TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Evgeny Belyaev

An Energy-efficient Live Video Coding and Communication over Unreliable Channels

Julkaisu 1283 • Publication 1283

Tampere 2015

Tampereen teknillinen yliopisto. Julkaisu 1283
Tampere University of Technology. Publication 1283

Evgeny Belyaev

An Energy-efficient Live Video Coding and Communication over Unreliable Channels

Thesis for the degree of Doctor of Science in Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB109, at Tampere University of Technology, on the 20th of February 2015, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2015

ISBN 978-952-15-3463-8 (printed)
ISBN 978-952-15-3471-3 (PDF)
ISSN 1459-2045

Abstract

In the field of multimedia communications there exist many important applications where live or real-time video data is captured by a camera, compressed and transmitted over the channel which can be very unreliable and, at the same time, computational resources or battery capacity of the transmission device are very limited. For example, such scenario holds for video transmission for space missions, vehicle-to-infrastructure video delivery, multimedia wireless sensor networks, wireless endoscopy, video coding on mobile phones, high definition wireless video surveillance and so on. Taking into account such restrictions, a development of efficient video coding techniques for these applications is a challenging problem.

The most popular video compression standards, such as H.264/AVC, are based on the hybrid video coding concept, which is very efficient when video encoding is performed off-line or non real-time and the pre-encoded video is played back. However, the high computational complexity of the encoding and the high sensitivity of the hybrid video bit stream to losses in the communication channel constitute a significant barrier of using these standards for the applications mentioned above.

In this thesis, as an alternative to the standards, a video coding based on three-dimensional discrete wavelet transform (3-D DWT) is considered as a candidate to provide a good trade-off between encoding efficiency, computational complexity and robustness to channel losses. Efficient tools are proposed to reduce the computational complexity of the 3-D DWT codec. These tools cover all levels of the codec's development such as adaptive binary arithmetic coding, bit-plane entropy coding, wavelet transform, packet loss protection based on error-correction codes and bit rate control. These tools can be implemented as end-to-end solution and directly used in real-life scenarios. The thesis provides theoretical, simulation and real-world results which show that the proposed 3-D DWT codec can be more preferable than the standards for live video coding and communication over highly unreliable channels and or in systems where the video encoding computational complexity or power consumption plays a critical role.

Preface

The research presented in this thesis has been carried out by the author during years 2011–2014 at the Department of Signal Processing, Tampere University of Technology, Finland. Partially, this work can be seen as a continuation of the results presented in the author's earlier "Candidate of Science" (which is a Russian equivalent for Ph.D) thesis, defended in 2009 at the Saint-Petersburg State University of Aerospace Instrumentation, Russia.

This work has been accomplished under the supervision of Prof. Karen Egiazarian and co-supervision of Prof. Moncef Gabbouj. I am extremely thankful to them for their highly professional guidance and careful review of the thesis.

I highly appreciate the joint work with my co-authors, Prof. Andrey Turlikov and Anton Veselov (Saint-Petersburg State University of Aerospace Instrumentation, Russia) and Prof. Kai Liu (Xidian University, China), based on which this thesis has been written. Many thanks go in particular to Dr. Alexey Vinel (Halmstad University, Sweden) for his help with a practical verification of the results of this thesis in the field of vehicle-to-vehicle video communications.

I acknowledge the financial support of Tampere Doctoral Program in Information Science and Engineering (TISE), Finland and TUT Graduate School. Without their support this research would have not been possible.

Special thanks go to Virve Larmila, Ulla Siltaloppi, Rotola-Pukkila Noora and Elina Orava for their friendly support in daily routines.

Last but not least, I wish to express my warmest thanks to my parents and my brother who inspired me towards scientific work.

Tampere, January 2015 *Evgeny Belyaev*

Contents

List of Publications

This thesis is composed of a summary part and five publications listed below as appendices. The publications are refereed to as [P1], [P2], *etc.* in the thesis.

[P1] E.Belyaev, A.Turlikov, K.Egiazarian and M.Gabbouj, "An efficient adaptive binary arithmetic coder with low memory requirement," *IEEE Journal of Selected Topics in Signal Processing. Special Issue on Video Coding: HEVC and beyond*, vol.7, iss.6, pp.1053–1061, 2013.

[P2] E.Belyaev, K.Egiazarian and M.Gabbouj, "A low-complexity bit-plane entropy coding and rate control for 3-D DWT based video coding," *IEEE Transactions on Multimedia*, vol.15, iss.8, pp.1786–1799, 2013.

[P3] E.Belyaev, K.Egiazarian, M.Gabbouj and K.Liu, "A Low-complexity joint source-channel video coding for 3-D DWT codec," *Journal of Communications*, vol.8, iss.12, pp.893–901, 2013.

[P4] E.Belyaev, A.Veselov, A.Turlikov and Kai Liu, "Complexity analysis of adaptive binary arithmetic coding software implementations," Proceedings of *The 11th International Conference on Next Generation Wired/Wireless Advanced Networking*, Saint-Petersburg, Russia, August 23–25, 2011.

[P5] E.Belyaev, K.Egiazarian and M.Gabbouj, "A real-time simulcast multi-view wavelet video coding based on skipping of spatial subbands," Proceedings of *8th International Symposium on Image and Signal Processing and Analysis*, Trieste, Italy, September 4–6, 2013.

List of Figures

List of Tables

List of Abbreviations

1-D DWT	One-dimensional Discrete Wavelet Transform
2-D DWT	Two-dimensional Discrete Wavelet Transform
3-D DWT	Three-dimensional Discrete Wavelet Transform
ABAC	Adaptive Binary Arithmetic Coder
ABRC	Adaptive Binary Range Coder
CABAC	Context-adaptive Binary Range Coder
GOF	Group of Frames
H.264/AVC	Video Coding Standard
H.264/SVC	Scalable Extension of H.264/AVC
M-coder	ABAC in H.264/AVC and H.265/HEVC Standards
RS codes	Reed-Solomon Error-correction Codes
SVC	Scalable Video Coding
ULP	Unequal Loss Protection
Y-PSNR	Peak Signal-to-Noise Ratio for Luminance Component

Chapter 1

Introduction

Video coding and transmission are core technologies used in numerous applications such as streaming, surveillance, conferencing, broadcasting and so on. There exists an important subset of these applications in which:

a) video capturing, video encoding, decoding, packet loss protection and playback should operate in real-time;

b) computational resources (or power consumption) for encoding are very restricted;

c) video communication channels can be very unreliable.

For example, the above requirements and application environments hold for video transmission for space missions [1], vehicle-to-infrastructure video delivery [2], underwater wireless video transmission [3], video transmission for wireless sensor networks [4], wireless endoscope capsules [5], video coding on mobile devices, high definition video surveillance, wireless home IPTV broadcast [6], 3D television and free viewpoint video [7, 8].

Additionally, in many cases, the transmitted video bit rate must be easily varied at the encoder side, and it should be easily truncated at any intermediate network node for adaptation to the channel bandwidth at the current link and or end-user display (see Figure 1.1). Taking into account the reasoning described above, a *scalable video coding* (SVC) is the most preferable compression method for such applications [9]. During the last decade, in a number of papers it was demonstrated that in a combination with unequal loss protection (ULP) of different video stream layers, SVC provides robust transmission (see, for example, [10, 11]).

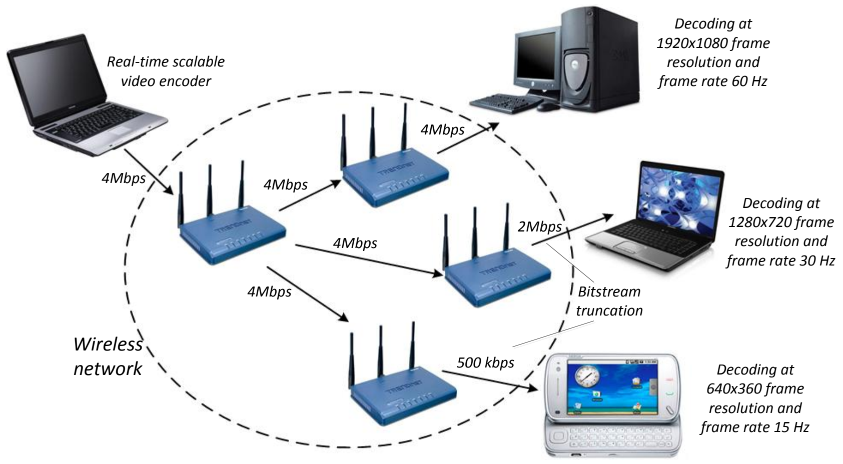

Real-time scalable video encoder
4Mbps
4Mbps
4Mbps
4Mbps
4Mbps
4Mbps
2Mbps
500 kbps
Decoding at 1920x1080 frame resolution and frame rate 60 Hz
Decoding at 1280x720 frame resolution and frame rate 30 Hz
Decoding at 640x360 frame resolution and frame rate 15 Hz
Bitstream truncation
Wireless network

Figure 1.1: A general structure of wireless video communication

Taking into account that bit errors are usually corrected at the lower network levels, only packet losses (or erasures) can happen at the application layer. Therefore, ULP can be easily implemented at the application layer using inter-packet error-correction codes, for example, Reed-Solomon (RS) codes without any modification of other network layers. In this case, the base video stream layer is protected using error-correction codes with a high redundancy level while the remaining layers are protected with a lower redundancy level or not protected at all.

A scalable extension of the H.264/AVC standard [12] (H.264/SVC), which is currently the most popular video coding approach, includes temporal, spatial and quality scalability and provides high compression efficiency due to block-based motion compensation and inter-layer prediction exploiting the video source temporal redundancy and redundancy between different layers. However, the high computational complexity required by H.264/SVC constitutes a significant barrier for certain important applications.

The high computational complexity of H.264/SVC, as well as other hybrid video coding algorithms (MPEG-1 [13], H.261 [14], MPEG-2/H.262 [15], H.263 [16], MPEG-4 Visual [17], H.265/HEVC [18]), are due to the following main reasons:

1. Block-based motion estimation and compensation require a lot of computations even if fast motion vector searching algorithms are used. Additional problem occurs in case of high resolution video sequences which requires high motion search radius for efficient coding in rate-distortion sense. This contradicts with a low complexity coding case which requires as small as possible motion search radius.

2. To avoid resynchronization with the decoder side, the motion compensation, as well as other encoder parts like intra-prediction, require backward loop in the encoding scheme. This loop includes inverse transform and quantization, deblocking filter and other extra calculations.

3. Macro-block coding mode selection as well as rate control also requires additional computations.

4. Distortion estimation caused by packet losses, error-correction coding and ULP optimization have relatively high computational complexity. Without ULP the video bit stream generated by H.264/SVC is very sensitive to packet losses and even 1% packet loss rate is enough to produce significant visual quality degradation when the reconstructed video becomes not authentic (new objects, which are not present in the original video, can appear).

As a result, for a live video transmission applications the H.264/SVC encoder should be significantly simplified, resulting in a decrease of its rate-distortion performance [19].

As an alternative to H.264/SVC encoders, distributive video coding (DVC) can be used. Based on Slepian-Wolf [20] and Wyner-Ziv [21] theories, DVC transfers the encoder complexity to the decoder side. Herewith, the decoder complexity is dramatically increased, and this complicates DVC usage in real-time applications. Moreover, many existing DVC encoders use a traditional codec for key frames compression (for example, H.264/AVC INTRA is embedded in Discover codec [22]) that requires the implementation of two encoders at the same platform. Finally, some papers (for example, [23]) show that DVC encoder power consumption is close to H.264/AVC INTER power consumption, when zero motion vectors are used for temporal redundancy elimination.

Finally, a video coding based on three-dimensional transforms such as three-dimensional discrete cosine transform (3-D DCT) [24, 25] and three-dimensional discrete wavelet transform (3-D DWT) [26–29] can be considered as candidates to provide a good trade-off between rate-distortion performance and computational complexity. Herewith, 3-D DWT codecs have more advantages, because they naturally support temporal, spatial and quality scalability, they can be implemented without multiplications and divisions, they do not cause blocking artifacts and they can easily generate a progressive bit stream which simplifies the estimation of end-to-end distortion due to packet losses as well as optimal selection of ULP mode.

During the last decade, several 3-D DWT approaches have been proposed such as 3-D set partitioning in hierarchical trees (3-D SPIHT) [27], 3-D embedded subband coding with optimal truncation (3-D ESCOT) [28], embedded video coding using zeroblocks of motion compensated 3-D subband/wavelet coefficients (MC-EZBC) [29] and so on. Due to intensive investigation of Motion Compensated Temporal Filtering (MCTF), currently 3-D DWT schemes have a comparable rate-distortion performance with the H.264/SVC [30]. However these schemes require also high computational resources. In [6], a real-time 3-D run-length wavelet video encoder has been proposed, which, however, does not include a rate control algorithm which is a key component of any video codec. In [31] and [32], rate control algorithms based on an extension of the Embedded Block Coding with Optimized Truncation (EBCOT) [33] were proposed. These rate controllers minimize the distortion for a given bit rate budget, but require a lot of computations due to the use of lossless compression of wavelet subbands requiring Lagrange multiplier selection. Other works propose rate control algorithms based on heuristics and additional video source characteristics (see, for example, [34]) and also require extra computations.

Thus, a development of efficient low-complexity scalable video coders with joint ULP and rate control is an important and challenging problem.

1.1 Objective and Scope of the Research

The main objective of this research is the development of efficient tools to reduce the computational complexity of joint video compression, unequal packet loss protection and rate control for video coding scheme based on 3-D DWT. The goal of this thesis is to show that the proposed 3-D DWT codec can be more preferable than the state-of-the-art scalable video codecs, such as H.264/SVC and 3-D SPIHT, in the case of live video transmission over highly unreliable channels and or in systems where the video encoding computational complexity or power consumption plays a critical role.

1.2 Summary of the Contributions

1.2.1 Efficient Adaptive Binary Arithmetic and Range Coding

Efficient look-up table-free adaptive binary arithmetic coder (ABAC) and adaptive binary range coder (ABRC) are proposed (**publications [P1],[P4]**). In comparison with the M-coder (ABRC from H.264/AVC and H.265/HEVC standards) they have the following advantages:

1. The proposed ABAC and ABRC do not use any look-up tables allowing a reduction in entropy encoder memory consumption (6-20% for H.264/AVC) and (32-42% for H.265/HEVC).
2. The proposed ABAC and ABRC provide a simple mechanism to control the trade-off between the speed of probability adaptation and the precision of probability estimation.
3. The proposed ABAC and ABRC do not require any changes in the context-modeling and can be easily embedded for performance improvement in any compression scheme which is based on binary arithmetic coding.
4. The proposed ABAC provides bit rate savings from 0.5 to 2.3% for H.264/AVC standard and from 0.6 to 3.6% for H.265/HEVC standard on average.
5. The proposed ABAC has a comparable computational complexity to the M-coder, while the proposed ABRC is up to 34% less complex than both ABAC and the M-coder.

1.2.2 Bit-plane Entropy Coding and Skipping of Wavelet Subbands

Efficient tools are proposed to decrease the computational complexity of the scalable video coding scheme based on 3-D DWT (**publications [P2]**,**[P5]**):

1. An efficient bit-plane entropy coder based on two cores: very simple zero-run coding based on Levenstein codes for low entropy contexts and ABRC for the remaining contexts is proposed. Simulation results show that the proposed combined bit-plane entropy coder is around 2 times less complex than the traditional one.

2. A rate-distortion efficient parent-child tree based subband skipping criterion is proposed. It allows skipping the 2-D DWT and bit-plane entropy encoding so that the entropy encoder complexity as well as the transform complexity are reduced with increasing of the compression ratio.

3. Simulation results show that 3-D DWT codec with the proposed bit-plane entropy encoder and subband skipping has a much lower computational complexity (from 2 to 6 times) for the same quality level compared to the H.264/AVC standard in the low complexity mode.

1.2.3 Joint Unequal Loss Protection and Rate Control

Efficient low-complexity error-resilient, unequal packet loss protection and rate control for video codec based on 3-D DWT are proposed (**publication [P3]**). In comparison with H.264/SVC, the proposed 3-D DWT codec has the following advantages:

1. It is much less sensitive to packet losses and provides robust video transmission and authentic reconstruction for 1–5% packet loss rates without any packet loss protection.

2. 3-D DWT codec equipped with the proposed ULP and rate control demonstrates better performance for high packet loss rates.

3. The computational complexity of 3-D DWT encoder with ULP and rate control is up to 6 times less compared to the H.264/AVC standard in the low complexity mode.

1.2.4 Author's Contributions

The author's contributions to low-complexity joint video compression and packet loss protection in 3-D DWT codec are included in the publications, denoted as [P1],[P2],...,[P5]. The author is the main author of these publications. He has been the main contributer in proposing the methods, carrying out the simulations and writing the publications.

1.3 Thesis Outline

This thesis includes two parts: an introduction composed of five chapters followed by an appendix containing five publications. Chapter 2 is dedicated to adaptive binary arithmetic and range coding. Chapter 3 presents bit-plane entropy coding and skipping of wavelet subbands. Chapter 4 introduces joint unequal loss protection and rate control, while Chapter 5 concludes the thesis.

Chapter 2

Adaptive Binary Arithmetic and Range Coding

2.1 Introduction

This chapter is dedicated to adaptive binary arithmetic coding (ABAC) which is a key component of Context-adaptive binary arithmetic coding (CABAC) used in most common image and video compression standards such as JPEG [37], JPEG2000 [35], H.264/AVC [12] and H.265/HEVC [38]. CABAC first divides the input data into several non-stationary binary sources using context modeling. Then, each binary source is compressed by ABAC according to the probability estimation. Taking into account that the vast majority of the output bit stream (excluding headers) is generated by ABAC, the overall compression performance and computational complexity of a video coding system highly depends on its efficiency. Therefore, ABAC with a lower computation complexity, a smaller memory footprint, and a higher compression efficiency remains an important challenge.

The rest of the chapter is organized as follows. Section 2.2 reviews the integer implementation of binary arithmetic coding, the probability estimation for binary sources and the M-coder [39], which is the current state-of-the-art ABAC implementation used in the H.264/AVC and H.265/HEVC standards. Sections 2.3 and 2.4 introduce novel efficient ABAC and ABRC algorithms proposed in the thesis. Comparative results of the proposed coders with the M-coder are presented in Section 2.5. Conclusions are drawn in Section 2.6.

2.2 State-of-the-art ABAC Implementations

2.2.1 Integer Implementation of Binary Arithmetic Coding

Let us consider a stationary discrete memoryless binary source with p denoting the probability of ones. In a binary arithmetic encoding codeword for a binary sequence $\mathbf{x}^N = \{x_1, x_2, ..., x_N\}$, $x_t \in \{0, 1\}$ is represented as $\lceil -\log_2 P(\mathbf{x}^N) + 1 \rceil$ bits of a number

$$Q(\mathbf{x}^N) + P(\mathbf{x}^N)/2, \tag{2.1}$$

where $P(\mathbf{x}^N)$ and $Q(\mathbf{x}^N)$ are the probability and the cumulative probability of a sequence $\mathbf{x}^N$, respectively, which can be calculated by the recurrent relations:
If $x_t = 0$, then

$$\begin{cases} Q(\mathbf{x}^t) \leftarrow Q(\mathbf{x}^{t-1}) \\ P(\mathbf{x}^t) \leftarrow P(\mathbf{x}^{t-1})(1-p), \end{cases}$$

if $x_t = 1$, then

$$\begin{cases} Q(\mathbf{x}^t) \leftarrow Q(\mathbf{x}^{t-1}) + P(\mathbf{x}^{t-1})(1-p) \\ P(\mathbf{x}^t) \leftarrow P(\mathbf{x}^{t-1})p. \end{cases}$$

In this thesis we use “$\leftarrow$“ as the assignment operation, “$\ll$“ and “$\gg$“ as left and right arithmetic shift, and “!“ as bitwise NOT operation.

An integer implementation of an arithmetic encoder is based on two registers: L and R size of b bits (see Algorithm 1). Register L corresponds to $Q(\mathbf{x}^N)$ and register R corresponds to $P(\mathbf{x}^N)$. The precision required to represent registers L and R grows with the increase of N. In order to decrease the coding latency and avoid registers underflow, the *renormalization* procedure is used for each output symbol (see Algorithm 2).

Algorithm 1 : Binary symbol x_t encoding procedure

```
1: $T \leftarrow R \times p$
2: $T \leftarrow \max(1, T)$
3: $R \leftarrow R - T$
4: if $x_t$ =1 then
5:     $L \leftarrow L + R$
6:     $R \leftarrow T$
7: end if
8: call Renormalization procedure
```

Algorithm 2 : Renormalization procedure

```
1: while $R < 2^{b-2}$ do
2:   if $L \geq 2^{b-1}$ then
3:     $bits\_plus\_follow(1)$
4:     $L \leftarrow L - 2^{b-1}$
5:   else if $L < 2^{b-2}$ then
6:     $bits\_plus\_follow(0)$
7:   else
8:     $bits\_to\_follow \leftarrow bits\_to\_follow + 1$
9:     $L \leftarrow L - 2^{b-2}$
10:  end if
11:  $L \leftarrow L \ll 1$
12:  $R \leftarrow R \ll 1$
13: end while
```

2.2.2 Probability Estimation for Binary Sources

In real applications the probability of ones is unknown. In this case for an input binary symbol x_t the probability estimation of ones $\hat{p}_t$ is calculated and used in line 1 of Algorithm 1 instead of p. One of the well known probability estimation algorithms is based on a *sliding window* concept. The probability of a source symbol is estimated by analyzing the content of a special buffer [46]. This buffer keeps W previously encoded symbols, where W is the length of the buffer. After the encoding of each symbol the buffer's content is shifted by one position, a new symbol is written to the free cell and the earliest symbol in the buffer is erased.

For the binary sources, the probability of ones is estimated by the Krichevsky-Trofimov formula [47]:

$$\hat{p}_{t+1} = \frac{s_t + 0.5}{W + 1}, \tag{2.2}$$

where s_t is the number of ones in the window before encoding the symbol with the index t.

The advantage of using a sliding window is the possibility of a more accurate evaluation of the source statistics and a fast adaptation to changing statistics. However, the window has to be stored in the encoder and the decoder memory, which is a serious drawback of this algorithm. To avoid this, the *Imaginary Sliding Window* technique (ISW) was proposed [46]. The ISW technique does not require storing the content of the window, and instead, it estimates symbols count from the source alphabet stored in the window.

Let us consider the ISW method for a binary source. Define $x_t \in \{0,1\}$ as a source input symbol with index t, $y_t \in \{0,1\}$ as a symbol deleted from the window after adding x_t. Suppose at every time instance a symbol in a random position is erased from the window instead of the last one. Then the number of ones in the window is recalculated by the following recurrent randomized procedure.

Step 1. Delete a random symbol from the window

$$s_{t+1} \leftarrow s_t - y_t, \tag{2.3}$$

where y_t is a random value generated with probabilities

$$\begin{cases} Pr\{y_t = 1\} = \dfrac{s_t}{W}, \\ Pr\{y_t = 0\} = 1 - \dfrac{s_t}{W}. \end{cases} \tag{2.4}$$

Step 2. Add a new symbol from the source

$$s_{t+1} \leftarrow s_{t+1} + x_t. \tag{2.5}$$

For the implementation of the ISW algorithm, a random variable must be generated. This random variable should take the same values at the corresponding steps of the encoder and the decoder. However, there is a way to avoid generating a random variable [40]. At step 1 of the algorithm let us replace a random value y_t with its probabilistic average. Then the rule for recalculating the number of ones after encoding of each symbol x_t can be presented in two steps.

