative high probability can be chosen, for sake of stream size saving.

**2.4. Theoretical analysis**

Since error control schemes are generally achieved by adding redundancy, the expansion of target stream size versus control efficiency needs to be estimated for measurement.

As stated above, in the previous work of Boyd *et al.* [1], an error detection scheme is proposed within arithmetic coding. The file redundancy and detective ability can be traded off by a reduction factor $\varepsilon$. To add $\Delta_f$% redundancy, choose $\varepsilon = \frac{1}{1+0.00693\Delta_f}$.

Differ from the general strategy on introducing a reduction factor; our detection scheme by using markers can give unequal error detection capability. Since precedence in CABAC syntax elements vary, a worst-case design would lead to a prohibitive amount of redundancy.

In our scheme, if a precise and quicker detection is needed, low frequency in *Fig.*3 can be chosen, thus means large probability state index. Video stream size will become considerable if rigor misdetection is needed. Any LPS probability index can be chosen with regard to practical needs.

With this marker strategy, special markers are chosen to be embedded in CABAC, whose probability is $p_{marker}$. If error occurs in the encoded data, the probability of misdetecting $p_{mis}$ is the probability of still having the marker at the specific location, which is $p_{marker}$. Clearly, the smaller the $p_{marker}$, the lower the $p_{mis}$. Since we use more than one marker in a single macroblock, $p_{mis} = p^t{}_{marker}$, where $t$ is the number of times the markers encoded into each macroblock.

However, the size of the stream increases. The probability $p_{SE_i}$ of certain syntax element is $N(SE_i)/L$, where $N(SE_i)$ is the total number of times $SE_i$ occurs in the stream, and L is the size of the stream. By adding a marker after each block of size $k_m$, the new probability of occurrence of each syntax element in CABAC not used as a marker is

$$p'(SE_i) = \frac{N(SE_i)}{L+tM} = \frac{p(SE_i)}{1+\frac{tM}{L}} \quad (1)$$

where M is the number of blocks in the file, that is $\sum_{m=1}^{M} k_m = L$

For marker $SE_n$, the new probability is

$$p'(SE_n) = \frac{N(SE_n)+tM}{L+tM} = \frac{p(SE_n)+\frac{tM}{L}}{1+\frac{tM}{L}} \quad (2)$$

Before adding markers, the compression ratio is given by $R \le \log_2 l / H(p)$. $l$ is the number of probability states of syntax elements in the file, in CABAC, $l = 63$. After adding a marker, this becomes $R' \le \log_2 l / H'(p)$. If the compression algorithm is effective enough, equalities can be used. Thus, the size of compression file after adding the marker is $\frac{L+tM}{R'} = \frac{(L+tM)H'(p)}{\log_2 l}$ (3)

The percentage of file expansion (Normalizing to $\frac{L}{\log_2 l}$) is

$$\Delta_f = \frac{\log_2 l}{L} \times (\frac{L+tM}{R'} - \frac{L}{R}) / H(p) =$$

$$\frac{1}{H(p)}[(1+\frac{tM}{L})\log(1+\frac{tM}{L}) + p(SE_n)\log p(SE_n)$$

$$-(p(SE_n)+\frac{tM}{L})\log(p(SE_n)+\frac{tM}{L})] \quad (4)$$

Here, we can infer that stream size expansion can be controlled by marker numbers and marker probabilities as well. Trade off can be made based on real situation.

## 3. EXPERIMENTAL RESULTS

As stated above, smaller misdetection probability will induce lager file size. Stream size expansion versus misdetection probability is shown in *Table.*1. Our experiment selects the LPS state index denoting by pentagram in *Fig.*2. Markers in this test are placed at Position A and Position C with reference to *Fig.* 4. "Forman" with QCIF resolution (176× 144) are encoded to produce the H.264/AVC test bitstream, JM10.1 is used here as test source coder. .

It is obvious that more markers placed behind different syntax elements will generate more precise detection. However, stream size will become large. Thus Marker can be chosen according to the specific application as well as video stream characteristics. In test sequence "Foreman", two markers are suggested to be implanted in position A and B respectively referring *Fig.* 4. In "Mobile" sequence, due to the high frequency information it possesses, one marker is inserted in position C as a complement.

*Fig.* 5 compares the bit-rate increasing on different sequences. Markers are placed at position A, B and C (*Fig.*4) in all sequence. Sequences which possess more abundant texture and complex motion are less vulnerable to bit-rate increasing, because marker information in them is miniature

comparing to the original dense motion or texture information.