Step 1. Delete an average number of ones from the window

$$s_{t+1} \leftarrow s_t - \frac{s_t}{W}. \tag{2.6}$$

Step 2. Add a new symbol from the source

$$s_{t+1} \leftarrow s_{t+1} + x_t. \tag{2.7}$$

By combining (2.6) and (2.7), the final rule for recalculating the number of ones can be given as follows:

$$s_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s_t + x_t. \tag{2.8}$$

Equation (2.8) corresponds to the following probability estimation rule:

$$p_{t+1} = \left(1 - \frac{1}{W}\right) \cdot p_t + \frac{1}{W} x_t. \tag{2.9}$$

2.2.3 M-coder used in H.264/AVC and H.265/HEVC Standards

One way for implementation of probability estimation can be based on the *state machine* approach. Each state of this machine corresponds to some probability value. Transition from state to state is defined by the value of the input symbol. This approach does not require multiplications or divisions for probability calculation. In addition, the fixed set of states allows to implement the interval division part of the arithmetic coder without multiplications.

For example, let us consider a state machine based probability estimation in state-of-the-art M-coder [39] used in H.264/AVC and H.265/HEVC standards. In the M-coder input symbols are divided into two types: Most Probable Symbols (MPS) and Least Probable Symbols (LPS). State machine contains 64 states. Each state s defines probability estimation for Least Probable Symbol. Set of probability values $\{\hat{p}_0, \hat{p}_1, ..., \hat{p}_{63}\}$ is defined as:

$$\begin{cases} \hat{p}_s = (1-\gamma)\hat{p}_{s-1}, \text{ where } s = 1, ..., 63, \hat{p}_0 = 0.5, \\ \gamma = 1 - \left(\dfrac{\hat{p}_{min}}{0.5}\right)^{\frac{1}{63}}, \hat{p}_{min} = 0.01875. \end{cases} \tag{2.10}$$

Probability estimation for symbol x_{t+1} is calculated as

$$\hat{p}_{t+1} = \begin{cases} (1-\gamma)\hat{p}_t + \gamma, \text{ if } x_t = \text{LPS}, \\ \max\{(1-\gamma)\hat{p}_t, \hat{p}_{min}\}, \text{ if } x_t = \text{MPS}, \end{cases} \tag{2.11}$$

and implemented by using tables *TransStateLPS*$[s]$ and *TransStateMPS*$[s]$ which contain number of the next probabilities after compression of the current symbol. It is important to notice that if we define $\gamma = 1/W$, then it is easy to see that the probability estimation rule (2.11) is based on rule (2.9).

To remove the multiplication in line 1 of Algorithm 1, the M-coder uses its approximation [48]. After the renormalization procedure in Algorithm 2, register R satisfies the following inequality:

$$\frac{1}{2} 2^{b-1} \le R < 2^{b-1}. \tag{2.12}$$

From (2.12) it follows that a multiplication can be approximated in the following way:

$$T = R \times \hat{p}_t \approx \alpha 2^{b-1} \times \hat{p}_t, \tag{2.13}$$

where $\alpha \in [\frac{1}{2}, ..., 1)$. To improve the precision of the approximation, the M-coder quantizes the interval $[\frac{1}{2}2^{b-1}; 2^{b-1})$ uniformly to four cells. Then each multiplication of the corresponding probability $\hat{p}_s$ and the interval cell with index $\Delta \in \{0, 1, 2, 3\}$ are stored in two-dimensional table $\mathit{TabRangeLPS}[s][\Delta]$ which contains 64×4 values.

Thus, the ABAC algorithm in H.264/AVC and H.265/HEVC standards is implemented in the following way (see Algorithm 3).

Algorithm 3 : Binary symbol x_t encoding procedure

```
1: $\Delta \leftarrow (R - 2^{b-2}) \gg (b-4)$
2: $T \leftarrow \mathit{TabRangeLPS}[s][\Delta]$
3: $R \leftarrow R - T$
4: if $x_i \neq$ MPS then
5:     $L \leftarrow L{+}R$
6:     $R \leftarrow T$
7:     if $s = 0$ then
8:         MPS $\leftarrow$!MPS;
9:     end if
10:    $s \leftarrow \mathit{TransStateLPS}[s]$
11: else
12:    $s \leftarrow \mathit{TransStateMPS}[s]$
13: end if
14: call Renormalization procedure
```

2.3 The Proposed Adaptive Binary Arithmetic Coder

Let us multiply both sides of (2.8) by $\alpha 2^{b-1}$:

$$s'_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s'_t + \alpha 2^{b-1} x_t, \tag{2.14}$$

where $s'_t = \alpha 2^{b-1} s_t$. After integer rounding of equation (2.14), we obtain

$$s'_{t+1} = \begin{cases} s'_t + \left\lfloor \dfrac{\alpha 2^{b-1} 2^w - s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 1 \\ \\ s'_t - \left\lfloor \dfrac{s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 0, \end{cases} \tag{2.15}$$

Taking into account (2.12) and (2.13)

$$T = R \times \hat{p}_t \approx \alpha 2^{b-1} \times \hat{p}_t = \frac{s'_t}{2^w}. \tag{2.16}$$

To improve a precision of the approximation (as in the M-coder) we quantize the interval $[\frac{1}{2}2^{b-1}; 2^{b-1})$ to four points:

$$\left\{\frac{9}{16}2^{b-1}, \frac{11}{16}2^{b-1}, \frac{13}{16}2^{b-1}, \frac{15}{16}2^{b-1}\right\}. \tag{2.17}$$

To implement this, we first calculate a state s'_t using (2.15) for $\alpha = \frac{9}{16}$. Then we approximate the multiplication in the following way:

$$T = R \times \hat{p}_t \approx \frac{s'_t + \Delta \times \frac{1}{4}s'_t}{2^w}, \text{ where } \Delta = \frac{R - 2^{b-2}}{2^{b-4}}. \tag{2.18}$$

Taking into account that the approximation of the multiplication is correct for $\hat{p}_t < \frac{2}{3}$ [48], we should work with $\hat{p}_t \in [0, .., 0.5]$ and use the Most Probable Symbol and the Least Probable Symbol. In our case, the MPS value should be changed if

$$\hat{p}_t = \frac{s'_t}{2^w} \frac{1}{\alpha 2^{b-1}} > 0.5, \tag{2.19}$$

or

$$s'_t > \alpha 2^{b-2} 2^w. \tag{2.20}$$

Thus, taking into account (2.15), (2.18) and (2.20), an ABAC is proposed in the following way (see Algorithm 4):

Algorithm 4 : Binary symbol x_i encoding procedure

```
1: $\Delta \leftarrow (R - 2^{b-2}) \gg (b-4)$
2: $T \leftarrow (s + \Delta \times (s \gg 2)) \gg w$
3: $T \leftarrow \max(1, T)$
4: $R \leftarrow R - T$
5: if $x_i \neq$ MPS then
6:     $L \leftarrow L+R$
7:     $R \leftarrow T$
8:     $s \leftarrow s + ((\alpha 2^{b-1} 2^w - s + 2^{w-1}) \gg w)$
9:     if $s > \alpha 2^{b-2} 2^w$ then
10:        MPS $\leftarrow$!MPS;
11:        $s \leftarrow \alpha 2^{b-2} 2^w$;
12:    end if
13: else
14:    $s \leftarrow s - ((s + 2^{w-1}) \gg w)$
15: end if
16: call Renormalization procedure
```

Since $\Delta \in \{0, 1, 2, 3\}$, a multiplication in line 2 of Algorithm 4 can be implemented based on conditional and addition operations.

2.4 The Proposed Adaptive Binary Range Coder

As an alternative to arithmetic coders, *range coders* use bytes as output bit stream element and do byte renormalization at a time [41–44]. Byte renormalization from [42] is presented in Algorithm 5.

Algorithm 5 : Byte renormalization

```
1: while $(L \oplus (L+R)) < 2^{24}$ or $R < 2^{16}$ do
2:     if $R < 2^{16} \wedge (L \oplus (L+R)) \geq 2^{24}$ then
3:         $R \leftarrow (!L + 1) \wedge (2^{16} - 1)$
4:     end if
5:     PUTBYTE $(L \gg 24)$
6:     $R \leftarrow R \ll 8$
7:     $L \leftarrow L \ll 8$
8: end while
```

In range coders the interval division part cannot be well approximated as it is shown in (2.13), because after renormalization R value belongs to the wide interval $[2^8, ..., 2^{24}]$. Therefore, in this thesis it is proposed to use interval division part with multiplication using probability estimation by virtual sliding window [40]. Let us multiply both sides of (2.8) by W:

$$s'_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s'_t + W x_t, \tag{2.21}$$

where $s'_t = W s_t$. Let us define $W = 2^w$, where w is an integer positive value. After integer rounding of equation (2.21), we obtain

$$s'_{t+1} = \begin{cases} s'_t + \left\lfloor \dfrac{2^{2w} - s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 1 \\ \\ s'_t - \left\lfloor \dfrac{s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 0, \end{cases} \tag{2.22}$$

and the probability of ones is calculated as

$$\hat{p}_t = \frac{s'_t}{2^{2w}}. \tag{2.23}$$

Thus, the adaptive binary range coder (ABRC) algorithm based on virtual sliding window is presented in Algorithm 6.

Algorithm 6 : Binary symbol x_t encoding procedure

```
1: $T \leftarrow (R \times s) \gg (2w)$
2: $T \leftarrow \max(1, T)$
3: $R \leftarrow R - T$
4: if $x_t$ =1 then
5:     $L \leftarrow L+R$
6:     $R \leftarrow T$
7:     $s \leftarrow s + ((2^{2w} - s + 2^{w-1}) \gg w)$
8: else
9:     $s \leftarrow s - ((s + 2^{w-1}) \gg w)$
10: end if
11: call Renormalization procedure
```

From (2.22) it follows that if the initial value s'_0 satisfies

$$2^{w-1} - 1 \le s'_0 \le 2^{2w} - 2^{w-1} + 1, \tag{2.24}$$

then during all operational time the minimum possible probability estimation of ones is

$$\hat{p}_{min} = \frac{s'_{min}}{2^{2w}} = \frac{2^{w-1} - 1}{2^{2w}}, \tag{2.25}$$

and maximum possible probability estimation of ones is

$$\hat{p}_{max} = \frac{s'_{max}}{2^{2w}} = \frac{2^{2w} - 2^{w-1} + 1}{2^{2w}}. \tag{2.26}$$

Following properties of the arithmetic encoder, the maximum possible output bit stream size per binary symbol (when $x_t = 1$ and $s'_t = 2^{w-1} - 1$ or when $x_t = 0$ and $s'_t = 2^{2w} - 2^{w-1} + 1$) can be estimated as

$$\lceil \max\{\log_2(\hat{p}_{min}), \log_2(1 - \hat{p}_{max})\} \rceil . \tag{2.27}$$

TABLE 2.1: Maximum bit stream size per binary symbol

w	4	5	6	7	8
Size, bits	6	7	8	9	10

Table 2.1 shows the maximum possible output bit stream size per input binary symbol depending on the window size. One can see that if we select $w \leq 6$, this value is less than one byte. It means, that during renormalization process, not more than one byte can be output without encoder/decoder overloading. Therefore, we can simplify the renormalization by replacing *while* by *if* in Algorithm 5 as it is shown in Algorithm 7.

Algorithm 7 : Proposed byte renormalization

```
1: if $(L \oplus (L + R)) < 2^{24}$ then
2:     PUTBYTE $(L \gg 24)$
3:     $R \leftarrow R \ll 8$
4:     $L \leftarrow L \ll 8$
5: else if $R < 2^{16}$ then
6:     $R \leftarrow (!L + 1) \wedge (2^{16} - 1)$
7:     PUTBYTE $(L \gg 24)$
8:     $R \leftarrow R \ll 8$
9:     $L \leftarrow L \ll 8$
10: end if
```

2.5 Comparative Results

2.5.1 Computation Complexity

From Algorithms 1, 3, 4 and 5 it follows, that the number of operations in the interval division part of an arithmetic coder is directly proportional to the number of input binary symbols. On the other hand, the number of times steps in lines 2–12 are used is proportional to the number of bits in the output bit stream (see Algorithm 2). Let us define N as a number of the input binary symbols, R as a size of the output bit stream. Then, the complexity per input symbol for the binary arithmetic coder C_A can be written as follows:

$$C_A = \frac{\alpha_A \cdot N + \beta_A \cdot R}{N}, \tag{2.28}$$

where α_A is the computation complexity of the interval division part per input binary symbol, β_A is the computation complexity of the renormalization part per output binary symbol. For arithmetic coder the output bit stream size

$$R = N \cdot \Big(h(p) + r(p)\Big), \tag{2.29}$$

where $h(p)$ is an entropy of binary memoryless source with probability of ones p, $r(p)$ is an coding redundancy. Therefore, the complexity per input symbol is a linear function of the source entropy and coding redundancy:

$$C_A(p) = \alpha_A + \beta_A \cdot \Big(h(p) + r(p)\Big). \tag{2.30}$$

From (2.30) it follows that for zero-entropy source ($h(p) = 0$) the complexity $C(p)$ is the lowest possible and is mainly determined by the complexity of the division part. With an increase of $h(p)$ the renormalization part is used more and more often; therefore, the complexity also increases and reaches the maximum value when $h(p) = 1$. Moreover, from the model (2.30) it follows, that if everything else being equal, the compression scheme which has a higher coding redundancy has a higher complexity.

Using the reasoning described above, the complexity for binary range coder can be written as:

$$C_R(p) = \alpha_R + \beta_R \cdot \frac{1}{8} \cdot \Big(h(p) + r(p)\Big), \tag{2.31}$$

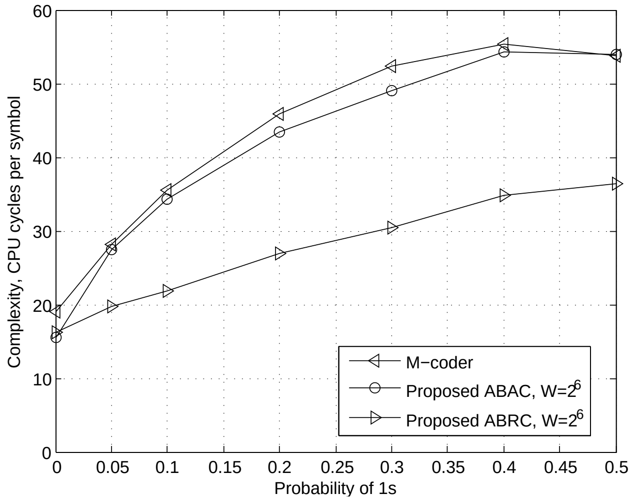


FIGURE 2.1: CPU cycles per symbol of Processor Intel Core i3M 2.1 GHz for the M-coder and the proposed coders

where α_R is the computation complexity of the interval division part per input binary symbol, β_R is the computation complexity of the renormalization part per output byte. For simplification let us assume that $\alpha_A \approx \beta_A \approx \alpha_R \approx \beta_R$, then

$$\frac{C_A(p) - C_R(p)}{C_A(p)} \approx \frac{\frac{7}{8} \cdot h(p)}{1 + h(p)} \in [0, ..., 0.4375]\,. \tag{2.32}$$

Figure 2.1 shows the complexity for the M-coder, the proposed ABAC and the proposed ABRC. In this thesis the complexity is measured in average CPU cycles per input symbol. For the measurements both coders were implemented as separate software programs which include the encoding of a binary string only. For each probability p we have generated $N = 10^8$ input binary symbols and, then, the CPU cycles of Processor Intel Core i3M 2.1 GHz was measured by the *Average CPU Cycles* software[1]. One can see that the complexities of the M-coder and the proposed ABAC are comparable, while the proposed ABRC is up to 34% less complex, that is close to the model (2.32).

[1]`http://user.tninet.se/~jad615g/averagecpu/`

2.5.2 Memory Consumption

Table 2.2 shows memory consumption needed for full implementation of Context-adaptive binary arithmetic coder depending on the number of binary contexts n_c in the video compression algorithm. One can see that the proposed ABAC and ABRC does not need memory for look-up tables. One the other side, they needs more bits for representation of the probability estimation (state). Therefore, the overall memory consumption savings is depends on n_c. Table 2.3 shows the overall memory consumption for CABAC depending on video compression algorithm. Taking into account that H.265/HEVC uses less number of binary contexts [45], the memory consumption savings caused by replacing the M-coder by the proposed ABAC or ABRC in H.265/HEVC is higher than for H.264/AVC.

TABLE 2.2: CABAC memory consumption for given number of contexts n_c, bits

	M-coder	Proposed ABAC, $W = 2^6$	Proposed ABRC, $W = 2^6$
Range look-up table	$64 \times 4 \times 8 = 2048$	0	0
Transition look-up tables	$2 \times 64 \times 6 = 768$	0	0
Initialization tables	$6 \times n_c$	$6 \times n_c$	$6 \times n_c$
State	$6 \times n_c$	$14 \times n_c$	$12 \times n_c$
MPS/LPS	$1 \times n_c$	$1 \times n_c$	0

TABLE 2.3: CABAC memory consumption for different codecs, bits

	n_c	M-coder	Proposed ABAC, $W = 2^6$	Proposed ABRC, $W = 2^6$
H.264/AVC	299	6703	6279 (-6%)	5382 (-20%)
H.265/HEVC	154	4818	3234 (-32%)	2772 (-42%)
3-D DWT [**P2**]	12	2972	252 (-91%)	216 (-92%)

2.5.3 Bit rate Savings for H.264/AVC and H.265/HEVC Standards

For practical experiments the proposed ABAC was embedded into the JM codec v.12.1 [49] which is the reference software of the H.264/AVC video coding standard and into the HM 3.0 reference software [18] of the H.265/HEVC video coding standard. JM codec was used with the Main profile, while HM codec was running with the low-delay configuration.

The initial state for each binary context was calculated as:

$$s_0 = \max\left\{2^{w-1} - 1, \left\lfloor \alpha 2^{b-1} 2^w \hat{p}_0 + 0.5 \right\rfloor\right\} \tag{2.33}$$

where $\hat{p}_0$ is the initial probability predefined in the standards.

In fact, the predefined initial probability can differ significantly from the best initial probability for the current binary source from the compression efficiency point of view. Therefore, it is important to use a high speed of probability adaptation at the initial stage of the coding, especially in the case of compression of short binary sequences. Then it is better to switch to a probability estimation with a lower adaptation speed, but with a better precision of the probability estimation. This approach is adopted in JPEG2000 [35] utilizing three adaptation mechanisms, which are embedded into one state machine. It has also been used in [40] by applying Krichevsky-Trofimov formula at the initial stage of the coding. However, in both cases, an additional table or a division operation is needed.

In this thesis a similar idea is implemented in the following way. First, for a binary sequence compression a small window length $W = 2^w$ is used. After the compression of n_1 binary symbols the window length and the state s are increasing two times to $W = 2^{w+1}$ and $2s$. Then, after the compression of n_2 binary symbols the window length and the state are increased two times again, and so on, until the maximum window length is achieved. In this case a high speed of probability adaptation is achieved by setting small window lengths at the initial stage of coding and, then, a high probability estimation precision is achieved by setting longer window lengths.

Bit rate savings in percentage related to the M-coder for different quantization parameters are presented in Tables 2.4–2.5. In all cases the reconstructed video for the corresponding quantization parameter (QP) is identical for the M-coder and for the proposed coder. Practical results were obtained for the first 60 frames of the test video sequences [50, 51] with different frame resolutions: 352×288

(“Foreman“, “Mobile“, “Akyio“, “Mother Daughter“), 640×480 (“Vassar“, “Ballroom“), 704×576 (“City“, “Crew“) and 1920×1080 (“Pedestrian Area“, “Rush Hour“, “Station“, “Tractor“). In the experiments, in case of H.264/AVC, for all contexts an initial window length is $W = 2^4$, the maximum window length is $W = 2^6$ and the values $n_1 = 24$ and $n_2 = 48$. In case of H.265/HEVC, an initial window length is $W = 2^3$, the maximum window length is $W = 2^6$ and the values $n_1 = 12$, $n_2 = 24$ and $n_3 = 48$.

The results show that for fixed visual quality the proposed ABAC allows to decrease the bit rate by 0.5–2.3% for H.264/AVC standard and 0.6–3.6% for H.265/HEVC standard on average. Herewith, for low QP’s (0–30) the bit rate savings is caused mostly by less coding redundancy of window length $W = 2^6$, while for high QP’s (40–50) the savings results mostly from the higher speed of the probability adaptation, which is very crucial for encoding of short binary sequences. It is important to notice, that for an implementation of the proposed window selection approach additional tables or multiplication/division operations are not needed. Therefore, in this scenario the computation complexity is comparable with the M-coder.

Table 2.4: Bit rate savings (in %) compared to the M-coder for H.264/AVC standard

QP	0	10	20	30	40	50
Foreman	0.93	0.46	0.43	0.10	0.02	0.15
Mobile	0.04	0.17	0.30	0.46	0.20	0.49
Akyio	0.85	0.43	0.12	-0.17	0.68	5.04
Mother Daughter	1.14	0.81	0.56	0.06	0.44	4.81
Vassar	1.35	0.77	0.81	0.42	1.72	8.25
Ballroom	1.23	0.66	0.62	0.50	0.51	0.84
City	0.79	0.53	0.58	0.73	0.60	1.24
Crew	1.01	0.43	0.73	0.59	0.79	1.58
Pedestrian Area	1.14	0.69	0.72	0.79	1.07	1.47
Rush Hour	1.19	0.87	0.88	0.98	0.97	1.46
Station	1.39	1.06	1.07	0.59	0.70	1.20
Tractor	1.02	0.53	0.61	0.79	0.84	0.96
Average	**1.01**	**0.62**	**0.62**	**0.49**	**0.71**	**2.29**

Table 2.5: Bit rate savings (in %) compared to the M-coder for H.265/HEVC standard

QP	0	10	20	30	40	50
Foreman	0.64	0.58	0.32	0.18	0.80	2.76
Mobile	0.45	0.35	0.52	0.72	1.23	2.80
Akyio	0.75	0.73	0.63	0.74	2.07	6.18
Mother Daughter	0.74	1.06	0.89	0.89	2.21	5.54
Vassar	0.49	0.29	0.96	0.73	1.57	6.27
Ballroom	0.40	0.27	0.44	0.46	1.01	1.93
City	0.52	0.56	0.91	1.04	1.94	3.37
Crew	0.58	0.59	0.78	0.52	1.18	3.98
Pedestrian Area	0.72	0.75	0.61	0.66	0.87	1.54
Rush Hour	0.79	0.91	0.75	0.82	0.91	1.66
Station	0.78	1.01	0.87	0.62	0.95	2.72
Tractor	0.48	0.52	0.62	1.38	2.73	4.31
Average	**0.61**	**0.63**	**0.69**	**0.73**	**1.46**	**3.59**

2.6 Conclusion

Efficient look-up table-free ABAC and ABRC have been proposed. In comparison to the M-coder they have the following advantages:

1. The proposed ABAC and ABRC do not use any look-up tables allowing a reduction in entropy encoder memory consumption (6-20% for H.264/AVC) and (32-42% for H.265/HEVC).
2. The proposed ABAC and ABRC provide simple mechanism to control the trade-off between the speed of probability adaptation and the precision of probability estimation through a single parameter, the window length.
3. The proposed ABAC and ABRC do not require any changes in the context-modeling and can be easily embedded for performance improvement to any compression scheme which is based on binary arithmetic coding.
4. The proposed ABAC provides bit rate savings from 0.5 to 2.3% for H.264/AVC standard and from 0.6 to 3.6% for H.265/HEVC standard on average.
5. The proposed ABAC has comparable computational complexity in comparison with the M-coder, while the proposed ABRC is up to 34% less complex than both ABAC and the M-coder. Taking into account that the proposed ABRC needs multiplication in interval division part, it is more preferable in software applications where a multiplication cost is low. The proposed ABAC does not need multiplication and it is attractive for software as well as for hardware applications.

Chapter 3

Bit-plane Entropy Coding and Skipping of Wavelet Subbands

3.1 Introduction

This chapter is dedicated to the development of computationally efficient video coding algorithm based on 3-D DWT. First of all, taking into account a high computational complexity of the motion compensated temporal filtering used in wavelet schemes (see, for example, [30]), in this thesis it is proposed to avoid a step of motion compensation for temporal redundancy removal. Instead, a simple one-dimensional discrete wavelet transform (1-D DWT) is performed for samples with the same coordinates in each group of frames (GOF). As a result, the complexity of the encoder includes two main parts:

1. Wavelet transform complexity.
2. Wavelet subbands entropy encoding complexity.

In most of the existing video coding approaches, increasing the compression ratio decreases the complexity of the entropy encoder. This is due to the fact that after quantization, the entropy encoder has less input data to process. Herewith, the complexity of the transform part is kept close to a constant value. Therefore, it is important to develop approaches which allow decreasing of the entropy encoder complexity as well as the transform complexity depending on the compression ratio while maintaining the codec rate-distortion performance.

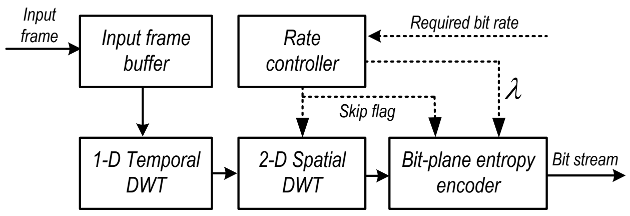


FIGURE 3.1: Considered 3-D DWT codec

The rest of the chapter is organized as follows. Section 3.2 reviews the basic video codec based on 3-D DWT. Section 3.3 proposes a low-complexity bit-plane entropy coder, while Section 3.4 proposes parent-child skipping of 2-D DWT and entropy bit-plane encoding. Rate-distortion-complexity comparison of the 3-D DWT codec with the state-of-the-art video codecs are presented in Section 3.5. Conclusions are drawn in Section 3.6.

3.2 Basic Video Codec Based on 3-D DWT

In this thesis it is proposed to use a simplified 3-D extension of the JPEG 2000 standard [35, 56] as the basic video coder (see Figure 3.1). First, a GOF of length N are accumulated in the input frame buffer. Then, a one-dimensional multilevel DWT of length N in the temporal direction is applied. After this, 2-D DWT is applied for each frame. This transform easily provides both temporal and spatial scalability of the video stream.

Figure 3.2 illustrates the example of the wavelet decomposition for a GOF size $N = 4$ with two-level temporal and two-level spatial wavelet decomposition. One can see that this decomposition allows to have 3 temporal scalable layers (full, half and quarter frame rate) and 3 spatial scalable layers (full, quarter and eighth frame resolution).

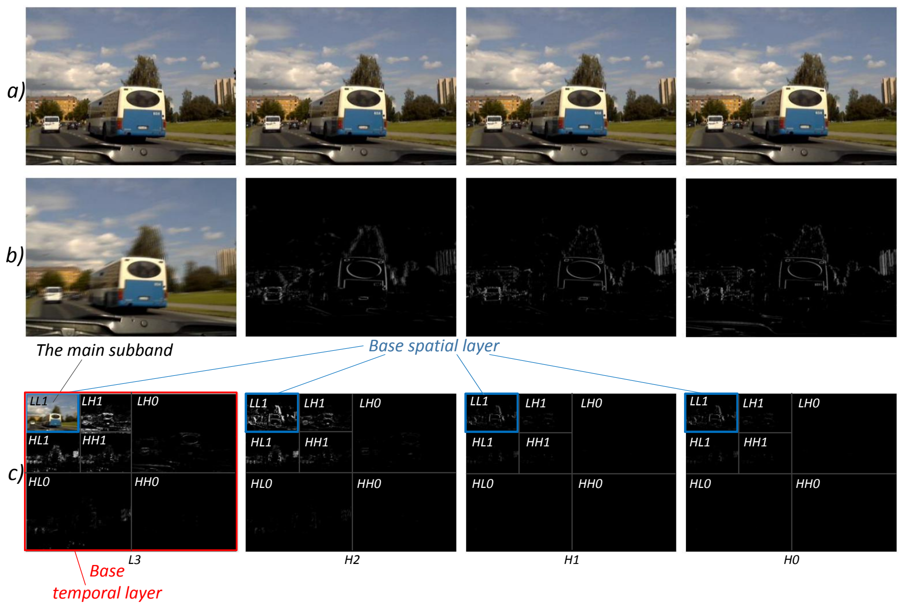
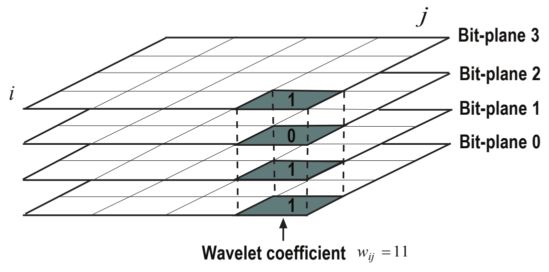


FIGURE 3.2: An example of the wavelet decomposition. a) original frames b) frames after 1-D Temporal DWT c) frames after 1-D Temporal DWT and 2-D Spatial DWT



(a) Subband representation using bit-planes

D_0	V_0	D_1
H_0	X	H_1
D_2	V_1	D_3

(b) Context model

FIGURE 3.3: Bit-plane entropy coding of a wavelet subband

Each spatial wavelet subband is represented as a set of bit-planes (see Figure 3.3 *a*) and independently compressed using bit-plane entropy coding. The bit with number n of each wavelet coefficient belongs to the bit-plane n. First, the number of significant bit-planes in subband n_{max} is calculated and encoded and all coefficients are set to insignificant. Then all coefficients are processed from the highest to the lowest bit-plane, in a raster scan. If the current bit of a coefficient is one, then the coefficient starts to be significant. Each bit of insignificant coefficient is encoded by the M-coder. The context number c is calculated according to Table 3.1 depending on the wavelet subband type (HH, HL, LH or LL) and

Table 3.1: Bit-plane contexts model for wavelet subband

	LL and LH subbands			HL subbands			HH subbands	
c	H	V	D	H	V	D	$(H+V)$	D
0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	1	0
2	0	0	2	0	0	2	2	0
3	0	1	–	1	0	–	0	1
4	0	2	–	2	0	–	1	1
5	1	0	0	0	1	0	2	1
6	1	0	≥ 1	0	1	≥ 1	0	2
7	1	≥ 1	–	≥ 1	1	–	1	2
8	2	–	–	–	2	–	2	2
9	bit is first after significant bit							
10	bit is not first after significant bit							
11	sign of a coefficient							

the amount of horizontal (H_0 and H_1), vertical (V_0 and V_1) and diagonal (D_0, D_1, D_2 and D_3) significant coefficients (see Figure 3.3 *b)*). If the current coefficient is significant, then its bit is compressed depending on the position after a significant bit. Finally, all signs are compressed according to a one context.

After encoding of each bit-plane n the bit stream size $r(n)$ and distortion $d(n)$ of the subband are determined. If the Lagrange sum $\psi(n) = d(n) + \lambda \cdot r(n)$ for the bit-plane n is less than the sum $\psi(n+1) = d(n+1) + \lambda \cdot r(n+1)$ for the bit-plane $n+1$, then the bit stream is truncated at the bit-plane $n+1$, and the encoding process is stopped. Otherwise, encoding is continued.

3.3 The Proposed Low-complexity Bit-plane Entropy Coder

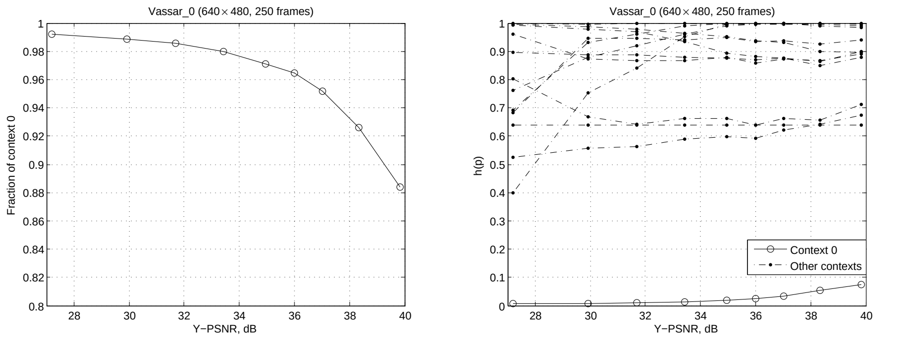
3.3.1 Usage of ABRC Instead of ABAC

As it is shown in Chapter 2, the proposed ABRC has up to 34% less computational complexity due to use of bytes as an output bit stream element and byte renormalization (see Section 2.5.1). Therefore, the computational complexity of the bit-plane entropy coder, described in section 3.2, can be reduced due to the replacement of the M-coder by the ABRC. For better compression efficiency it is proposed to use virtual sliding window length $w = 2^{10}$ for low-entropy context $c = 0$ and $w = 2^6$ for the rest contexts ($c = 1, ..., 11$). As shown in Figure 3.6, using the ABRC increases the encoding speed[1] of 3-D DWT coder from 20% to 30% on average with an increase in compression performance.

[1]The encoding speed in this thesis is defined as the number of frames which can be encoded in one second on the PC with processor Intel Core 2 DUO CPU 3.0GHz. The computational complexity is considered as inverse value of the encoding speed

3.3.2 Combined Entropy Coding

In order to further decrease the computational complexity, let us consider the following properties of different binary contexts. Figure 3.4 *a)* shows the typical fraction of the context $c = 0$ in all input binary data depending on Peak signal-to-noise ratio of the luminance component (Y-PSNR) for the entropy encoder given in Section 3.2. Figure 3.4 *b)* shows a typical binary memoryless entropy estimation for different binary contexts. One can see that the context $c = 0$ has more than 88% fraction in all binary input data for any Y-PSNR. At the same time a binary entropy of the source which corresponds to the context $c = 0$ does not exceeds 0.1 and, in most cases, it is significanly less than that. This means that the source can be efficiently compressed by zerorun-length coding. In this case the compression efficiency is not decreasing. On the other hand, computational complexity is decreasing because of the ABRC compresses 12% or less of the binary input data.



(a) Fraction of context 0 in all input binary data (b) Binary entropy estimation for different contexts

FIGURE 3.4: Properties of different binary contexts

Taking into account the reasoning above, in this thesis the following combined bit-plane entropy encoding algorithm is proposed (see Fig. 3.5). The binary sequence corresponding to the context $c = 0$ is compressed by the zero-run length coding and placed into Buffer 1. The other binary sequences are compressed by the ABRC and placed into the Buffer 2. For zero-run coding it is proposed to use Levenstein codes with a prefix based on equal codes. It helps to avoid using look-up tables during the coding process in comparison with the Huffman codes or CAVLC in H.264/AVC that is more efficient from the computational complexity point of view.

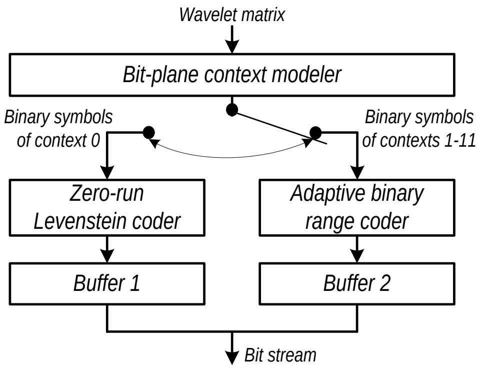

Wavelet matrix
Bit-plane context modeler
Binary symbols of context 0
Binary symbols of contexts 1-11
Zero-run Levenstein coder
Adaptive binary range coder
Buffer 1
Buffer 2
Bit stream

FIGURE 3.5: Proposed combined entropy coder

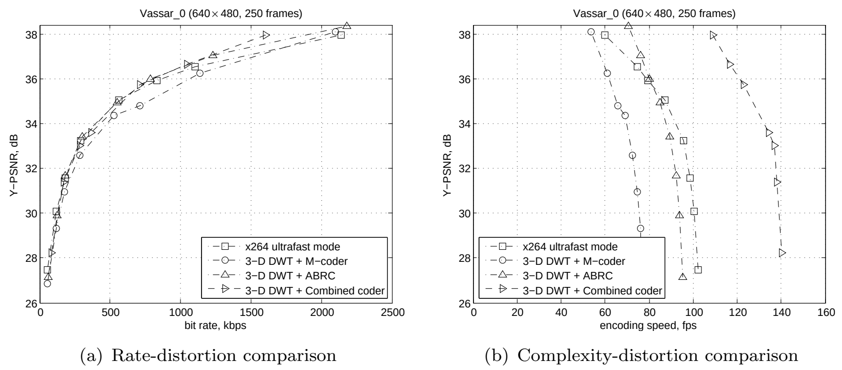
For compression of zero-run $\underbrace{000..0}_{L}1$ value m is calculated:

$$m = \begin{cases} 0, \text{ if } L = 0 \\ \lfloor \log_2 L \rfloor + 1, \text{ if } L > 0. \end{cases}$$

Then the value L is represented by the following two parts:

1. $\lfloor \log_2(\log_2 L_{max}) \rfloor + 1$ bits for binary representation of m;
2. $m-1$ bits for binary representation of $L-2^{m-1}$, where L_{max} is the maximum possible zero-run value.

As shown in Figure 3.6, the proposed combined-coder allows to increase the encoding speed of 3-D DWT coder from 77% to 85% on average in comparison with the one using the M-coder and it provides comparable rate-distortion performance with coder based on ABRC only.



(a) Rate-distortion comparison (b) Complexity-distortion comparison

FIGURE 3.6: Rate-distortion-complexity comparison of 3-D DWT codec based on the M-coder, ABRC and the proposed combined coder

3.4 The Proposed Skipping of 2-D Transform and Entropy Encoding

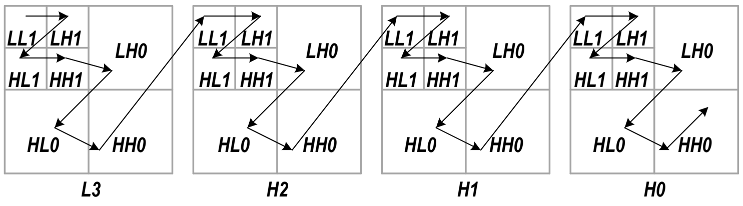
As it was mentioned in Section 3.2, after encoding of each bit-plane n the bit stream size $r(n)$ and distortion $d(n)$ of the subband are determined. If the Lagrange sum $\psi(n) = d(n) + \lambda \cdot r(n)$ for the bit plane n is less than the sum $\psi(n+1) = d(n+1) + \lambda \cdot r(n+1)$ for the bit-plane $n+1$, then the bit stream is truncated at the bit-plane $n+1$, and the encoding process is stopped. In some cases, after compression of the highest bit-plane n_{max} the following inequality holds:

$$d(n_{max}) + \lambda \cdot r(n_{max}) > \sum_{(i,j)} x^2[i,j], \tag{3.1}$$

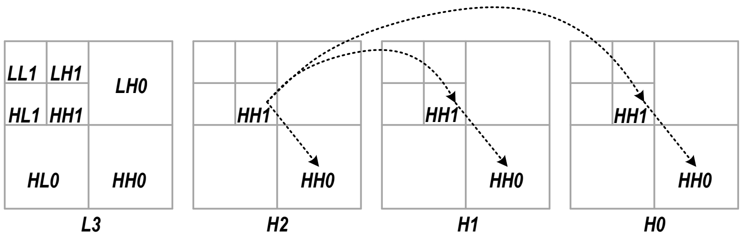
where $x^2[i,j]$ is wavelet coefficient with coordinates (i,j). In these cases, even the highest bit-plane is not included into the output bit stream, i.e. the subband is skipped. In this situation the encoder wastes computational resources to calculate the 2-D DWT and to form the subband bit stream which is not included in the output bit stream. Therefore, it is important to find a criterion which can guarantee with a high probability that the current subband will be skipped *before* processing of this subband.

In this thesis the parent-child tree based subband skip criterion is proposed. It is well known that the value of the wavelet coefficient in the child subband is correlated with the corresponding coefficient in the parent subband. This property is widely used in image and video compression algorithms based on inter-subband correlation (see, e.g. [27]). Using the same reasoning, all subbands are proceeded from low-frequency to high-frequency temporal subbands and from low-frequency to high-frequency spatial subbands. Second, the following *coding assumption* is made: if for any subband, inequality (3.1) holds, then for all temporal-spatial child-subbands, the same inequality also holds. In this case, all child-subbands will not be processed and hence spatial transform calculation and entropy coding steps are skipped. At the decoder side, the corresponding transform coefficients are considered to be zero.

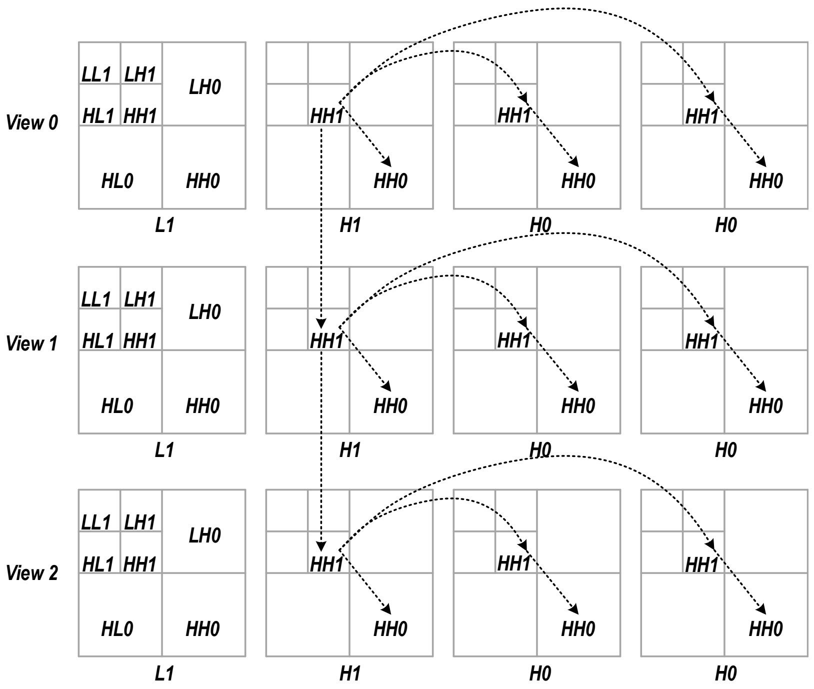
Figure 3.7 illustrates the proposed approach for a GOF size $N = 4$ with two-level temporal and two-level spatial wavelet decomposition. Frames after temporal wavelet decomposition are denoted as $L3, H2, H1, H0$, where $L3$ is the low-frequency frame.



(a) Spatial subbands encoding order



(b) Parent-child skipping tree



(c) Parent-child skipping tree in case of multi-view video coding

FIGURE 3.7: Illustration the proposed approach for a GOF size $N = 4$

Figure 3.7 *a)* shows the spatial subbands encoding order. If the spatial subband $HH1$ in the frame $H2$ is skipped (see Figure 3.7 *b)*) then the corresponding child-subbands $HH0$ in the frame $H2$, $HH1$ and $HH0$ in the frame $H1$, and $HH1$ and $HH0$ in the frame $H0$ are skipped without any processing. Additionally, in case of multi-view video coding, corresponding HH1 subbands in views 1 and 2 are skipped together with their child subbands (see Figure 3.7 *c)*).

Figure 3.8 shows the fraction of skipped wavelet subbands depending on the Y-PSNR for the different test video sequences. One can easily notice that this fraction is highly dependent on the motion level in the video sequence (compare for instance the results for "Football" which contains high motion with "Vtc1nw" which contains low motion). On the other hand, in all cases, as Y-PSNR decreases (or the compression ratio increases), the fraction of skipped subbands increases and reaches 95% in the limit. Therefore, as the compression ration increases, the computational complexity of the proposed algorithm tends to the complexity of the 1-D temporal transform.

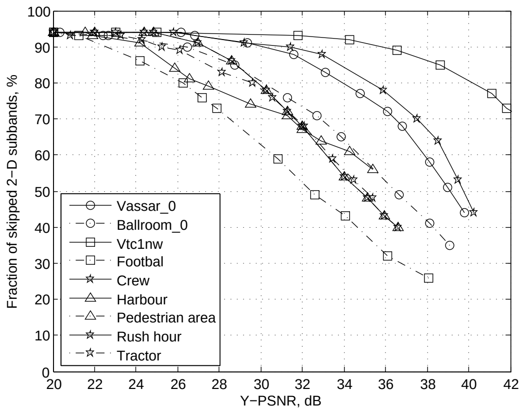
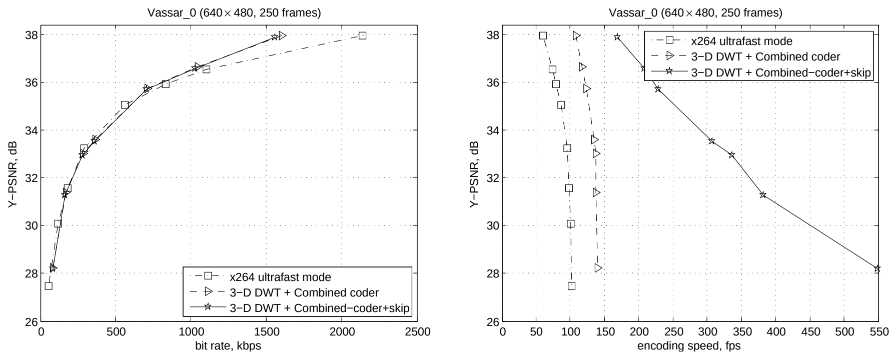


FIGURE 3.8: Efficiency of the proposed parent-child tree based subband skip criterion

As shown in Figure 3.9, the proposed skipping approach allows to increase the encoding speed from 1.6 to 3.6 times in comparison with the 3-D DWT codec with the combined coder and without skipping, and from 2 to 7 times in comparison with one using the M-coder without any significant degradation of the rate-distortion performance.



(a) Rate-distortion comparison (b) Complexity-distortion comparison

FIGURE 3.9: Rate-distortion-complexity comparison of 3-D DWT codec with and without the proposed parent-child skipping

3.5 Comparative Results

Simulation results were obtained for the test video sequences "Vassar_0", "Ballroom_0" and "Football" [50, 51] with frame resolutions 640×480 (more results can be found in [P2]). For our experiments, the proposed 3-D DWT codec is compared with the x.264 codec [52] which, as shown in [19], provides close to optimum rate-distortion performance for the H.264/AVC standard when computational complexity is significantly restricted. Therefore, this codec can be used as an upper bound of rate-distortion performance which can be achieved by H.264/AVC standard in a low complexity case.

The proposed 3-D DWT codec was run with GOF size $N = 16$ using the Haar wavelet transform in the temporal direction and the 5/3 spatial lifting wavelet transform at three-levels of the decomposition[2]. x.264 codec was run in very low complexity mode which corresponds to the Baseline profile of H.264/AVC with intra-frame period 16. Additionally, we have measured the rate-distortion-complexity performance for the reference software of the H.265/HEVC standard (HM version 9.1 in low delay configuration), 3-D SPIHT codec, x.264 codec in high complexity mode (which corresponds to the High profile of the H.264/AVC standard) and x.264 codec in very low complexity mode with Intra-frame coding only. In all cases, the codecs were simulated using a constant bit rate mode and the encoding speed was measured without any use of assemblers, threads, or other program optimization techniques.