*Table.*1 Stream size versus Misdetection probability.

| Marker Probability state index | Misdetection Probability | Stream Size expansion |
|---|---|---|
| 5 | 0.1205 | 1.92% |
| 20 | 0.0135 | 2.54% |
| 30 | 0.0031 | 4.03% |
| 40 | 7.29E-04 | 5.39% |
| 45 | 3.52E-04 | 6.15% |
| 55 | 8.17E-05 | 6.40% |
| 63 | 2.54E-05 | 10.32% |

To give a comprehensive demonstration of the detective ability in this scheme, we adopt the multipath RayLeigh Fading Channel to simulate the real wireless transmission environment whose multipath profile is Brazil C with 6 paths. Time selective maximum Doppler shift is 4 Hz. RS (204,188) is utilized as the channel codes. Considering the actual channel state, decoding BER is confined below $10^{-4}$. In *Table*.2, we compare two detective methods under the CABAC framework. *Scheme A* in Table.2 is marker strategy, comparing to *Scheme B* ("forbidden" symbol) [1]. Detected error location offset is defined to evaluate detective ability. The offset means the distance between the detected error location and the actual error location. Obviously, the smaller the offset is, the faster the detective speed it possesses. "Foreman" with QCIF resolution is used here as the test stream. Results indicate that nearly under all LPS probability states and decoding BER simulated, the marker strategy is superior to the reduction factor scheme in terms of detective speed. That's because *Scheme B* needs some time to narrow down the range in order to discover the former transmission errors.

## 4. CONCLUSTION

The proposed coding into CABAC uses context-based markers to determine errors, which demonstrates an efficient performance in terms of flexibility and detection speed. This context-based error detection scheme can be used in cooperation with channel error control to guarantee transmission quality. Error tracking scheme can also utilize marker strategy for video transmission, which is combined with intra/inter-mode switching at source side.

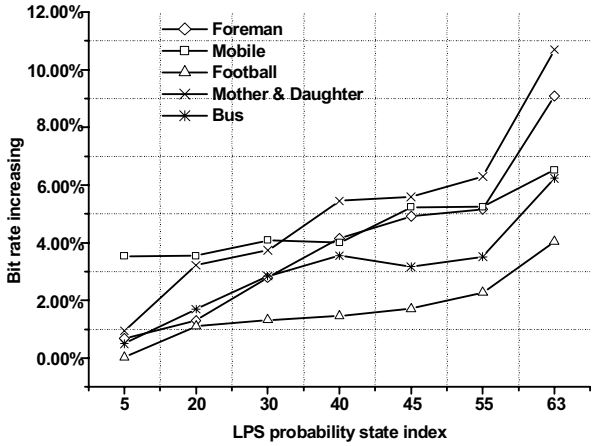


*Fig.* 5 Comparison on bit rate increasing

*Table*.2 Marker Detection ability versus other scheme

| LPS Probability state index | Decoding BER | Detected Error Location Offset Scheme A | Detected Error Location Offset Scheme B |
|---|---|---|---|
| 5 | E-04 | 3 Byte | 10 Byte |
| | E-05 | 7 Byte | 21 Byte |
| 20 | E-04 | 23 Byte | 21 Byte |
| | E-05 | 5 Byte | 8 Byte |
| 30 | E-04 | 53 Byte | 58 Byte |
| | E-05 | 1 Byte | 6 Byte |
| 40 | E-04 | 4 Byte | 6 Byte |
| | E-05 | 3 Byte | 31 Byte |
| 45 | E-04 | 181 Byte | 196 Byte |
| | E-05 | 2 Byte | 12 Byte |
| 55 | E-04 | 352 Byte | 354 Byte |
| | E-05 | 1 Byte | 28 Byte |
| 63 | E-04 | 551 Byte | 555 Byte |
| | E-05 | 4 Byte | 172 Byte |

## 6. REFERENCES


[1] C.Boyd, J.Cleary, S.Irvine, I.Rinsma-Melchert, and I.Witten, "Intergrating error detection into arithmetic coding," IEEE Trans. Commun., vol.45, pp.1-3, Jan. 1997.

[2] Jim Chou, Kannan Ramchandran, "Arithmetic coding-based continuous error detection for efficient ARQ-based image transmission," IEEE Journal on Selected Areas in Communications, vol.18, No.6, pp.861-867, June. 2000.

[3] George F.Elmasry, "Joint Lossless-Source and Channel Coding Using Automatic Repeat Request," IEEE Transactions on Communications, vol. 47, No. 7, pp.953-955, July. 1999.

[4] Detlev Marpe, Heiko Schwarz, Thomas Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," IEEE Transaction on Circuits and Systems for Video Technology, vol.13, No.7, pp.620-636, July. 2003.