[2]The proposed codec can be found at `http://www.cs.tut.fi/~belyaev/3d_dwt.htm`

Simulation results, presented in Figures 3.10, show the rate-distortion-complexity comparison for the considered codecs. From Figures 3.10, it can be seen that the highly complex video codecs (H.264/HEVC, 3-D SPIHT and x.264 in high complexity mode) provide significantly better rate-distortion performance than low complexity video codecs. But, the reduction in complexity causes a significant degradation in the rate distortion performance of H.264/AVC, especially for sequences with high level of motion (for example, up to 4 dB for "Football").

Furthermore, the proposed 3-D DWT video codec provides from 2 to 6 times lower computational complexity for the same Y-PSNR level compared to the H.264/AVC in the low complexity mode. For visual assessment frame 45 of video sequence "Ballroom_0" is depicted, when the required bit rate is 500 kbps (see Figure 3.11). One can see, that the visual quality for the proposed codec is comparable with x.264 in the low-complexity mode and significantly better than x.264 in the INTRA mode.

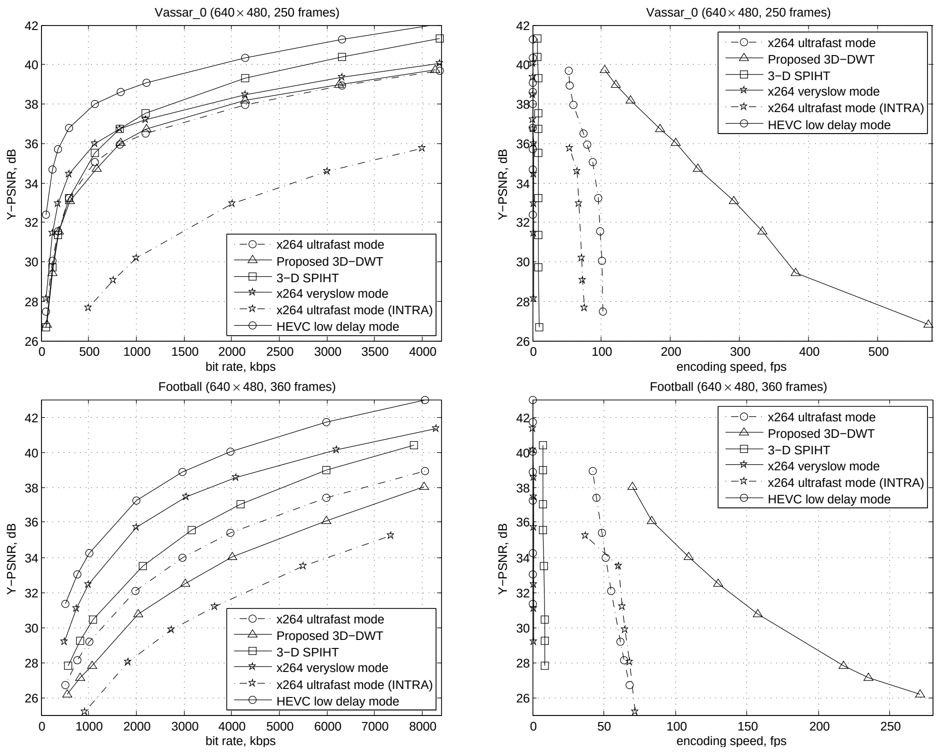
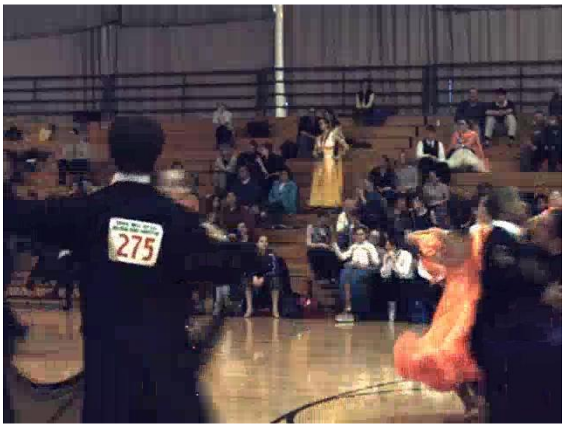
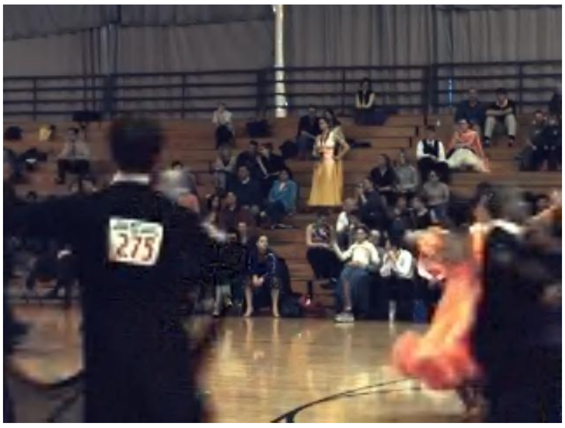


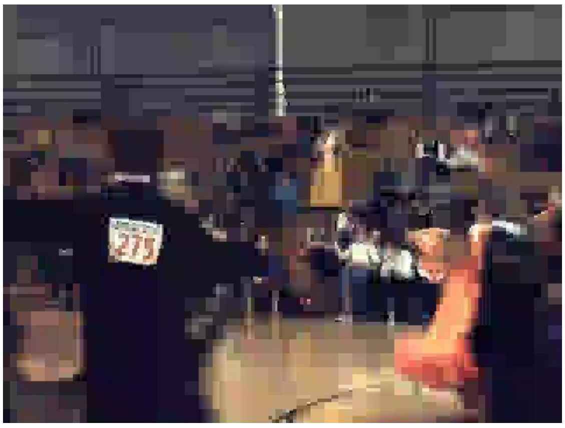
FIGURE 3.10: Rate-distortion-complexity comparison different codecs



(a) x.264 ultrafast, Y-PSNR=29.8 dB, Encoding speed=79.6 fps



(b) Proposed 3-D DWT, Y-PSNR=31.2 dB, Encoding s=238 fps



(c) x.264 INTRA, Y-PSNR=24.4 dB, Encoding speed=74.1 fps

FIGURE 3.11: Visual comparison of different low-complexity codecs for target bit rate 500 kbps

3.6 Conclusion

Efficient tools are proposed to decrease the computational complexity of SVC scheme based on 3-D DWT:

1. The proposed bit-plane entropy encoder uses very simple zero-run coding based on Levenstein codes for low entropy contexts, while the remaining contexts are compressed by ABRC proposed in Section 2.4. As a result, the complexity of the proposed combined bit-plane entropy coder is around 2 times less than the traditional one.

2. The proposed parent-child tree based subband skip criterion allows skipping the 2-D DWT and entropy encoding so that the entropy encoder complexity as well as the transform complexity are decreasing with an increase of the compression ratio. Herewith, the use of this criterion does not lead to a significant degradation of the rate-distortion performance.

3. Simulation results shows that 3-D DWT codec with the proposed bit-plane entropy encoder and subband skipping has much lower computational complexity (from 2 to 6 times) for the same quality level compared to the H.264/AVC standard in the low complexity mode (x264 codec in ultrafast profile). Taking into account that H.264/SVC has lower rate-distortion performance [57] and higher computation complexity than H.264/AVC single-layer coding (due to additional inter-layer prediction, input frame down-sampling etc.), the proposed 3-D DWT codec can be more preferable than H.264/SVC in many applications where computational complexity and scalability play a critical role.

Chapter 4

Joint Unequal Loss Protection and Rate Control

4.1 Introduction

Taking into account that video communication channels can be unreliable, a video bit stream generated by a codec should not be very sensitive to packet losses, i.e. packet losses should not lead to error propagation. In the case of hybrid video coding, a high compression efficiency is achieved due to a block-based motion compensation. On the other hand, if some parts of a reference frame with a residual data or motion vectors are lost, then a significant error propagation can occur. Therefore, it is important to develop video coding algorithms achieving a good balance between compression efficiency and sensitivity to packet losses. For better visual quality achievement, inter-packet error-correction codes (for example, RS codes) can be used at the application layer. Herewith, to select an optimal code redundancy, i.e. portion of the source and parity packets in the overall bit stream, the expected end-to-end distortion caused by packet losses should be calculated on the transmitter side. In case of hybrid video coding this calculation can be highly complex because it requires the full decoding process for each combination of packet losses. Taking into account that the code redundancy selection should be done for a given overall bit rate constraint in real-time and with low complexity, this task becomes challenging.

This chapter shows how this challenging task can be solved for 3-D DWT codec proposed in Chapter 3. The rest of the chapter is organized as follows. Section 4.2 investigates the influence of packet losses on the reconstructed video data for the proposed 3-D DWT codec and introduces a low-complexity error-resilient coding and error concealment taking into account this investigation. Section 4.3.1 proposes a low-complexity joint ULP based on inter-packet RS codes and source and parity packets bit rate control. Rate-distortion-complexity comparison of the 3-D DWT codec with the state-of-the-art H.264/SVC codec in case of ULP are presented in Section 4.4. Conclusions are drawn in Section 4.6.

4.2 Influence of Packet Losses on the Reconstructed Video

4.2.1 Video Bit-stream Packetization

For simplification, let us consider an example of coding and packetization of a wavelet subband with four non-zero bit-planes in 3-D DWT codec proposed in Chapter 3 (see Figure 4.1). Let us assume that at the encoder side the subband was truncated at the bit-plane $t^* = 1$. In this case Levenstein coder as well as the range coder generates three subpackets which contain information about bit-planes 3, 2, and 1. If no packet losses happened, then $r(1)$ bits are received, that corresponds to distortion $d(1)$. Let us assume that subpacket 1 from Levenstein coder buffer and subpacket 2 from the range coder buffer are lost. Taking into account that subpackets from both coders are needed for bit-plane decoding, only bit-plane 3 will be reconstructed by the decoder that corresponds to distortion $d(3)$. In this case, the distortion for the subband corresponds to the case when, at the encoder side, the subband is truncated at bit-plane $t^* = 3$ and there is no packet losses, i.e. packet losses in the 3-D DWT codec does not lead to error propagation in the bit stream and corresponds to higher truncation (quantization) of the subband.

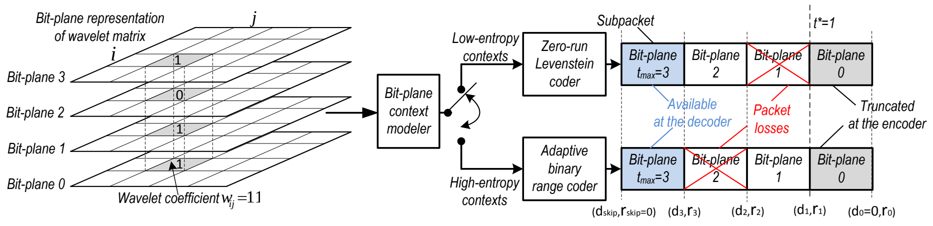


FIGURE 4.1: Illustration of bit-plane entropy encoding, packetization, truncation, packet losses and bit stream available at the decoder

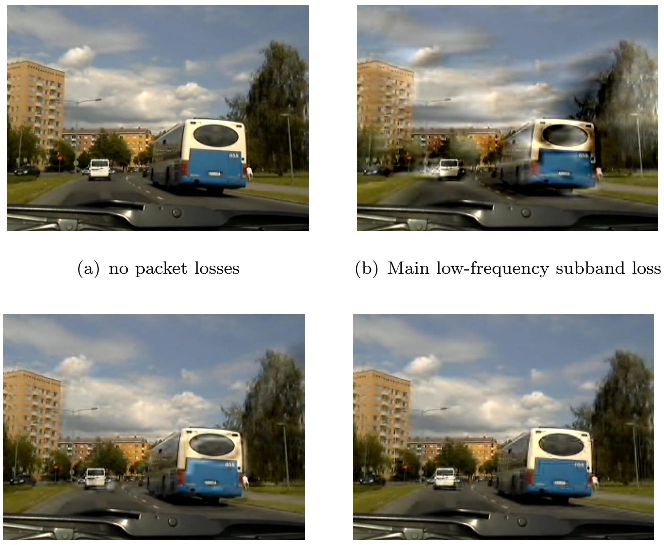
Taking into account that in many cases a subpacket length generated by Levenstein coder and range coder can be significantly less than network packet length, it is needed to packetize it to a network packet of larger length. Let us assume the packet loss probability is p. Then if two subpackets corresponding to the same bit-plane are placed into different packets, then the probability that both of them will be not received is $\pi = 1 - (1 - p)^2$, and $\pi = p$, if these subpackets are placed into the same packet. It is easy to see, that in the second case, the probability that the subpackets are lost is always less than or equal to that of the first case. Therefore, we are using the second type of packetization in all cases, when the subpackets can be placed into the same packet and the first type for the rest of the cases.

4.2.2 Error-resilience Coding and Error Concealment

The video bit stream generated by the 3-D DWT codec can be divided into the main components, which can be listed in order of importance with respect to their influence on the packet losses on the reconstructed video as follows:

1. *The main subband in the base spatial layer* (LL1 subband in frame L3 in Figure 3.2) contains the brightness information for the whole GOF. Therefore, the loss of this subband corresponds to the loss of all frames in the GOF.
2. Due to properties of the 1-D temporal wavelet transform, in case of losses of *the remaining subbands of the base spatial layer* (LL1 subband in frames H2, H1 and H0 in Figure 3.2), at the decoder side the reconstructed video will contain specific temporal artifacts, like a “ghost effect“ (see Figure 4.2*c*).
3. Losing the *remaining subbands* leads to the spatial artifacts, when frames in the GOF appear such as after downsampling (see Figure 4.2*d*). But in this case the reconstructed video has an acceptable visual quality.

Applying the above reasoning, at the decoder side we use the following error concealment. First, the bit stream of each subband is decoded in a progressive way until a loss is detected. In this case, any losses in a stream correspond to the higher quantization at the encoder side and do not lead to an error propagation. Second, if the main subband, which contains the brightness information for all GOFs, is lost, then we copy the corresponding subband from the previous GOF. In this case an error propagation can happen (see the example in Figure 4.2*b*). To minimize the probability of this event we use the following simple error-resilient



(a) no packet losses (b) Main low-frequency subband loss

(c) temporal artifacts (d) spatial artifacts

FIGURE 4.2: Examples of artifacts caused by packet losses

coding. At the encoder side for the main subband it is proposed to use repetition of the highest bit-planes which can be placed into *two* additional packets. We use two packets because it is the minimum data portion which is enough to represent this brightness acceptably. For the remaining subbands of the base spatial layer it is proposed to use repetition of the highest bit-planes which can be placed into *one* additional packet. Here we use one packet, because it is enough to avoid significant temporal artifacts caused by inverse temporal wavelet transform.

Notice, that the proposed error-resilience coding uses some insignificant amount of operations needed for packet repetition at the encoder side. Therefore, it does not affect the overall encoder complexity.

4.2.3 Comparative Results

Influence of packet losses on reconstructed video data were compared for the proposed 3-D DWT codec and H.264/SVC. For 3-D DWT we have used GOF size $N = 16$ with Haar transform in the temporal direction and the 5/3 spatial wavelet transform with a three-level decomposition. For H.264/SVC we have used the JSVM 9.8 reference software [55] with 2 spatial and 5 temporal scalable layers. The GOP size and intra-frame period are set to 16. For error-resilient coding, flexible macroblock ordering with two slice groups and loss-aware rate-distortion optimized macroblock mode decision have been used. At the decoder side, the frame copy error concealment method is applied.

Simulation results which illustrate influence of packet losses on the video bit stream generated by the proposed 3-D DWT codec and H.264/SVC standard were obtained for the video sequences “Tampere_01“ (640×480, 300 frames, 30 Hz) [54] and “Football“ (640×480, 360 frames, 30 Hz) [51]. For comparison, we have used the following approach. First, each video codec generated the video stream as a set of packets. The packet length was set to 800 bytes. Then, we randomly removed the packets from this set by using the independent packet losses model with a loss probability p. The resulting video stream was used as an input to the video decoder for reconstruction of the received video data. Then, the mean square error was calculated as:

$$D = \frac{1}{F \cdot W \cdot H} \sum_{f=0}^{F-1} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \left(s[x,y,f] - s'[x,y,f]\right)^2, \tag{4.1}$$

where $s[x,y,f]$ and $s'[x,y,f]$ are the luma values for the pixel with coordinates (x,y) in the frame f for the original and reconstructed video sequences, respectively, W is the frame width, H is the frame height, and F is the number of frames in the test video sequence. Note that losses of different packets lead to different distortion values D in the reconstructed video. Therefore, we repeat the mean square error calculation described above for K different packet loss realizations. Finally, the visual quality metric Expected Peak Signal-to-Noise Ratio (E[Y-PSNR]) of the luma color component is estimated as:

$$E[\texttt{Y-PSNR}] = 10 \lg \frac{s_{max}^2}{\frac{1}{K} \sum_{i=0}^{K-1} D_i}, \tag{4.2}$$

where s_{max} is the maximum possible luma value.

Figure 4.3 shows the expected visual quality for different packet loss probabilities and channel rates for the proposed codec and for H.264/SVC. As it can be noted, while H.264/SVC provides significantly better visual quality in case of no packet losses, it is much more sensitive to packet losses. One can see that the proposed codec provides better authenticity of the reconstructed video while H.264/SVC has frames with unrecognizable or new objects, which do not exist in the original video, even for relatively low packet loss probabilities (see the visual comparisons in Figure 4.4).

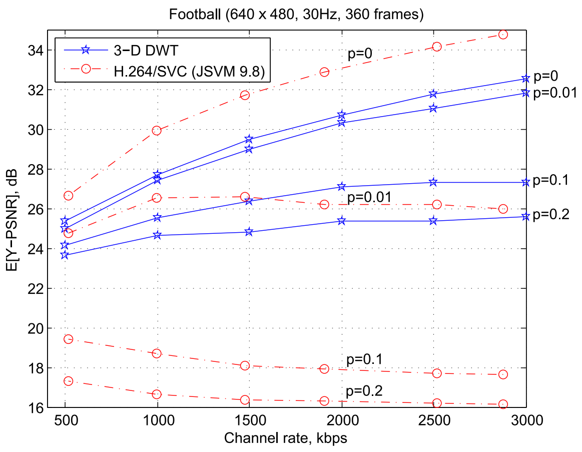


FIGURE 4.3: Expected visual quality comparison for 3-D DWT without packet loss protection and H.264/SVC without packet loss protection

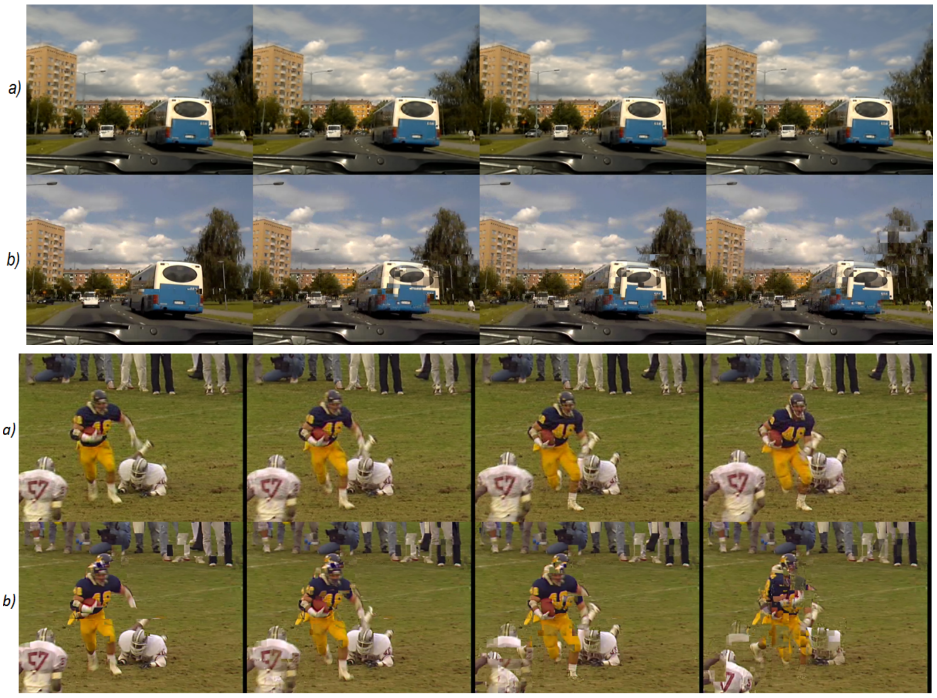


FIGURE 4.4: Visual comparison for 5% packet loss.
a) Proposed 3-D DWT b) H.264/SVC (JSVM 9.8)

4.3 The Proposed Joint Unequal Loss Protection and Rate Control

4.3.1 Inter-packet Loss Protection using Reed-Solomon Codes

For better visual quality achievement the inter-packet loss protection based on systematic RS (n, k) codes in the finite field $GF(2^8)$ is selected in this thesis. In this approach k source bytes with the same index are used to form $r = n-k$ parity bytes. Herewith, if the source byte with the current index does not exist (stuffing byte), zero byte is used instead (see example in Figure 4.5).

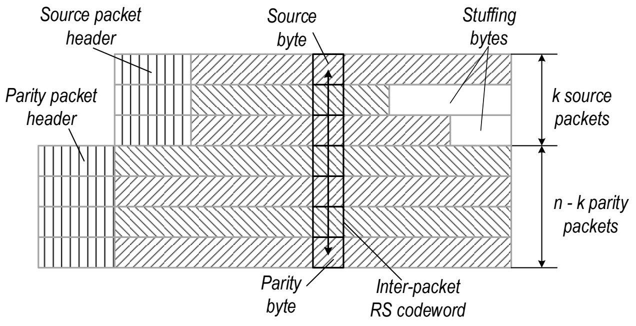


FIGURE 4.5: Inter-packet erasure protection example for RS $(7, 3)$ code

If k or more packets from n are delivered, then the RS decoder is able to recover all lost source packets. Let p denote the packet loss probability, then the probability that a source packet will be lost and not recovered by the RS decoder is

$$p_{dec}(p, n, k) = \sum_{i=n-k+1}^{n} \frac{i}{n} \cdot \binom{n}{i} \cdot p^i \cdot (1-p)^{n-i}. \tag{4.3}$$

Figure 4.6 illustrates the expected Y-PSNR for different loss protection redundancies $\frac{n-k}{n}$ when the source and parity packets bit rate is 3000 kbps and the packet loss probability is 20%. One can see, that if the redundancy is zero (no protection is used), then the expected visual quality has a minimum value. With an increase of the redundancy the distortion caused by packet loss protection decreases and the expected visual quality increases until some maximum value. A further increase of the redundancy does not lead to an increase of the quality, because in order to keep the source and parity packets bit rate equal to the channel rate, it is needed to compress the wavelet subbands with higher and higher compression ratios. In this case the distortion caused by compression increases and the distortion caused by packet losses is not significant because the video bit stream is overprotected. Therefore, the visual quality decreases. Thus, for the expected visual quality maximization it is important to find the optimal redundancy level for the given video sequence and packet loss rate.

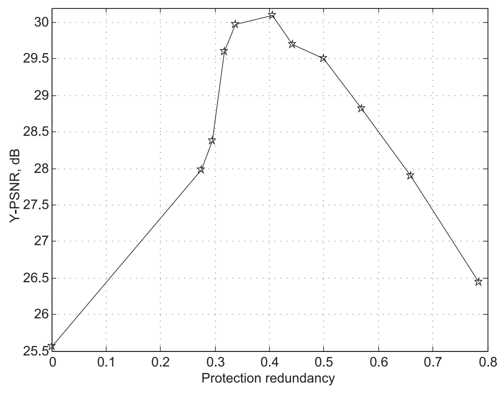


FIGURE 4.6: Expected visual quality in case of protection by RS codes for packet loss probability 20% and channel rate 3000 kbps for video sequence "Football"

4.3.2 End-to-end Expected Distortion Calculation

In this thesis it is proposed to use ULP based on inter-packet RS codes by selection of different protection levels (coding redundancies) for each wavelet subband. Figure 4.7 shows how the proposed bit-plane entropy encoder with two buffers (see Section 3.3) is connected with RS encoders with different coding redundancies. Compression starts from the highest bit-plane t_{max} and, when a subband is truncated at the bit-plane t^{*}, each buffer contains $t_{max} - t^{*} + 1$ *subpackets* (each bit-plane fits into two subpackets). If the RS code (n_i, k_i) for the protection of two subpackets, containing a bit stream of bit-plane i, is used, then the probability

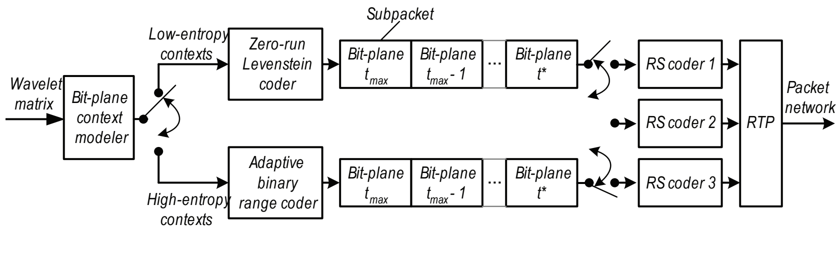


FIGURE 4.7: Proposed joint bit-plane wavelet subband entropy encoding and ULP scheme

that these two subpackets will not be received and not recovered is

$$\pi_i = 1 - \Big(1 - p_{dec}(p, n_i, k_i)\Big)^2, \tag{4.4}$$

when these subpackets are placed into the different source packets and

$$\pi_i = p_{dec}(p, n_i, k_i), \tag{4.5}$$

when the subpackets are placed into the same source packet.

Let $\psi = \{t^*, n^*, k^*\}$ denote a decision vector for a subband, which is truncated at a bit-plane t^* and for each bit-plane RS code (n^*, k^*) is used. Denote by d_i the subband distortion when the bit-planes $t_{max}, ..., i$ are received and the bit-plane $i-1$ is not received. Then the expected distortion after truncation at bit-plane t^* is

$$E[d(\psi)] = \pi_{t_{max}} \cdot d_{skip} + \sum_{i=t^*+1}^{t_{max}} \prod_{j=i}^{t_{max}} (1-\pi_j) \cdot \pi_{i-1} \cdot d_i + \prod_{i=t^*}^{t_{max}} (1-\pi_i) d_{t^*}, \tag{4.6}$$

where d_{skip} is the distortion, when the all bit-planes are not received. It is important to notice, that $E[d(\psi)]$ calculation is low complex, because it does not requires any decoding or other extra video data processing.

4.3.3 Problem Formulation and Solution by Lagrange Relaxation

Let $\Psi = \{\psi_i\}$ be the set of decision vectors, where ψ_i is the decision vector for the i'th subband in a GOF. The overall expected GOF distortion is

$$E[D(\Psi)] = \sum_i E[d(\psi_i)], \tag{4.7}$$

and the resulting GOF bit stream size is

$$R(\Psi) = \sum_i r(\psi_i), \tag{4.8}$$

where $r(\psi_i)$ is the bit stream size of the i'th subband when the decision vector ψ_i is applied. Then, source and parity packets rate control problem can be formulated as follows. For each GOF we should determine the set of decision vectors Ψ^*, such that

$$\begin{cases} \Psi^* = \arg\min_{\{\Psi\}} E[D(\Psi)] \\ R(\Psi^*) \leq R_{max}, \end{cases} \tag{4.9}$$

where $R_{max} = \frac{N \cdot C}{f}$ is the bit budget for each GOF, N is the GOF size, C is the channel rate, and f is the video source frame rate.

The rate control task (4.9) can be solved using the Lagrangian relaxation. It can be proven [53, 58], that for $\lambda \geq 0$, the decision vector Ψ^*_λ minimizing $E[D(\Psi)] + \lambda \cdot R(\Psi)$ is the solution of the rate control task (4.9) for $R_{max} = R(\Psi^*_\lambda)$. From this statement it follows that in order to solve (4.9) one should find λ, such that $R(\Psi^*_\lambda) = R_{max}$. It can be also proven that $R(\Psi^*_\lambda)$ is a non-increasing function of λ, i.e. λ can be found by the bisection method [53].

For the considered 3-D DWT codec, the rate-distortion function for the subband i is independent on the truncation and RS code parameters selected for other subbands, because all subbands are compressed independently. Therefore, a solution of the task (4.9) can be written as $\Psi^* = \{\psi^*_i\}$, where

$$\psi^*_i = \arg\min_{\{\psi_i\}} \{E[d(\psi_i)] + \lambda \cdot r(\psi_i)\}. \tag{4.10}$$

Minimization in (4.10) is solved analogically as it is described in Section 3.2 by replacing distortion $d(n)$ caused by truncation by expected distortion $E[d(\psi_i)]$ which is calculated using (4.6) and caused by truncation, loss protection mode and packet loss probability.

4.3.4 The Proposed Source and Parity Packets Rate Control

To find the necessary λ value with the bisection method, one needs to calculate the 2-D DWT and losslessly compress each subband in order to determine all sets of truncation points, which requires significant computational resources and contradicts to the wavelet subband skipping approach proposed in Section 3.4. Therefore, in this thesis it is proposed to use the *virtual buffer* concept to estimate λ without necessity of 2-D DWT calculation and lossless coding. Let us define b_{virt} to be the number of bits in a virtual buffer and B_{max} the virtual buffer size determined as $B_{max} = L \cdot C$, where L is the required buffering latency at the decoder side for continuous video playback, C is the channel rate. The proposed rate control is presented in Algorithm 8.

Algorithm 8 Proposed rate control

```
1: $b_{virt} \leftarrow b_{virt}(0)$
2: **for** $j = 0,..,j_{max} - 1$ **do**
3:    Calculate 1-D temporal DWT for GOF $j$.
4:    $\lambda \leftarrow \lambda_{max} \left(\frac{b_{virt}}{B_{max}}\right)^{\gamma}$
5:    **for** $i = 0,..,i_{max} - 1$ **do**
6:       **if** parent subband was skipped **then** skip subband $i$;
7:       Calculate 2-D spatial DWT for subband $i$;
8:       Find $\psi_i^* = \arg\min_{\{\psi_i\}}\{E[d(\psi_i)] + \lambda \cdot r(\psi_i)\}$
9:       **if** $r(\psi_i^*) = 0$ **then** skip subband $i$;
10:      $b_{virt} \leftarrow b_{virt} + r(\psi_i^*)$
11:   **end for**
12:   $b_{virt} \leftarrow b_{virt} - R_{max}$
13: **end for**
```

At step 4, Algorithm 8 selects the Lagrange multiplier λ value in the proportional to the virtual buffer fullness, where γ defines the proportion degree. It allows to increase or decrease the compression ratio for the current GOF depending on the buffer fullness. At step 5, all frames in the GOF are processed starting from low-frequency to high-frequency frames. For each frame, the spatial subbands are also processed from low-frequency to high-frequency spatial subbands (see Figure 3.7 *a)*). At step 8, each subband is compressed starting from the highest bit-plane, until truncation point and loss protection mode is found accordingly to (4.10). At step 9, if the highest bit-plane is not included into the output bit stream, then all corresponding child-subbands of the subband are skipped. At steps 10 and 12 the virtual buffer fullness is updated. One can see, that the proposed rate control does not contradicts to wavelet subband skipping proposed in Section 3.4. Moreover, in this case if a subband is skipped, then loss protection mode selection as well as packetization and RS encoding are skipped together with entropy encoding and 2-D DWT.

As an example, Figure 4.8 shows the Lagrange multiplier selected by the bisection method and the proposed rate control for different initial virtual buffer fullness. One can see that, in practice, after some adaptation period, the value of the λ approaches the optimal one, computed by the bisection method. It is important to notice, that in comparison with other heuristic rate control techniques, the proposed algorithm does not require any additional computations and provides close to the optimal Lagrange multiplier as in the bisection method.

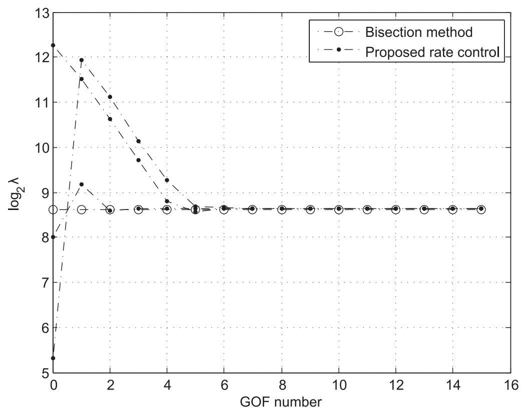


FIGURE 4.8: Lagrange multiplier selection

4.4 Comparative Results

Figures 4.9 show the expected PSNR for different packet loss probabilities and channel rates for the proposed 3-D DWT codec and for the H.264/SVC with unequal packet loss protection. Simulation results were obtained for the video sequence “Football“ (640 × 480, 360 frames, 30 Hz) [51]. For 3-D DWT codec, the joint source-channel video coding algorithm was implemented in real-time mode as it was described above, while ULP optimization for H.264/SVC was realized in off-line mode through multiple encodings with different quantization steps and protection modes.

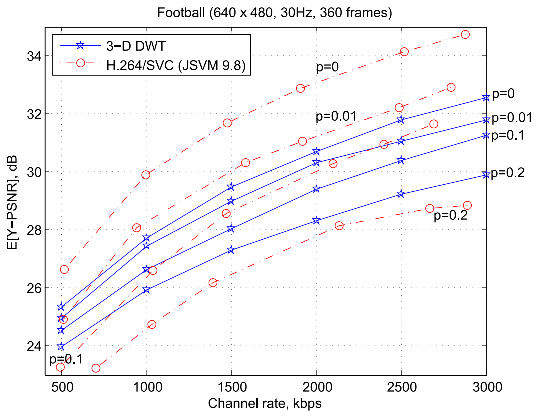


FIGURE 4.9: Expected visual quality comparison for 3-D DWT with ULP and H.264/SVC with ULP

For both codecs the RS codes from Table 4.1 were used, so different level of protection can be achieved for each scalable layer.

Table 4.1: Packet loss protection modes

Protection mode	1	2	3	...	11	12
Protection level	Four-fold repetition	RS (6,2)	RS (7,3)	...	RS (15,11)	No protection

One can see that in case of ULP, H.264/SVC provides better performance (up to 1.5dB) for low packet loss probabilities, but with increasing of it the proposed codec provides the same or better performance (up to 1.5dB). It can be explained by the following reason. Since H.264/SVC is more sensitive to packet losses it is needed to add more redundancy by RS coding than 3-D DWT codec, which has less compression performance, but requires less redundancy for protection (see Table 4.2).

Table 4.2: Protection redundancy

Packet loss probability, p	0	0.01	0.1	0.2
H.264/SVC (JSVM 9.8)	0	0.276	0.464	0.617
3-D DWT	0	0	0.271	0.412

Table 4.3 shows encoding speed for the proposed codec and H.264/SVC depending on the video bit rate. Taking into account that JSVM 9.8 reference software is not optimized for real-time operation we have also used fast implementation of the H.264/AVC standard in single-layer mode (x.264 codec in ultrafast profile [52]) to estimate possible upper bound of the encoding speed of H.264/SVC after optimization. Our estimation shows that the complexity of the proposed codec is 3–6 times less than H.264/SVC at least. Herewith, a higher complexity gain is expected in real-life implementations.

Additionally, we have added the full complexity for the proposed codec with ULP for different packet loss probabilities. One can see that the complexity of the proposed codec decreases with increasing of the packet loss probability or with decreasing of the bit rate. This can be explained as follows. An increase of a packet loss probability leads to an increase of the portion of parity packets in the output bit stream. In order to keep the source and parity data bit rate equal to the channel rate, the video compression ratio is increased, and condition $r(\psi_i^*) = 0$ in Algorithm 8 holds for more and more subbands which are skipped together with the corresponding child-subbands. As a result, 2-D spatial DWT, entropy coding and ULP mode selection are not performed for a significant portion of spatial subbands.

TABLE 4.3: Encoding speed for CPU Intel Core 2.1 GHz, fps

Video bit rate, kbps	500	1000	2000	2500	3000
H.264/SVC (JSVM 9.8)	< 1	< 1	< 1	< 1	< 1
x.264	49	44	40	37	36
3-D DWT	279	215	150	133	120
3-D DWT + ULP ($p = 0.1$)	288	185	160	137	130
3-D DWT + ULP ($p = 0.2$)	315	198	177	160	145

4.5 Real-world Experimental Results

For practical verification of the proposed results, first, the 3-D DWT codec was implemented and tested for video streaming over WiFi network of Tampere University of Technology (see Figure 4.10).

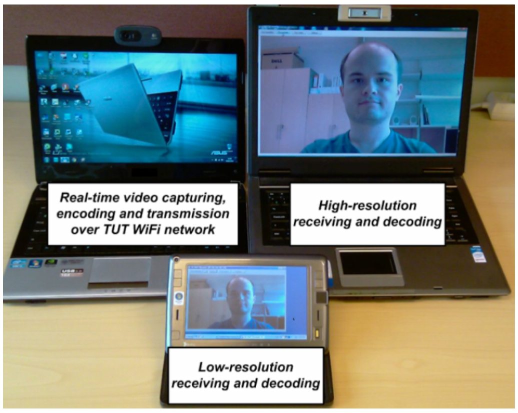
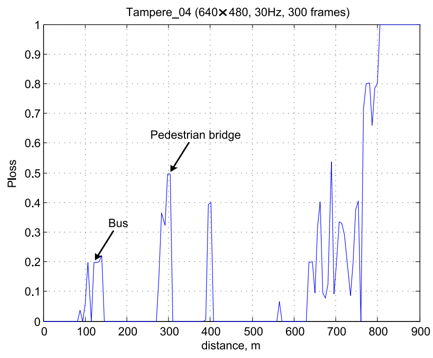
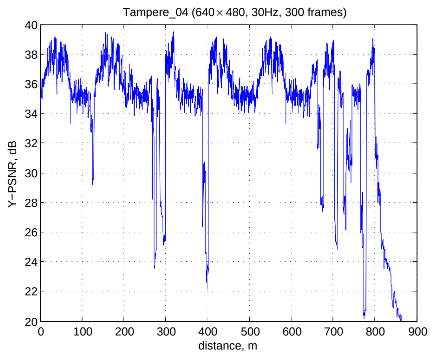

Real-time video capturing, encoding and transmission over TUT WiFi network
High-resolution receiving and decoding
Low-resolution receiving and decoding

FIGURE 4.10: Illustration of scalable video transmission over wireless network of Tampere University of Technology (TUT) based on the proposed 3-D DWT codec

Then, for further practical verification, the 3-D DWT codec was tested in the case of live video communication between moving vehicles [58]. The following experiment was handled in a major street close to the university campus in Hervanta, a suburb of Tampere, Finland. The video data was transmitted when the vehicle was moving at the velocity of 50 km/h starting from the distance 0 meters. Figure 4.11 *a)* shows the measured packet loss rate per GOF depending on the distance, while Figure 4.11 *b)* presents corresponding visual quality. During the experiment, the traffic was moderate and some cars, buses and pedestrian bridge occasionally obstructed the line of sight between the transmitting vehicle and receiving roadside unit (see labels on Figure Figure 4.11 *a)*). One can see that the proposed 3-D DWT codec provides acceptable visual quality even when the wireless channel is very unreliable and packet loss rate reaches 40%.



(a) Packet loss rate per GOF depending on the distance



(b) Visual quality depending on the distance

FIGURE 4.11: Real-world experimental results for the proposed 3-D DWT codec

4.6 Conclusion

Efficient low-complexity error-resilient, ULP and rate control for video codec based on 3-D DWT have been proposed. As a result, in comparison with H.264/SVC, the proposed 3-D DWT codec has the following advantages:

1. It is much less sensitive to packet losses and provides robust video transmission and authentic reconstruction for 1–5% packet loss rates without any packet loss protection. In the case of H.264/SVC even 1% packet loss rate is enough for significant visual quality degradation. Reconstructed video of H.264/SVC can be not authentic, i.e. new objects, which are not present in original video, can appear.

2. 3-D DWT codec equipped with the proposed ULP and rate control demonstrates better performance for high packet loss rates.

3. The computational complexity of 3-D DWT encoder with ULP and rate control is up to 6 times less compared to the H.264/AVC standard in the low complexity mode (x264 codec in ultrafast profile). Therefore, a full video transmission system based H.264/SVC with expected distortion estimation, error-correction coding and ULP optimization is likely to have much higher computational complexity.

Chapter 5

Conclusion and Future Work

In this thesis efficient tools are proposed to decrease the computational complexity of scalable video coding with unequal packet loss protection for video compression scheme based on 3-D DWT. These tools cover all levels of the codec development and include low-complexity and efficient adaptive binary arithmetic and range coding, low-complexity and efficient bit-plane entropy coding of wavelet subbands, rate-distortion efficient skipping of spatial wavelet transform and entropy coding, low-complexity and efficient joint unequal packet loss protection, rate control of source and parity video data. All above mentioned tools can be implemented as end-to-end solution and directly used in real-life scenarios.

Theoretical, simulation and real-world results presented in this thesis show that the proposed 3-D DWT codec can be efficiently used for live video transmission over highly unreliable channels and or in systems in which the video encoding computational complexity or power consumption plays a critical role.

Herewith, in comparison with the state-of-the-art hybrid video coding algorithms the proposed 3-D DWT codec has the following limitations. First, frames accumulation needed for temporal wavelet transform introduces additional algorithmic delay, which makes it impossible to use the 3-D DWT codec for videoconferencing applications. Second, if the video encoder complexity is not significantly restricted (in all applications using pre-encoded video), then the hybrid video coding achieves much better rate-distortion performance. Taking this into account, the future work can be dedicated to adaptation of the proposed tools for video coding schemes based on low-delay temporal transform and development of tools which can be used to increase the rate-distortion performance for scenarios when pre-encoded video is used.

References

[1] R.Viola, L.Chiariglione and R.Russo, "Video compression for manned space missions," *IEEE Global Telecommunications Conference*, 1989.

[2] A. Vinel, E. Belyaev, O. Lamotte, M. Gabbouj, Y. Koucheryavy, K. Egiazarian, "Video transmission over IEEE 802.11p: real-world measurements," *IEEE Workshop on Emerging Vehicular Networks*, 2013.

[3] J.Ribas, D.Sura and M.Stojanovic, "Underwater wireless video transmission for supervisory control and inspection using acoustic OFDM,"*OCEANS 2010*, 2010.

[4] I.Politis, M.Tsagkaropoulos and S.Kotsopoulos, "Optimizing Video Transmission over Wireless Multimedia Sensor Networks,"*IEEE Global Telecommunications Conference*, 2008.

[5] D.Turgis, R.Puers, "Image compression in video radio transmission for capsule endoscopy," *Sensors and Actuators A*, vol. 123–124, pp. 129-136, 2005.

[6] O. Lopez, P. Pinol, M. Martinez-Rach, M.P. Malumbres, J. Oliver,"Low-complexity 3D-DWT video encoder applicable to IPTV," *Signal Processing: Image Communication*, vol.36, pp.358–359, 2011.

[7] D.Baltieri, R.Vezzani, R.Cucchiara, A.Utasi, "Multi-view people surveillance using 3D information," *IEEE International Conference on Computer Vision Workshops*, 2011.

[8] P.Andono, E.Yuniarno, M.Hariadi, V.Venus, "3D reconstruction of under water coral reef images using low cost multi-view cameras," *2012 International Conference on Multimedia Computing and Systems*, 2012.

[9] M. van der Schaar, Y. Andreopoulos, and Z. Hu, "Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints," *IEEE Transactionns on Mobile Compuing*, vol. 5, no. 6, pp. 755–768, 2006.

[10] E. Maani, A. Katsaggelos, "Unequal Error Protection for Robust Streaming of Scalable Video Over Packet Lossy Networks," *IEEE Transaction on Circuits ans Systems for Video Technology*, vol. 20, no. 3, pp.407–416, 2010.

[11] Y.Huo, M. El-Hajjar, and L.Hanzo, "Inter-Layer FEC Aided Unequal Error Protection for Multilayer Video Transmission in Mobile TV," *IEEE Transaction on Circuits ans Systems for Video Technology*, vol. 20, no. 3, pp.1622–1634, 2013.

[12] Advanced video coding for generic audiovisual services, *ITU-T Recommendation H.264 and ISO/IEC 14496-10 (AVC)*, 2009.

[13] ISO/IEC JTC 1, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s Part 2: Video," *ISO/IEC 11172 (MPEG-1)*, 1993.

[14] ITU-T, "Video codec for audiovisual services at 64 kbits/s," *ITU-T Recommendation H.261*, 1993.

[15] ITU-T and ISO/IEC JTC 1, "Generic coding of moving pictures and associated audio information Part 2: Video," *ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2)*, 1994.

[16] ITU-T, "Video coding for low bit rate communication," *ITU-T Rec. H.263*, 2000.

[17] ISO/IEC JTC 1, "Coding of audio-visual objects Part 2: Visual," *ISO/IEC 14496-2 (MPEG-4 Visual)*, 1999.

[18] High Efficiency Video Coding, `http://www.h265.net/`

[19] X. Li, M. Wien, J.-R. Ohm, "Rate-Complexity-Distortion Optimization for Hybrid Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.21, pp.957 – 970, 2011.

[20] D.Slepian, J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, 1973.

[21] A. Wyner, J. Ziv, "The rate-distortion function for source coding with side information at the decoder,"*IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, 1976.

[22] DISCOVER codec, `http://discoverdvc.org/`

[23] A.Ukhanova, E.Belyaev, Soren Forchhammer, "Encoder power consumption comparison of Distributed Video Codec and H.264/AVC in low-complexity mode," *The 18th International Conference on Software, Telecommunications and Computer Networks*, 2010.

[24] E.Belyaev, “Low bit rate video coding based on three-dimensional discrete pseudo cosine transform,“ *International Conference on Ultra Modern Telecommunications*, 2010.

[25] J. Li, J. Takala, M. Gabbouj, and H. Chen, “Modeling of 3-D DCT Coefficients For Fast Video Encoding,“ *IEEE International Symposium on Communications, Control and Signal Processing*, pp. 634–638, 2008.

[26] H. Devos, P. Lambert, D. De Schrijver, W. Van Lancker, V. Nollet, P. Avasare, T. Clerckx, F. Verdicchio, M. Christiaens, P. Schelkens, R. Van de Walle, D. Stroobandt, “Scalable, Wavelet-Based Video: From Server to Hardware-Accelerated Client“, *IEEE Transactions on Multimedia*, Vol.9, pp.1508 – 1519, 2007.

[27] B.J. Kim, Z. Xiong, and W.A. Pearlman,“Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT),“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.10, pp.1374–1378, 2000.

[28] S. Lim J. Xu, Z. Xioong and Y. Zhang, “3-D embedded subband coding with optimal truncation (3-D ESCOT),“ *Applied and Computational Harmonic Analysis*, vol.10, pp.290–315, 2001.

[29] P. Chen and J. Woods, “Bidirectional MC-EZBC with lifting implementation,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.14, no.10,pp.1183-1194, 2004.

[30] J. Xu, F. Wu, S. Li,“Barbell-Lifting Based 3-D Wavelet Coding Scheme,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.9, pp. 1256 – 1269, 2007.

[31] J. Xu, F. Wu, S. Li, and Ya-Qin Zhang, “Subband Coupling Aware Rate Allocation for Spatial Scalability in 3-D Wavelet Video Coding,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.17, pp.1311 – 1324, 2007.

[32] J. Xuesong,L. Maoliu, Z. Zhijie, C. Ming, F. Zhipeng, “Bitrate allocation for 3-D wavelet video coder,“ *2011 3rd International Conference on Advanced Computer Control*, pp.309 – 312, 2011.

[33] D. Taubman, “High performance scalable image compression with EBCOT,“ *IEEE Transactions on Image Processing*, vol.9, pp. 1151-1170, 2000.

[34] F. Yang,S. Wan,E. Izquierdo, “Lagrange Multiplier Selection for 3-D Wavelet Based Scalable Video Coding,“ *IEEE International Conference on Image Processing*, pp.309–312, 2007.

[35] ITU-T and ISO/IEC JTC 1, “JPEG 2000 Image Coding System: Core Coding System, ITU-T Recommendation T.800 and ISO/IEC 15444-1,“ *JPEG 2000 Part 1*, 2000.

[36] J. Hua, Z. Xiong, X. Wu, “High-performance 3-D embedded wavelet video (EWV) coding,“ *2001 IEEE Fourth Workshop on Multimedia Signal Processing*, pp. 569–574, 2001.

[37] ITU-T and ISO/IEC JTC1, “Digital Compression and coding of continuoustone still images,“ *ISO/IEC 10918-1 - ITU-T Recommendation T.81 (JPEG)*, 1992.

[38] V. Sze, M. Budagavi, “Overview of the High Efficiency Video Coding (HEVC) Standard,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.22, iss.12, pp. 1649–1668, 2012.

[39] D. Marpe, T. Wiegand, “A Highly Efficient Multiplication-Free Binary Arithmetic Coder and Its Application in Video Coding,“ *IEEE International Conference on Image Processing*, 2003.

[40] E. Belyaev, M. Gilmutdinov, A. Turlikov, “Binary arithmetic coding system with adaptive probability estimation by Virtual Sliding Window,“ *10th IEEE International Symposium on Consumer Electronics*, pp.194–198, 2006.

[41] M. Schindler, “Byte oriented arithmetic coding,“ *Proceedings of Data Compression Conference*, 1998.

[42] D. Subbotin, “Carryless Rangecoder,“ 1999. `http://search.cpan.org/src/SALVA/Compress-PPMd-0.10/Coder.hpp`.

[43] P. Lindstrom, M. Isenburg, “Fast and Efficient Compression of Floating-Point Data,“ *IEEE Transactions on Visualization and Computer Graphics*, vol.12, iss.5, pp.1245 – 1250, 2006.

[44] A. Said, “Comparative analysis of arithmetic coding computational complexity,“ *Hewlett-Packard Laboratories Report*, 2004.

[45] F.Bossen, B.Bross, K.Suhring, and D.Flynn, “HEVC Complexity and Implementation Analysis,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol.22, iss.12, pp.1685 – 1696, 2012.

[46] B. Ryabko, “Imaginary sliding window as a tool for data compression,“ *Problems of Information Transmission*, pp. 156–163, 1996.

[47] E. Krichevski and V. Trofimov, “The performance of universal encoding,“ *IEEE Transactions on Information Theory*, vol.27, pp. 199-207, 1981.

[48] D. Taubman and M. Marcellin, “JPEG2000: Image Compression, Fundamentals, Standards, and Practice,“ *Kluwer Academic Publishers*, 2002.

[49] H.264/AVC JM Reference Software, `http://iphome.hhi.de/suehring/tml/`

[50] MVC test sequences, `http://www.merl.com/pub/avetro/mvc-testseq/orig-yuv/`

[51] Xiph.org test media, `http://media.xiph.org/video/derf/`

[52] x.264 video codec, `http://x264.nl/`

[53] G.M. Schuster, A.K. Katsaggelos, “Rate-Distortion Based Video Compression, Optimal Video Frame Compression, and Object Boundary Encoding,“ *Kluwer Academic Publisher*, 1997.

[54] Test video sequences for VANETs, `http://www.cs.tut.fi/~belyaev/v2v.htm`.

[55] JSVM 9.8 software package, `http://iphome.hhi.de/`

[56] B. Penna, T. Tillo, E. Magli, and G. Olmo, “Progressive 3-D coding of hyperspectral images based on JPEG 2000,“ *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 125–129, 2006.

[57] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the Scalable Video Coding Extension of the H.264 / AVC Standard,“ *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, iss. 9, pp. 1103 – 1120, 2007.

[58] E.Belyaev, A.Vinel, A.Surak, M.Gabbouj, M.Jonnson, K.Egiazarian, “Robust vehicle-to-infrastructure video transmission for road surveillance applications,“ *IEEE Transactions on Vehicular Technology*, 10.1109/TVT.2014.2354376, 2014.

Publications

[P1] E.Belyaev, A.Turlikov, K.Egiazarian and M.Gabbouj, "An efficient adaptive binary arithmetic coder with low memory requirement," *IEEE Journal of Selected Topics in Signal Processing. Special Issue on Video Coding: HEVC and beyond*, vol.7, iss.6, pp.1053–1061, 2013.

An efficient adaptive binary arithmetic coder with low memory requirement

Evgeny Belyaev, *IEEE Member*, Andrey Turlikov, Karen Egiazarian, *IEEE Senior Member* and Moncef Gabbouj, *IEEE Fellow*

***Abstract*—In this paper we propose a novel efficient adaptive binary arithmetic coder which is multiplication-free and requires no look-up tables. To achieve this, we combine the probability estimation based on a virtual sliding window with the approximation of multiplication and the use of simple operations to calculate the next approximation after the encoding of each binary symbol. We show that in comparison with the M-coder the proposed algorithm provides comparable computational complexity, less memory footprint and bitrate savings from 0.5 to 2.3% on average for H.264/AVC standard and from 0.6 to 3.6% on average for HEVC standard.**

***Index Terms*—arithmetic coding, H.264/AVC, HEVC, M-coder.**

I. Introduction

Adaptive binary arithmetic coding (ABAC) is an essential component in most common image and video compression standards and several non standardized codecs such as JPEG [1], JPEG2000 [2], H.264/AVC [3], HEVC [4] and Dirac [5]. Arithmetic coders implemented in these codecs are based on the so called Q-coder [6] which is a multiplication-free adaptive binary arithmetic coder with a bit renormalization and look-up tables used for multiplication approximation and probability estimation.

The most efficient ABAC implementation is the M-coder [7], which is the core of the Context-adaptive binary arithmetic coding (CABAC) used in the H.264/AVC standard. The emerging HEVC standard [4], [9] also uses the M-coder as an encoding engine in CABAC, but in contrast with H.264/AVC the context-modeling in HEVC is significantly simplified. Therefore, the computation complexity and memory consumption portions of the M-coder in CABAC of HEVC are higher than those in H.264/AVC.

Since the M-coder is a key component of the entropy encoding in both compression schemes, the implementation of ABAC with a smaller memory footprint, a lower computation complexity and a higher compression efficiency remains an important challenge.

There exist several approaches to improve the compression efficiency of ABAC; however, all of them require either a multiplication operation in the interval division part, or consume additional memory. In H.264/AVC and HEVC standards, CABAC first divides the input data into several non-stationary binary sources using context modeling. Next, each binary source is compressed by an M-coder which estimates the probabilities using one state machine for binary sources with different statistical properties. However, from a compression efficiency point of view, it is better to find the trade-off between adaptation speed and precision of the probability estimation for each binary source. This task can be solved by utilizing several state machines with different adaptation speeds and precision of probability estimation [5], which unfortunately leads to an increase in memory consumption for storing different look-up tables. Another solution uses look-up table-free approach based on virtual sliding window [10] (VSW). In this approach, the probability estimation is calculated using a simple rule with one parameter – window length, and a trade-off is reached by assigning a specific window length selected according to the statistical properties of the corresponding binary source. The main disadvantage of this approach is that a multiplication operation is required. Improving of compression efficiency can be also achieved by modification of the context modeling in CABAC [11], [12], [13] but it also accompanied by significant increase of computation complexity.

Decreasing the computational complexity can be achieved by modifying the renormalization procedure. In [14], a faster renormalization method is proposed, but it is based on additional look-up tables. As an alternative to arithmetic coders, *range coders* use bytes as output bit stream element and do byte renormalization at a time [15], [16], [17], [18]. In [19] it was shown that adaptive binary range coder allows to decrease computation complexity. However, range coder is not efficient for compression of small length binary sequences and it also uses a multiplication in the interval division part of ABAC, so it is not suitable for hardware applications. Additional interesting direction of the computation complexity decrease can be based on development of a CABAC with a parallelism in data processing [20], [21].

Decreasing memory consumption of CABAC is also a very important issue, especially for hardware implementation. That is why a number of proposals for HEVC standard aim at minimizing memory consumption. In [22] the new CABAC architecture was proposed. This architecture reduces the memory requirement by 67%, but at the expense of 0.25–6.84% coding loss. In [23] a reduction of initialization tables for CABAC contexts lead to coding losses up to 0.4%. In [24]

Evgeny Belyaev, Karen Egiazarian and Moncef Gabbouj are with the Department of Signal Processing, Tampere University of Technology, Finland, e-mail: {evgeny.belyaev, karen.egiazarian, moncef.gabbouj}@tut.fi

Andrey Turlikov is with the Department of Information and Communication Systems, Saint-Petersburg State University of Aerospace Instrumentation, Russia, e-mail: turlikov@vu.spb.ru

the LPS range look-up table size was reduced by 34% without any degradation of the compression efficiency. To reduce the chip area for hardware implementations in [25], the authors proposed to remove look-up tables, but had to introduce multiplications.

In this paper, we present an efficient adaptive binary arithmetic coder. The main contributions of the paper are the following:

1) A new adaptive binary arithmetic coder is proposed which, in contrast with the previous coders, does not use multiplications nor any look-up tables[1].
2) We evaluated the proposed coder's performance and show that it allows allows to easily control the trade-off between the speed of probability adaptation and the precision of probability estimation through a single parameter, the window length. For software applications we show that the proposed coder has comparable computation complexity with M-coder.
3) Two application scenarios for the proposed coder are investigated. In the first scenario, in comparison to the M-coder the proposed coder provides bitrate savings from 0.5 to 2.3% for H.264/AVC standard and 0.6–3.6% for HEVC standard. In the second scenario, a compression improvement of 0.3% in comparison to the first scenario is achieved.

The rest of the paper is organized as follows. Section II reviews the integer implementation of binary arithmetic coding. Section III is dedicated to the look-up table-free probability estimation of ones for a binary source. Section IV describes the state-of-the-art adaptive binary arithmetic encoder implementations. Section V introduces the proposed multiplication-free and look-up table-free adaptive binary arithmetic coder. Section VI evaluates the performance of the proposed coder and M-coder using three main features: coding redundancy, speed of probability adaptation and computation complexity. Two scenarios of usage of the proposed coder and comparative results with the M-coder are presented in Section VII. Conclusions are drawn in Section VIII.

II. Integer implementation of binary arithmetic coding

Let us consider a stationary discrete memoryless binary source with p denoting the probability of ones. In a binary arithmetic encoding codeword for a binary sequence $\mathbf{x}^N = \{x_1, x_2, ..., x_N\}$, $x_t \in \{0,1\}$ is represented as $\lceil -\log_2 P(\mathbf{x}^N) + 1 \rceil$ bits of a number

$$Q(\mathbf{x}^N) + P(\mathbf{x}^N)/2, \tag{1}$$

where $P(\mathbf{x}^N)$ and $Q(\mathbf{x}^N)$ are the probability and the cumulative probability of a sequence $\mathbf{x}^N$, respectively, which can be calculated by the recurrent relations:
If $x_t = 0$, then

$$\begin{cases} Q(\mathbf{x}^t) \leftarrow Q(\mathbf{x}^{t-1}) \\ P(\mathbf{x}^t) \leftarrow P(\mathbf{x}^{t-1})(1-p), \end{cases} \tag{2}$$

[1]This result was briefly presented by the authors in [26]

if $x_t = 1$, then

$$\begin{cases} Q(\mathbf{x}^t) \leftarrow Q(\mathbf{x}^{t-1}) + P(\mathbf{x}^{t-1})(1-p) \\ P(\mathbf{x}^t) \leftarrow P(\mathbf{x}^{t-1})p. \end{cases} \tag{3}$$

In this paper we use "$\leftarrow$" as the assignment operation, "$\ll$" and "$\gg$" as left and right arithmetic shift, and "!" as bitwise NOT operation.

An integer implementation of an arithmetic encoder is based on two registers: L and R size of b bits (see Algorithm 1). Register L corresponds to $Q(\mathbf{x}^N)$ and register R corresponds to $P(\mathbf{x}^N)$. The precision required to represent registers L and R grows with the increase of N. In order to decrease the coding latency and avoid registers underflow, the *renormalization* procedure [27] is used for each output symbol (see Algorithm 2).

Algorithm 1 : Binary symbol x_t encoding procedure

```
1: $T \leftarrow R \times p$
2: $T \leftarrow \max(1, T)$
3: $R \leftarrow R - T$
4: if $x_t = 1$ then
5:     $L \leftarrow L + R$
6:     $R \leftarrow T$
7: end if
8: call Renormalization procedure
```

Algorithm 2 : Renormalization procedure

```
1: while $R < 2^{b-2}$ do
2:     if $L \geq 2^{b-1}$ then
3:         $bits\_plus\_follow(1)$
4:         $L \leftarrow L - 2^{b-1}$
5:     else if $L < 2^{b-2}$ then
6:         $bits\_plus\_follow(0)$
7:     else
8:         $bits\_to\_follow \leftarrow bits\_to\_follow + 1$
9:         $L \leftarrow L - 2^{b-2}$
10:    end if
11:    $L \leftarrow L \ll 1$
12:    $R \leftarrow R \ll 1$
13: end while
```

III. Probability estimation

In real applications the probability of ones is unknown. In this case for an input binary symbol x_t the probability estimation of ones $\hat{p}_t$ is calculated and used in line 1 of Algorithm 1 instead of p. One of the well known probability estimation algorithms is based on a *sliding window* concept. The probability of a source symbol is estimated by analyzing the content of a special buffer [28]. This buffer keeps W previously encoded symbols, where W is the length of the buffer. After the encoding of each symbol the buffer's content is shifted by one position, a new symbol is written to the free cell and the earliest symbol in the buffer is erased.

For the binary sources, the probability of ones is estimated by the Krichevsky-Trofimov formula [29]:

$$\hat{p}_{t+1} = \frac{s_t + 0.5}{W+1}, \tag{4}$$

where s_t is the number of ones in the window before encoding the symbol with the index t.

The advantage of using a sliding window is the possibility of a more accurate evaluation of the source statistics and a fast adaptation to changing statistics. However, the window has to be stored in the encoder and the decoder memory, which is a serious drawback of this algorithm. To avoid this, the *Imaginary Sliding Window* technique (ISW) was proposed [28]. The ISW technique does not require storing the content of the window, and instead, it estimates symbols count from the source alphabet stored in the window.

Let us consider the ISW method for a binary source. Define $x_t \in \{0,1\}$ as a source input symbol with index t, $y_t \in \{0,1\}$ as a symbol deleted from the window after adding x_t. Suppose at every time instant a symbol in a random position is erased from the window instead of the last one. Then the number of ones in the window is recalculated by the following recurrent randomized procedure.

Step 1. Delete a random symbol from the window

$$s_{t+1} \leftarrow s_t - y_t, \tag{5}$$

where y_t is a random value generated with probabilities

$$\begin{cases} Pr\{y_t = 1\} = \dfrac{s_t}{W}, \\ Pr\{y_t = 0\} = 1 - \dfrac{s_t}{W}. \end{cases} \tag{6}$$

Step 2. Add a new symbol from the source

$$s_{t+1} \leftarrow s_{t+1} + x_t. \tag{7}$$

For the implementation of the ISW algorithm, a random variable must be generated. This random variable should take the same values at the corresponding steps of the encoder and the decoder. However, there is a way to avoid generating a random variable [10]. At step 1 of the algorithm let us replace a random value y_t with its probabilistic average. Then the rule for recalculating the number of ones after encoding of each symbol x_t can be presented in two steps.

Step 1. Delete an average number of ones from the window

$$s_{t+1} \leftarrow s_t - \frac{s_t}{W}. \tag{8}$$

Step 2. Add a new symbol from the source

$$s_{t+1} \leftarrow s_{t+1} + x_t. \tag{9}$$

By combining (8) and (9), the final rule for recalculating the number of ones can be given as follows:

$$s_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s_t + x_t. \tag{10}$$

Equation (10) corresponds to the following probability estimation rule:

$$p_{t+1} = \left(1 - \frac{1}{W}\right) \cdot p_t + \frac{1}{W} x_t. \tag{11}$$

IV. State-of-the-art adaptive binary arithmetic coder implementations

A. Multiplication-free implementation based on state machine

One way for implementation of probability estimation can be based on the *state machine* approach. Each state of this machine corresponds to some probability value. Transition from state to state is defined by the value of the input symbol. This approach does not require multiplications or divisions for probability calculation. In addition, the fixed set of states allows to implement the interval division part of the arithmetic coder without multiplications [6].

For example, let us consider a state machine based probability estimation in state-of-the-art M-coder [7] from H.264/AVC and HEVC standards. In the M-coder input symbols are divided into two types: Most Probable Symbols (MPS) and Least Probable Symbols (LPS). State machine contains 64 states. Each state s defines probability estimation for Least Probable Symbol. Set of probability values $\{\hat{p}_0, \hat{p}_1, ..., \hat{p}_{63}\}$ is defined as:

$$\begin{cases} \hat{p}_s = (1-\gamma)\hat{p}_{s-1}, \text{ where } s = 1, ..., 63, \hat{p}_0 = 0.5, \\ \gamma = 1 - \left(\dfrac{\hat{p}_{min}}{0.5}\right)^{\frac{1}{63}}, \hat{p}_{min} = 0.01875. \end{cases} \tag{12}$$

Probability estimation for symbol x_{t+1} is calculated as

$$\hat{p}_{t+1} = \begin{cases} (1-\gamma)\hat{p}_t + \gamma, \text{ if } x_t = \text{LPS}, \\ \max\{(1-\gamma)\hat{p}_t, \hat{p}_{min}\}, \text{ if } x_t = \text{MPS}, \end{cases} \tag{13}$$

and implemented by using tables *TransStateLPS*$[s]$ and *TransStateMPS*$[s]$ which contain number of the next probabilities after compression of the current symbol. It is important to notice that if we define $\gamma = 1/W$, then it is easy to see that the probability estimation rule (13) is based on rule (11).

To remove the multiplication in line 1 of Algorithm 1, M-coder uses its approximation [30]. After the renormalization procedure in Algorithm 2, register R satisfies the following inequality:

$$\frac{1}{2} 2^{b-1} \le R < 2^{b-1}. \tag{14}$$

From (14) it follows that a multiplication can be approximated in the following way:

$$T = R \times \hat{p}_t \approx \alpha 2^{b-1} \times \hat{p}_t, \tag{15}$$

where $\alpha \in [\frac{1}{2}, ..., 1)$. To improve the precision of the approximation, the M-coder quantizes the interval $[\frac{1}{2}2^{b-1}; 2^{b-1})$ uniformly to four cells. Then each multiplication of the corresponding probability $\hat{p}_s$ and the interval cell with index $\Delta \in \{0,1,2,3\}$ are stored in two-dimensional table *TabRangeLPS*$[s][\Delta]$ which contains 64×4 values.

Thus, the adaptive binary arithmetic coding algorithm in H.264/AVC and HEVC standards is implemented in the following way (see Algorithm 3).

B. Look-up table-free implementation based on virtual sliding window

Based on (10), a probability estimation using virtual sliding window was proposed in [10]. Let us multiply both sides

Algorithm 3 : Binary symbol x_t encoding procedure

```
1: $\Delta \leftarrow (R - 2^{b-2}) \gg (b-4)$
2: $T \leftarrow \mathit{TabRangeLPS}[s][\Delta]$
3: $R \leftarrow R - T$
4: if $x_i \neq$ MPS then
5:     $L \leftarrow L+R$
6:     $R \leftarrow T$
7:     if $s = 0$ then
8:         MPS $\leftarrow$!MPS;
9:     end if
10:    $s \leftarrow \mathit{TransStateLPS}[s]$
11: else
12:    $s \leftarrow \mathit{TransStateMPS}[s]$
13: end if
14: call Renormalization procedure
```

of (10) by W:

$$s'_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s'_t + Wx_t, \tag{16}$$

where $s'_t = Ws_t$. Let us define $W = 2^w$, where w is an integer positive value. After integer rounding of equation (16), we obtain

$$s'_{t+1} = \begin{cases} s'_t + \left\lfloor \dfrac{2^{2w} - s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 1 \\ \\ s'_t - \left\lfloor \dfrac{s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 0, \end{cases} \tag{17}$$

and the probability of ones is calculated as

$$\hat{p}_t = \frac{s'_t}{2^{2w}}. \tag{18}$$

Thus, the adaptive binary arithmetic coding algorithm based on virtual sliding window is presented in Algorithm 4.

Algorithm 4 : Binary symbol x_t encoding procedure

```
1: $T \leftarrow (R \times s) \gg (2w)$
2: $T \leftarrow \max(1, T)$
3: $R \leftarrow R - T$
4: if $x_t$ =1 then
5:     $L \leftarrow L+R$
6:     $R \leftarrow T$
7:     $s \leftarrow s + ((2^{2w} - s + 2^{w-1}) \gg w)$
8: else
9:     $s \leftarrow s - ((s + 2^{w-1}) \gg w)$
10: end if
11: call Renormalization procedure
```

In comparison with the M-coder, Algorithm 4 provides a better compression efficiency due to an assignment of a specific virtual sliding window length selected according to the statistical properties of the corresponding binary source [10] and using multiplication in interval division part of the arithmetic coder instead of its approximation. Nevertheless, using this multiplications is not efficient, especially for a hardware implementation.

V. The proposed adaptive binary arithmetic coder

To eliminate the multiplication operation from Algorithm 4, we use a similar reasoning as the one described in Section IV. Let us multiply both sides of (10) by $\alpha 2^{b-1}$:

$$s'_{t+1} = \left(1 - \frac{1}{W}\right) \cdot s'_t + \alpha 2^{b-1} x_t, \tag{19}$$

where $s'_t = \alpha 2^{b-1} s_t$. After integer rounding of equation (19), we obtain

$$s'_{t+1} = \begin{cases} s'_t + \left\lfloor \dfrac{\alpha 2^{b-1} 2^w - s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 1 \\ \\ s'_t - \left\lfloor \dfrac{s'_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_t = 0, \end{cases} \tag{20}$$

Taking into account (14) and (15)

$$T = R \times \hat{p}_t \approx \alpha 2^{b-1} \times \hat{p}_t = \frac{s'_t}{2^w}. \tag{21}$$

To improve precision of the approximation (as in M-coder) we quantize the interval $[\frac{1}{2}2^{b-1}; 2^{b-1})$ to four points:

$$\left\{ \frac{9}{16}2^{b-1}, \frac{11}{16}2^{b-1}, \frac{13}{16}2^{b-1}, \frac{15}{16}2^{b-1} \right\}. \tag{22}$$

To implement this, we first calculate a state s'_t using (20) for $\alpha = \frac{9}{16}$. Then we approximate the multiplication in the following way:

$$T = R \times \hat{p}_t \approx \frac{s'_t + \Delta \times \frac{1}{4}s'_t}{2^w}, \text{ where } \Delta = \frac{R - 2^{b-2}}{2^{b-4}}. \tag{23}$$

Taking into account that the approximation of the multiplication is correct for $\hat{p}_t < \frac{2}{3}$ [30], we should work with $\hat{p}_t \in [0, .., 0.5]$ and use the Most Probable Symbol and the Least Probable Symbol. In our case, the MPS value should be changed if

$$\hat{p}_t = \frac{s'_t}{2^w} \frac{1}{\alpha 2^{b-1}} > 0.5, \tag{24}$$

or

$$s'_t > \alpha 2^{b-2} 2^w. \tag{25}$$

Thus, taking into account (20), (23) and (25), an adaptive binary arithmetic coding is proposed in the following way (see Algorithm 5):

Since $\Delta \in \{0, 1, 2, 3\}$, a multiplication in line 2 of Algorithm 5 can be implemented based on conditional and addition operations.

From Algorithm 5, it follows that if initial state $s_0 \geq 2^{w-1} - 1$, then the minimum value of the probability estimation is

$$\hat{p}_{min} = \frac{2^{w-1} - 1}{\alpha \cdot 2^{b-1} \cdot 2^w}. \tag{26}$$

Algorithm 5 : Binary symbol x_i encoding procedure

```
1: $\Delta \leftarrow (R - 2^{b-2}) \gg (b - 4)$
2: $T \leftarrow (s + \Delta \times (s \gg 2)) \gg w$
3: $T \leftarrow \max(1, T)$
4: $R \leftarrow R - T$
5: if $x_i \neq$ MPS then
6:     $L \leftarrow L+R$
7:     $R \leftarrow T$
8:     $s \leftarrow s + ((\alpha 2^{b-1} 2^w - s + 2^{w-1}) \gg w)$
9:     if $s > \alpha 2^{b-2} 2^w$ then
10:        MPS ←!MPS;
11:        $s \leftarrow \alpha 2^{b-2} 2^w$;
12:    end if
13: else
14:    $s \leftarrow s - ((s + 2^{w-1}) \gg w)$
15: end if
16: call Renormalization procedure
```

VI. Performance evaluation of the proposed coder

A. Coding redundancy analysis

Let us assume that a stationary binary memoryless source with probability of ones p is to be compressed. Then the probability estimation process can be represented by the Markov chain with a finite number of states. Therefore, a coding redundancy $r(p)$ caused by the probability estimation precision can be calculated as:

$$r(p) = \sum_i \pi(s_i) \cdot \Big(-(1-p) \cdot \log_2(1-\hat{p}_i) - p \cdot \log_2 \hat{p}_i \Big) - h(p), \quad (27)$$

where $\pi(s_i)$ is a stationary probability of state s_i, $\hat{p}_i$ is a probability estimation value for this state, and $h(p)$ is the entropy of the source.

In practice, a coding redundancy also depends on the number of bits b for representation of registers L and R [31]. Taking this into account, we estimate the coding redundancy as

$$r(p) = \lim_{N \to \infty} \left(\frac{R}{N} \right) - h(p), \quad (28)$$

where N is the number of input binary symbols and R is the size of the compressed bit stream.

Table I show the coding redundancy (28) related to the binary entropy for the M-coder and the proposed coder with window lengths $W = 2^4$, 2^5 and 2^6, respectively. For both coders we set the values $b = 10$ and $N = 10^8$.

First, one can see that the coding redundancy for the proposed algorithm depends on the window length W: longer window length provides less redundancy. Second, in most of the cases, in comparison with the M-coder the proposed coder exhibits a higher redundancy if $W = 2^4$ and a lower redundancy if $W = 2^6$. For $W = 2^5$ it has a higher redundancy for low probabilities and a lower redundancy for high probabilities of ones. Finnaly, Table II shows the minimum probability estimation value $\hat{p}_{min}$ for both coders. One can see that the proposed coder provides significantly less $\hat{p}_{min}$ value and; therefore, it is more efficient for compressing low entropy sources.

TABLE I
Coding redundancy $r(p)$ for the M-coder and the proposed coder

p	M-coder	Proposed coder $W = 4$	Proposed coder $W = 5$	Proposed coder $W = 6$
0	0.029	0.0039	0.0039	0.0039
10^{-5}	0.0287	0.0037	0.0037	0.0037
10^{-4}	0.0281	0.0034	0.0033	0.0033
10^{-3}	0.0239	0.0021	0.0019	0.0016
10^{-2}	0.0102	0.011	0.0078	0.0052
0.02	0.008	0.023	0.015	0.008
0.03	0.009	0.03	0.016	0.007
0.04	0.012	0.034	0.017	0.008
0.06	0.017	0.034	0.015	0.006
0.08	0.02	0.033	0.014	0.007
0.1	0.021	0.031	0.014	0.006
0.2	0.021	0.027	0.013	0.007
0.3	0.022	0.028	0.014	0.007
0.4	0.02	0.024	0.013	0.008
0.5	0.02	0.02	0.01	0

TABLE II
Minimum probability estimation value for the M-coder and the proposed coder

coder	$\hat{p}_{min}$
M-coder	0.01875
Proposed coder, $W = 2^4$	0.00152
Proposed coder, $W = 2^5$	0.00163
Proposed coder, $W = 2^6$	0.00168

B. Speed of probability adaptation analysis

In the case of non-stationary source compression, the coding efficiency is highly depend on the speed of the probability adaptation. For quantitative comparison for different probability estimation algorithms, we propose the following approach. First, the initial value of the probability estimation is set to $\hat{p}_0 = 0.5$. Then, the binary source with probability of ones $p < 0.5$ is generated and compressed. Finally, the speed of the probability adaptation is defined as the inverse of the average number of binary symbols t which are needed for the first attainment of probability estimation value $\hat{p}_t \leq p$.

Table III shows the average number of binary symbols for the M-coder and the proposed coder with window lengths $W = 2^4$, 2^5 and 2^6. First, one can see that the adaptation speed for the proposed algorithm depends on the window length W: a shorter window length provides a higher adaptation speeds. Second, the adaptation speed of the M-coder is less than the one for the proposed coder with $W = 2^4$ and higher for $W = 2^5$ and $W = 2^6$.

C. Computation complexity analysis

From Algorithms 1, 3, 4 and 5 it follows, that the number of operations in the interval division part of an arithmetic coder is directly proportional to the number of input binary symbols.

TABLE III
THE AVERAGE NUMBER OF BINARY SYMBOLS t WHICH IS NEEDED TO TRANSFER FROM $\hat{p}_0 = 0.5$ TO $\hat{p}_t \leq p$

p	M-coder	Proposed coder $W = 4$	Proposed coder $W = 5$	Proposed coder $W = 6$
0.45	19	11	31	71
0.4	25	18	45	104
0.3	34	28	63	145
0.2	44	35	80	181
0.1	54	44	99	219
0.05	66	52	115	249
0.02	76	62	133	283

On the other hand, the number of times steps in lines 2–12 are used is proportional to the number of bits in the output bit stream (see Algorithm 2). Let us define N as a number of the input binary symbols, R as a size of the output bit stream. Then, the complexity per input symbol for the binary arithmetic coder C can be written as follows [19]:

$$C = \frac{\alpha \cdot N + \beta \cdot R}{N}, \tag{29}$$

where α is the computation complexity of the interval division part per input binary symbol, β is the computation complexity of the renormalization part per output binary symbol. For arithmetic coder the output bit stream size

$$R = N \cdot \big(h(p) + r(p)\big), \tag{30}$$

where $h(p)$ is an entropy of binary memoryless source with probability of ones p, $r(p)$ is an coding redundancy. Therefore, the complexity per input symbol is a linear function of the source entropy and coding redundancy:

$$C(p) = \alpha + \beta \cdot (h(p) + r(p)) \, . \tag{31}$$

From (31) it follows, that for zero-entropy source ($h(p) = 0$) the complexity $C(p)$ is the lowest possible and is mainly determined by the complexity of the division part. With an increase of $h(p)$ the renormalization part is used more and more often; therefore, the complexity also increases and reaches the maximum value when $h(p) = 1$. Moreover, from the model (31) it follows, that if everything else being equal, the compression scheme which has a higher coding redundancy has a higher complexity.

Table IV shows the complexity $C(p)$ for the M-coder and the proposed coder. In this work, as in many papers related to complexity comparisons (see, for example [32], [33]), we measure the complexity in CPU cycles. For the measurements both coders were implemented as separate software programs which include the encoding of a binary string only. For each probability p we have generated $N = 10^8$ input binary symbols and, then, the CPU cycles of Processor Intel Core i3M 2.1 GHz was measured by the *Average CPU Cycles* software[2]

First, one can observe that the complexity for the proposed algorithm depends on the window length W, which determines different coding redundancies: a longer window length provides less redundancy and, as it is shown in the model (31), less complexity. Second, the proposed coder has up to 18% less computation complexity for very low entropy sources, because its minimum possible probability estimation $\hat{p}_{min}$ is more than 10 times smaller than in the M-coder (see Table II). Therefore it has less redundancy $r(p)$ and, following model (31), the renormalization step is not so often used as in the case of the M-coder, which explains the lower complexity. Finally, for the remaining probabilities, the complexity difference is in the range of $\pm 5\%$, which means that in these cases the complexities of both coders are comparable.

[2]http://user.tninet.se/~jad615g/averagecpu/

TABLE IV
CPU CYCLES PER SYMBOL OF PROCESSOR INTEL CORE I3M 2.1 GHZ FOR M-CODER AND THE PROPOSED CODER

p	0	0.05	0.1	0.2	0.3	0.4	0.5
Proposed coder, $W = 4$	15.5	28.3	35.4	45.1	51.7	56.5	56.6
Proposed coder, $W = 5$	15.5	27.6	34.7	44.5	50.2	55.2	55.3
Proposed coder, $W = 6$	15.5	27.4	34.3	43.4	49.1	54.3	54.0
Proposed coder, average	15.5	27.8	34.8	44.3	50.3	55.3	55.3
M-coder	19.0	28.1	35.5	45.9	52.3	55.3	53.7
Average gain, %	***18.8***	***1.1***	***2.1***	***3.5***	***3.8***	***0.0***	***-3.0***

VII. APPLICATION OF THE PROPOSED CODER IN H.264/AVC AND HEVC STANDARDS

A. Scenario 1: Using fixed window length for all contexts

For practical experiments the proposed adaptive binary arithmetic coder was embedded into the JM codec v.12.1 [34] which is the reference software of the H.264/AVC video coding standard and into the HM 3.0 reference software [8] of the HEVC video coding standard. JM codec was used the Main profile, while HM codec was running with the low-delay configuration.

The initial state for each binary context was calculated as:

$$s_0 = \max\left\{2^{w-1} - 1, \left\lfloor \alpha 2^{b-1} 2^w \hat{p}_0 + 0.5 \right\rfloor\right\} \tag{32}$$

where $\hat{p}_0$ is the initial probability predefined in the standards.

In fact, the predefined initial probability can differ significantly from the best initial probability for the current binary source from the compression efficiency point of view. Therefore, it is important to use a high speed of probability adaptation at the initial stage of the coding, especially in the case of compression of short binary sequences. Then it is better to switch to a probability estimation with a lower adaptation speed, but with a better precision of the probability estimation. This approach is adopted in JPEG2000 [2] utilizing three adaptation mechanisms, which are embedded into one state machine. It has also been used in [10] by applying Krichevsky-Trofimov formula at the initial stage of the coding. However, in both cases, an additional table or a division operation is needed.

In this paper we implement a similar idea in the following way. First, for a binary sequence compression a small window length $W = 2^w$ is used. After the compression of n_1 binary symbols the window length and the state s are increasing two times to $W = 2^{w+1}$ and $2s$. Then, after the compression of n_2 binary symbols the window length and the state are increased two times again, and so on, until the maximum window length is achieved. In this case, as it is shown in Section VI, a high speed of probability adaptation is achieved by setting small window lengths at the initial stage of coding and, then, a high probability estimation precision is achieved by setting longer window lengths.

Bitrate savings in percentage related to the M-coder for different quantization parameters are presented in Tables V–VI. In all cases the reconstructed video for the corresponding quantization parameter (QP) is identical for the M-coder and for the proposed coder. Practical results were obtained for the first 60 frames of the test video sequences [35], [36] with different frame resolutions: 352×288 ("Foreman", "Mobile", "Akyio", "Mother Daughter"), 640×480 ("Vassar", "Ballroom"), 704×576 ("City", "Crew") and 1920×1080 ("Pedestrian Area", "Rush Hour", "Station", "Tractor"). In our experiments, in case of H.264/AVC, for all contexts an initial window length is $W = 2^4$, the maximum window length is $W = 2^6$ and the values $n_1 = 24$ and $n_2 = 48$. In case of HEVC, an initial window length is $W = 2^3$, the maximum window length is $W = 2^6$ and the values $n_1 = 12$, $n_2 = 24$ and $n_3 = 48$.

The results show that for fixed visual quality the proposed adaptive binary arithmetic coder allows to decrease the bitrate by 0.5–2.3% for H.264/AVC standard and 0.6–3.6% for HEVC standard on average. Herewith, for low QP's (0–30) the bitrate savings is caused mostly by less coding redundancy of window length $W = 2^6$, while for high QP's (40–50) the savings results mostly from the higher speed of the probability adaptation, which is very crucial for encoding of short binary sequences. It is important to notice, that for an implementation of the proposed approach additional tables or multiplication/division operations are not needed. Therefore, in this scenario the computation complexity is comparable with the M-coder.

TABLE V
BITRATE SAVINGS (IN %) COMPARED TO THE M-CODER FOR H.264/AVC STANDARD IN CASE OF USING FIXED WINDOW LENGTH

QP	0	10	20	30	40	50
Foreman	0.93	0.46	0.43	0.10	0.02	0.15
Mobile	0.04	0.17	0.30	0.46	0.20	0.49
Akyio	0.85	0.43	0.12	-0.17	0.68	5.04
Mother Daughter	1.14	0.81	0.56	0.06	0.44	4.81
Vassar	1.35	0.77	0.81	0.42	1.72	8.25
Ballroom	1.23	0.66	0.62	0.50	0.51	0.84
City	0.79	0.53	0.58	0.73	0.60	1.24
Crew	1.01	0.43	0.73	0.59	0.79	1.58
Pedestrian Area	1.14	0.69	0.72	0.79	1.07	1.47
Rush Hour	1.19	0.87	0.88	0.98	0.97	1.46
Station	1.39	1.06	1.07	0.59	0.70	1.20
Tractor	1.02	0.53	0.61	0.79	0.84	0.96
Average	**1.01**	**0.62**	**0.62**	**0.49**	**0.71**	**2.29**

B. Scenario 2: Adaptive window length selection for each context

As it was mentioned in the introduction, from a compression efficiency point of view, it is better to find the trade-off between the adaptation speed and the precision of the probability estimation for each binary source. In the proposed adaptive binary arithmetic coder, this trade-off can be realized due to an assignment of a specific window length selected according to the statistical properties of the corresponding binary source.

TABLE VI
BITRATE SAVINGS (IN %) COMPARED TO THE M-CODER FOR HEVC STANDARD IN CASE OF USING FIXED WINDOW LENGTH

QP	0	10	20	30	40	50
Foreman	0.64	0.58	0.32	0.18	0.80	2.76
Mobile	0.45	0.35	0.52	0.72	1.23	2.80
Akyio	0.75	0.73	0.63	0.74	2.07	6.18
Mother Daughter	0.74	1.06	0.89	0.89	2.21	5.54
Vassar	0.49	0.29	0.96	0.73	1.57	6.27
Ballroom	0.40	0.27	0.44	0.46	1.01	1.93
City	0.52	0.56	0.91	1.04	1.94	3.37
Crew	0.58	0.59	0.78	0.52	1.18	3.98
Pedestrian Area	0.72	0.75	0.61	0.66	0.87	1.54
Rush Hour	0.79	0.91	0.75	0.82	0.91	1.66
Station	0.78	1.01	0.87	0.62	0.95	2.72
Tractor	0.48	0.52	0.62	1.38	2.73	4.31
Average	**0.61**	**0.63**	**0.69**	**0.73**	**1.46**	**3.59**

In this paper we proposed to select the window length adaptively during the encoding and decoding process. Let us demonstrate this approach for the H.264/AVC standard. First $n_1 + n_2$ binary symbols are coded and decoded by the basic scheme described in Section VII-A. Then after the processing each binary symbol, the encoder and the decoder synchronously estimate the bit stream size $\hat{r}(w)$ for each window length from the set $W = 2^w \in \{2^4, 2^5, 2^6, 2^7\}$. To accomplish this, Algorithm 5 with the virtual renormalization procedure (see Algorithm 6) is applied for each window length. Then after processing of n_3 symbols the encoder and the decoder uses the window length

$$w^* = \arg \min_{i \in \{w\}} \hat{r}(i) \tag{33}$$

until the end of the binary sequence.

Algorithm 6 : Virtual renormalization procedure

```
 1: while $R(w) < 2^{b-2}$ do
 2:     if $L(w) \geq 2^{b-1}$ then
 3:         $\hat{r}(w) \leftarrow \hat{r}(w) + 1$
 4:         $L(w) \leftarrow L - 2^{b-1}$
 5:     else if $L(w) < 2^{b-2}$ then
 6:         $\hat{r}(w) \leftarrow \hat{r}(w) + 1$
 7:     else
 8:         $\hat{r}(w) \leftarrow \hat{r}(w) + 1$
 9:         $L(w) \leftarrow L(w) - 2^{b-2}$
10:     end if
11:     $L(w) \leftarrow L(w) \ll 1$
12:     $R(w) \leftarrow R(w) \ll 1$
13: end while
```

Bitrate savings related to the M-coder in the case of adaptive window length selection with $n_3 = 512$ are presented in Table VII. One can see that using different window lengths decreases up to 0.3% the bitrate additionally in comparison to the first scenario. On the other hand, because use of virtual renormalization, the complexity of this approach is much higher than the one in Scenario 1.

TABLE VII
BITRATE SAVINGS (IN %) COMPARED TO THE M-CODER IN CASE OF ADAPTIVE WINDOW LENGTH SELECTION FOR EACH CONTEXT

QP	0	10	20	30	40	50
Foreman	1.01	0.56	0.44	0.10	0.02	0.15
Mobile	0.05	0.25	0.37	0.50	0.21	0.48
Akyio	0.89	0.45	0.12	-0.17	0.68	5.04
Mother Daughter	1.26	0.92	0.59	0.06	0.39	4.81
Vassar	1.56	0.94	1.00	0.43	1.63	8.33
Ballroom	1.42	0.83	0.71	0.51	0.52	0.83
City	0.88	0.63	0.73	0.82	0.72	1.32
Crew	1.14	0.60	0.85	0.63	0.82	1.61
Pedestrian Area	1.34	0.90	0.88	0.91	1.22	1.68
Rush Hour	1.41	1.09	1.06	1.13	1.11	1.75
Station	1.69	1.30	1.35	0.70	0.87	1.42
Tractor	1.18	0.70	0.76	0.91	0.93	1.11
Average	**1.15**	**0.76**	**0.74**	**0.55**	**0.76**	**2.38**

VIII. CONCLUSION

A new efficient multiplication-free and look-up table-free adaptive binary arithmetic coder was presented. In comparison to the M-coder it has the following advantages:

1) It does not use any look-up tables allowing a reduction in memory consumption or chip area in a hardware implementation.
2) It provides a simply mechanism to control the trade-off between the speed of probability adaption and the precision of probability estimation through a single parameter, the window length.
3) It provides bitrate savings from 0.5 to 2.3% for H.264/AVC standard and from 0.6 to 3.6% for HEVC standard on average and has a comparable computational complexity.
4) It does not require any changes in the context-modeling and can be easily embedded for performance improvement to any compression scheme which is based on binary arithmetic coding.

Taking into account these advantages, the proposed coder can be more preferable for a future image and video coding standards or non standardized codecs.

REFERENCES

[1] ITU-T and ISO/IEC JTC1, “Digital Compression and cod- ing of continuous-tone still images“, ISO/IEC 10918-1 - ITU-T Recommendation T.81 (JPEG), 1992.

[2] ITU-T and ISO/IEC JTC 1, “JPEG 2000 Image Coding System: Core Coding System, ITU-T Recommendation T.800 and ISO/IEC 15444-1“ *JPEG 2000 Part 1*, 2000.

[3] D. Marpe, H. Schwarz, T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard“, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.7. pp.620–636, 2003.

[4] V. Sze, M. Budagavi, “Overview of the High Efficiency Video Coding (HEVC) Standard“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.22, Iss.12, pp. 1649–1668, 2012.

[5] H. Eeckhaut, B. Schrauwen, M. Christiaens, J. Van Campenhout, “Tuning the M-coder to improve dirac’s entropy coding“, *WSEAS transactions on Information Science and Applications*, pp. 1563–1571, 2005.

[6] W. Pennebaker, J. Mitchel, G. Langdon, R. Arps, “ An overview of the basic principles of the q-coder adaptive binary arithmetic coder“, *IBM J. Research and Development*, vol.32, pp.717–726, 1988.

[7] D. Marpe, T. Wiegand, “A Highly Efficient Multiplication-Free Binary Arithmetic Coder and Its Application in Video Coding“, *IEEE International Conference on Image Processing*, 2003.

[8] High Efficiency Video Coding, http://www.h265.net/

[9] V. Sze, M. Budagavi, “High Throughput CABAC Entropy Coding in HEVC“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.22, Iss.12, pp. 1778–1791, 2012.

[10] E. Belyaev, M. Gilmutdinov, A. Turlikov, “Binary arithmetic coding system with adaptive probability estimation by Virtual Sliding Window“, *Proceedings of the 10th IEEE International Symposium on Consumer Electronics*, pp.194–198, 2006.

[11] D. Karwowski, M. Domanski , “Improved context-adaptive arithmetic coding in H.264/AVC“, *17th European Signal Processing Conference*, 2009.

[12] T. Nguyen, H. Schwarz, H. Kirchhoffer, D. Marpe, T. Wiegand, “Improved context modeling for coding quantized transform coefficients in video compression“, *Picture Coding Symposium*, 2010.

[13] K. Vermeirsch, J. Barbarien, P. Lambert, R. Van de Walle, “Region-adaptive probability model selection for the arithmetic coding of video texture“, *2011 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011.

[14] D. Hong, A. Eleftheriadis, “Memory-Efficient Semi-Quasi Renormalization for Arithmetic Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, pp. 106–109, 2007.

[15] M. Schindler, “Byte oriented arithmetic coding“, *Proceedings of Data Compression Conference*, 1998.

[16] D. Subbotin, “Carryless Rangecoder“, 1999. http://search.cpan.org/src/SALVA/Compress-PPMd-0.10/Coder.hpp.

[17] P. Lindstrom, M. Isenburg, “Fast and Efficient Compression of Floating-Point Data“, *IEEE Transactions on Visualization and Computer Graphics*, Vol.12, Iss.5, pp.1245 – 1250, 2006.

[18] A. Said, “Comparative analysis of arithmetic coding computational complexity“, Hewlett-Packard Laboratories Report, HPL-2004-75, 2004.

[19] E. Belyaev, A. Veselov, A. Turlikov and Kai Liu, “Complexity analysis of adaptive binary arithmetic coding software implementations“, *The 11th International Conference on Next Generation Wired/Wireless Advanced Networking*, 2011.

[20] S. Chen, S. Chen, S. Sun, “P3-CABAC: A Nonstandard Tri-Thread Parallel Evolution of CABAC in the Manycore Era“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.20, Iss.6, pp.920 – 924, 2010.

[21] K. Sarawadekar, S. Banerjee, “An Efficient Pass-Parallel Architecture for Embedded Block Coder in JPEG 2000“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.21, Iss.6, pp.825 – 836, 2011.

[22] V. Sze, A.Chandrakasan, Joint Algorithm-Architecture Optimization of CABAC to Increase Speed and Reduce Area Cost, *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011.

[23] K.Sugimoto, A.Minezawa, S.Sekiguchi, K.Asai, T.Murakami, Reduction of initialization tables for CABAC contexts, *Input Document to JCT-VC JCTVC-H0646*, 2012.

[24] T.Chuang, C.Chen, Y.Huang, and S.Lei, CABAC with a Reduced LPS Range Table, *Input Document to JCT-VC JCTVC-F061*, 2011.

[25] C. Rosewarne, Modified probability update and table removal for multi-parameter CABAC update, *Input Document to JCT-VC JCTVC-H0140*, 2012.

[26] E.Belyaev, A.Turlikov, K.Egiazarian and M.Gabbouj, An efficient multiplication-free and look-up table-free adaptive binary arithmetic coder // *2012 IEEE International Conference on Image Processing*, 2012.

[27] A. Moffat, R. Neal, I. Witten, “Arithmetic Coding Revisited“, *ACM Transactions on Information Systems*, vol. 16, pp. 256–294, 1998.

[28] B. Ryabko, “Imaginary sliding window as a tool for data compression“, *Problems of Information Transmission*, pp. 156–163, 1996.

[29] E. Krichevski and V. Trofimov, “The performance of universal encoding“, *IEEE Transactions on Information Theory*, vol. IT-27, pp. 199-207, 1981.

[30] D. Taubman and M. Marcellin, “JPEG2000: Image Compression, Fundamentals, Standards, and Practice“, *Kluwer Academic Publishers*, 2002.

[31] B.Y. Ryabko, A. N. Fionov, “An efficient method for adaptive arithmetic coding of sources with large alphabets“, *Problems of Information Transmission*, vol.35, No. 4, pp. 95–108, 1999.

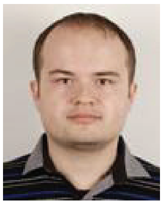
[32] C.Grecos, M.Yang, “Fast inter mode prediction for P slices in the H264 video coding standard“, *IEEE Transaction on Broadcasting*, vol.51, No. 2, pp. 256–253, 2005.

[33] W.Lee, H.Choi, S.Wonyong, “Fast Block Mode Decision for H.264/AVC on a Programmable Digital Signal Processor“, *IEEE Workshop on Signal Processing Systems*, 2007.

[34] H.264/AVC JM Reference Software, http://iphome.hhi.de/suehring/tml/

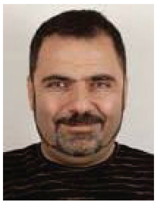
[35] MVC test sequences, http://www.merl.com/pub/avetro/mvc-testseq/orig-yuv/

[36] Xiph.org test media, http://media.xiph.org/video/derf/



Evgeny Belyaev (M'12) received the Engineer degree in automated systems of information processing and control and the Ph.D. (candidate of science) degree in technical sciences from State University of Aerospace Instrumentation (SUAI), Saint-Petersburg, Russia, in 2005 and 2009, respectively. He is currently a Researcher with the Institute of Signal Processing, Tampere University of Technology, Finland. His research interests include real-time video compression and transmission, video source rate control, scalable video coding, motion estimation and arithmetic encoding.

Andrey Turlikov is a professor at the Department of Information and Communication Systems of State University of Aerospace Instrumentation, Saint-Petersburg, Russia. Since 1987 he has been involved in teaching activity. He is the author of more than 100 research papers and has been the invited speaker at the number of international symposiums and seminars. His research interests cover multi-user telecommunication systems, real-time data transmission protocols, reliability theory, and video compression algorithms.



Karen Egiazarian (SM'96) was born in Yerevan, Armenia, in 1959. He received the M.Sc. degree in mathematics from Yerevan State University in 1981, the Ph.D. degree in physics and mathematics from Moscow State University, Moscow, Russia, in 1986, and the D.Tech. degree from the Tampere University of Technology (TUT), Tampere, Finland, in 1994. He has been Senior Researcher with the Department of Digital Signal Processing, Institute of Information Problems and Automation, National Academy of Sciences of Armenia. Since 1996, he has been an Assistant Professor with the Institute of Signal Processing, TUT, where he is currently a Professor, leading the Computational Imaging Group. His research interests are in the areas of applied mathematics, signal processing, and digital logic.

Moncef Gabbouj (F'11) received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, in 1985, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, in 1986 and 1989,respectively.

He is currently an Academy of Finland Professor with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He was the Head of the department from 2002 to 2007. He was on sabbatical leave at the American University of Sharjah, Sharjah, United Arab Emirates, from 2007 to 2008, and a Senior Research Fellow with the Academy of Finland, Helsinki, Finland, from 1997 to 1998 and in 2008. He is the Co-Founder and Past CEO with SuviSoft Oy, Ltd., Tampere. From 1995 to 1998, he was a Professor with the Department of Information Technology, Pori School of Technology and Economics, Pori, Finland, and from 1997 to 1998 he was a Senior Research Scientist with the Academy of Finland. From 1994 to 1995, he was an Associate Professor with the Signal Processing Laboratory, Tampere University of Technology. From 1990 to 1993, he was a Senior Research Scientist with the Research Institute for Information Technology, Tampere. His current research interests include multimedia content-based analysis, indexing and retrieval, nonlinear signal and image processing and analysis, and video processing and coding.

A low-complexity bit-plane entropy coding and rate control for 3-D DWT based video coding

Evgeny Belyaev, *IEEE Member*, Karen Egiazarian, *IEEE Senior Member* and Moncef Gabbouj, *IEEE Fellow*

***Abstract*—This paper is dedicated to fast video coding based on three-dimensional discrete wavelet transform. First, we propose a novel low-complexity bit-plane entropy coding of wavelet subbands based on Levenstein zero-run coder for low entropy contexts and adaptive binary range coder for other contexts. Second, we propose a rate-distortion efficient criterion for skipping 2-D wavelet transforms and entropy encoding based on parent-child subband tree. Finally, we propose one pass rate control which uses virtual buffer concept for adaptive Lagrange multiplier selection.**

Simulations results show that the proposed video codec has a much lower computational complexity (from 2 to 6 times) for the same quality level compared to the H.264/AVC standard in the low complexity mode.

***Index Terms*—3-D DWT, entropy coding, rate control**

I. Introduction

Video coding and transmission are core technologies used in numerous applications such as streaming, surveillance, conferencing, broadcasting and so on. Taking into account high bit error raties, packet losses and time-varying bandwidth, the *scalable video coding* (SVC) is the most preferable compression method for video transmission [1], [2].

A scalable extension of the H.264/SVC standard [3], which is currently the most popular video coding approach, includes temporal, spatial and quality scalability and provides high compression efficiency due to motion compensation and inter-layer prediction exploiting the video source temporal redundancy and redundancy between different layers.

However, high computational complexity is a significant barrier for usage of the H.264/SVC on mobile devices, personal computers and other systems such as high definition video surveillance, wireless home TV, mobile IPTV broadcast etc. [4], [5], [6], which compress one or more high resolution video sources in real-time. This becomes an even greater a challenge in future real-time scalable multiview video coding systems which must be able to handle several views simultaneously.

The high computational complexity of H.264/SVC, as well as other hybrid video coding algorithms, are due to the following main reasons:

1) Motion compensation and estimation require a lot of computations even if fast motion vector searching algorithms are used. Additional problem occur in case of high resolution video sequences which requires high motion search radius for efficient coding in rate-distortion sense. This contradicts with low complexity coding case which requires as less as possible motion search radius. Therefore, in real-time applications the motion estimation can be not so efficient, especially taking into account that in some cases the compressed video sources does not contain significant motion.
2) To avoid resynchronization with the decoder side the motion compensation, as well as other encoder parts like intra-prediction, require backward loop in the encoding scheme. This loop includes inverse transform and quantization, deblocking filter and other extra calculations.
3) Macro-block coding mode selection as well as rate control also requires additional computations.

In real-life applications the H.264/SVC encoder should be significantly simplified, resulting in a decrease of its rate-distortion performance [7]. However, even after simplifications, listed above encoding parts have relatively high computational complexity. Taking into account that the future video coding standard HEVC [8] (High Efficiency Video Coding) is also based on hybrid video coding, its scalable extension is likely to have a high computational complexity too.

As an alternative to H.264/SVC encoders, scalable video encoder based on three-dimensional discrete wavelet transform (3-D DWT) can be used. During the last decade, several 3-D DWT approaches were proposed such as 3-D Set partitioning in hierarchical trees (3-D SPIHT) [9], 3-D Embedded subband coding with optimal truncation (3-D ESCOT) [10], Embedded video coding using zeroblocks of motion compensated 3-D subband/wavelet coefficients (MC-EZBC) [11] and so on. Due to intensive investigation of Motion Compensated Temporal Filtering (MCTF), currently 3-D DWT schemes have a comparable rate-distortion performance with the H.264/SVC [12]. However, as in the previous case, these schemes require also high computational resources. Therefore, development of an efficient low-complexity scalable video coders is an important practical problem.

In [5], a real-time 3-D run-length wavelet video encoder was proposed, which, however, does not include a rate control algorithm which is a key component of any video codec. In [6] and [13], new rate control algorithms based on an extension of the Embedded Block Coding with Optimized Truncation (EBCOT) [14] were proposed. These rate controllers minimize distortion for a given bit rate budget, but require a lot of computations due to the use of lossless compression of wavelet subbands requiring Lagrange multiplier selection. Other works propose rate control algorithms based on heuristics and additional video source characteristics (see, for example, [15]) and also require extra computations.

Evgeny Belyaev, Karen Egiazarian and Moncef Gabbouj are with the Department of Signal Processing, Tampere University of Technology, Finland, e-mail: {evgeny.belyaev, karen.egiazarian, moncef.gabbouj}@tut.fi

The main contribution of this work is a scalable low-complexity video codec based on 3-D DWT efficiently optimizing the rate-distortion-complexity trade-off and comprising the following tools:

1) The proposed codec uses two entropy encoding cores: Levenstein zero-run coder for low entropy binary contexts and adaptive binary range coder (ABRC) [16] for the remaining binary contexts. Comparing to the traditional bit-plane encoders, e.g. [17], it allows to decrease the computation complexity without any significant rate-distortion performance degradation.
2) We propose an efficient rate-distortion criterion for skipping 2-D wavelet transforms and entropy encoding based on parent-child subbands tree. Using this criterion, the simulations results show that the computational complexity of the proposed codec tends to the complexity of the 1-D temporal transform since as the compression ratio increases, more and more 2-D subbands are skipped and 2-D transforms and entropy coding are only applied to at most 5% of the 2-D subbands.
3) We propose a one-pass rate control which uses the virtual buffer concept in order to estimate the Lagrange multiplier without resorting to lossless compressing or any additional computation.

The rest of the paper is organized as follows.

Section II presents a briefly review and categorization of existing wavelet-based video codecs along with a discussion of their computational complexity requirements. Section III describes a basic simplified JPEG2000 entropy encoding algorithm of wavelet subbands. Section IV introduces a low-complexity bit-plane entropy coding and parent-child skip criterion. Section V proposes the one-pass rate control based on adaptive Lagrange multiplier selection using the virtual buffer concept. The rate-distortion-complexity comparisons for the proposed codec, HEVC reference software, 3-D SPIHT and existing real-time and non real-time software implementations of H.264/AVC standard are presented in Section VI. Finally, conclusions are drawn in Section VII.

II. Overview of 3-D DWT Video Coding Schemes

From a video data decorrelation point of view, wavelet-based video coding schemes in the literature can be grouped in four major classes as follows:

1) The class of volumetric coding techniques, which treat the video sequence as a volume and perform a 3-D DWT followed by entropy coding of the resulting subbands [18], [19];
2) The class of in-band motion-compensated temporal filtering (MCTF) schemes performing first spatial discrete wavelet transforms followed by in-band MCTF and encoding of the resulting spatio-temporal subbands [20], [21];
3) The class of spatial-domain MCTF schemes which apply MCTF on the original image data and then transform and encode the residuals using a critically-sampled wavelet transform [11];

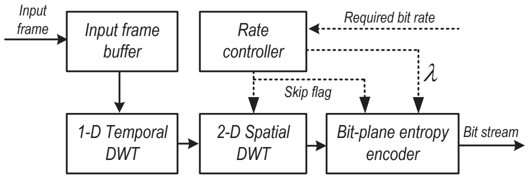


Fig. 1: Considered video coding scheme based on 3-D DWT

4) An extension of 2) and 3), a generic spatio-temporal wavelet decomposition structure alternating spatial transforms, MCTF and subsequent spatial transforms [22].

As computational complexity plays a key role, we shall next briefly evaluate the four classes in this regard. The main disadvantage of the second class is the necessity to calculate the spatial wavelet transform for each frame, even for video sequences which have group of frames (GOF) with low motion or without any motion. For such type of sequences, it is obvious that it is better to apply a temporal wavelet transform first, and then, depending on the motion level, one may skip spatial transform for several high-frequency frames in the GOF.

The third class and its extension (the fourth class) require motion compensation and motion estimation, which, as was mentioned in the introduction, improve rate-distortion performance but require a high computational complexity.

Following the reasoning above, for low complexity video coding, we consider in this paper the first class of wavelet-based video coding schemes and we use a simplified 3-D extension of the JPEG 2000 standard [23], [24] as the basic video coder. The adopted scheme is shown in Figure 1.

First, a group of frames (GOF) of length N are accumulated in the input frame buffer. Then, one-dimensional multilevel DWT of length N in the temporal direction is applied. All frames in the GOF are processed starting from low-frequency to high-frequency frames. For each frame, the spatial subbands are also processed from low-frequency to high-frequency spatial subbands.

Depending on a required bit rate and motion level in the current GOF, the rate controller chooses one of the following options for each spatial subband:

1) It permits calculation of 2-D spatial transform, selects the Lagrange multiplier λ (which defines the rate-distortion trade-off for the subband) and then permits entropy encoding for the subband;
2) It prohibits calculation of the 2-D spatial DWT and entropy coding for the subband. At the decoder side, the corresponding transform coefficients in the subband are considered to be zero.

III. Basic subband entropy encoder

Each wavelet subband is independently compressed using bit-plane entropy coding with a simplified JPEG2000 context modeling to split bit-planes into a set of binary sources.

TABLE I: Bit-plane contexts model for wavelet subband

	LL and LH subbands			HL subbands			HH subbands	
c	H	V	D	H	V	D	$(H+V)$	D
0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	1	0
2	0	0	2	0	0	2	2	0
3	0	1	–	1	0	–	0	1
4	0	2	–	2	0	–	1	1
5	1	0	0	0	1	0	2	1
6	1	0	≥ 1	0	1	≥ 1	0	2
7	1	≥ 1	–	≥ 1	1	–	1	2
8	2	–	–	–	2	–	2	2
9	bit is first after significant bit							
10	bit is not first after significant bit							
11	sign of a coefficient							

After splitting, each binary source is compressed by an adaptive binary arithmetic coder from H.264/AVC standard [3] (M-coder) which is faster and more efficient than the MQ-coder used in JPEG2000 standard [25].

Let us define $x[i,j]$ as a value of the coefficient with coordinates (i,j) in a wavelet subband of size $S_w \times S_h$, $s[i,j]$ as a significance flag, $f[i,j]$ as a flag which shows that the current bit is (or is not) first after the significant bit. The proposed bit-plane entropy encoding procedure is given in Algorithm 1.

First, the number of significant bit-planes in subband n_{max} is calculated and encoded, subband distortion D is calculated and flags $s[i,j]$ and $f[i,j]$ are set to zero. Then each coefficient is processed from the highest to the lowest bit plane in a raster scan. For each coefficient $x[i,j]$ the current bit value bit in bit-plane n is determined[1].

If $x[i,j]$ is not significant ($s[i,j]=0$), then the number of significant neighbors H, V, D are calculated, context number c is determined as it is shown in Table I and bit value is compressed by an adaptive binary arithmetic encoder corresponding to the context number c. If $bit = 1$ then $x[i,j]$ becomes significant. In this case the significance flag $s[i,j]$ is modified and the coefficient sign is compressed.

If $x[i,j]$ is significant ($s[i,j]=1$), then the current bit is compressed depending on the position after a significant bit. Finally, the subband distortion D is recalculated.

After encoding of each bit plane the bit stream size R of the subband is determined. If the Lagrange sum $D+\lambda R$ for the bit plane n is greater than the same sum for the bit plane $n+1$, then the bit stream is truncated at the bit plane $n+1$. Otherwise, encoding is continued.

Algorithm 1 Subband encoding procedure

```
Input: Subband $\{x[i,j]\}$, $\lambda$
 1: $n_{max} \leftarrow \left\lfloor \log_2(\max_{(i,j)} x[i,j]) \right\rfloor$, $D \leftarrow \sum_{(i,j)} x^2[i,j]$
 2: $\forall(i,j): s[i,j] \leftarrow 0,\ f[i,j] \leftarrow 0$
 3: arithmetic_encode($n_{max}$)
 4: $\psi \leftarrow D$
 5: for $n = n_{max}$,..,0 do
 6:   for $i = 0$,..,$S_h - 1$ do
 7:     for $j = 0$,..,$S_w - 1$ do
 8:       if $|x[i,j]| \& 2^n \neq 0$ then
 9:         $bit \leftarrow 1$
10:       else
11:         $bit \leftarrow 0$
12:       end if
13:       if $s[i,j] = 0$ then
14:         $H \leftarrow s[i–1,j] + s[i+1,j]$
15:         $V \leftarrow s[i,j–1] + s[i,j+1]$
16:         $D \leftarrow s[i–1,j–1] + s[i+1,j+1]$
17:         $c \leftarrow$ context_model($H,V,D$)
18:         arithmetic_encode($bit$,$c$)
19:         if $bit = 1$ then
20:           $s[i,j] \leftarrow 1$
21:           arithmetic_encode($sign(x[i,j])$,11)
22:         end if
23:       else
24:         arithmetic_encode($bit$, 9+$f[i,j]$)
25:         $f[i,j] \leftarrow 1$
26:       end if
27:       $D \leftarrow D - ((|x[i,j]| \& (2^{n+1}–1))^2$
28:       $D \leftarrow D + ((|x[i,j]| \& (2^{n}–1))^2$
29:     end for
30:   end for
31:   $R \leftarrow$ arithmetic_get_bit_stream_size()
32:   if $\psi < D + \lambda R$ then
33:     stop encoding
34:   else
35:     $\psi \leftarrow D + \lambda R$
36:   end if
37: end for
```

IV. Proposed low-complexity entropy coder

A. Combined entropy coding

In comparison with the M-coder from H.264/AVC standard, the adaptive binary range coder (ABRC) proposed in [16] with the probability estimation based on a virtual sliding window has less computational complexity due to use of bytes as an output bit stream element and byte renormalization; and better compression efficiency due to the assignment of a specific window length selected according to the statistical properties of the corresponding binary source. Therefore, the computational complexity of the bit-plane entropy coder, described in section II, can be reduced due to the replacement of the M-coder by the ABRC. As shown in Figure 5, using the ABRC increases the encoding speed of 3-D DWT coder from 20% to 30% on average with an increase in compression performance. The encoding speed in this paper is defined as the number of frames which can be encoded in one second on the hardware platform with processor Intel Core 2 DUO CPU 3.0GHz. The computational complexity is considered as inverse value of the encoding speed.

In order to further decrease the computational complexity, let us consider the following properties of different binary contexts. Figure 2 shows the typical fraction of the context $c = 0$ in all input binary data depending on Peak signal-to-

[1]Here we use "$\leftarrow$" as the assignment operation, and & as arithmetic 'and' operation

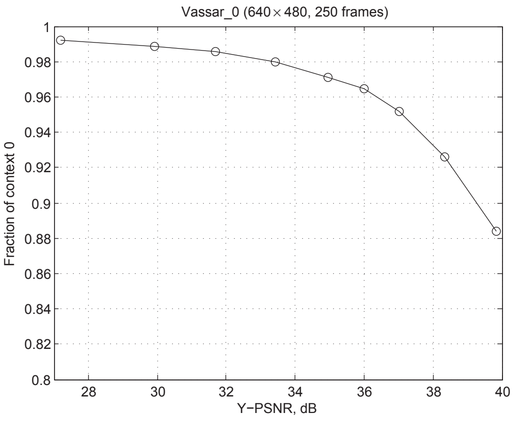


Fig. 2: Fraction of context 0 in all input binary data

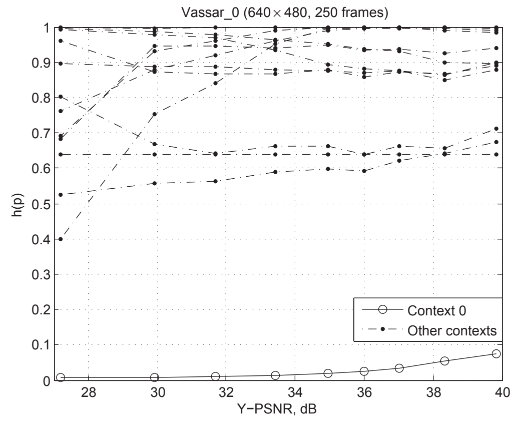


Fig. 3: Binary entropy estimation for different contexts

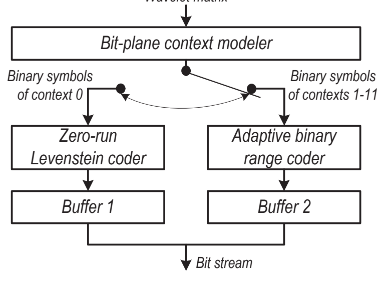

Wavelet matrix
Bit-plane context modeler
Binary symbols of context 0
Binary symbols of contexts 1-11
Zero-run Levenstein coder
Adaptive binary range coder
Buffer 1
Buffer 2
Bit stream

Fig. 4: Proposed combined entropy coder

noise ratio of the luma component (Y-PSNR) for the entropy encoder given in section II. Figure 3 shows a typical binary memoryless entropy estimation for different binary contexts. On can see that context $c = 0$ has more than 88% fraction in all binary input data for any Y-PSNR. At the same time a binary entropy of the source which corresponds to the context $c = 0$ does not exceeds 0.1 and in most cases, it is significanly less than that. This means that the source can be efficiently compressed by zero-run length coding. In this case the compression efficiency is not decreasing and, on the other hand, computational complexity is decreasing because of the ABRC compresses 12% or less of the binary input data. Taking into account the reasoning above, we propose the following combined bit-plane entropy encoding algorithm (see Fig. 4). The binary sequence corresponding to the context $c = 0$ is compressed by the zero-run length coding and placed into Buffer 1. The other binary sequences are compressed by the ABRC and placed into the Buffer 2[2]. For zero-run coding we propose to use Levenstein codes with a prefix based on equal codes. It helps to avoid using look-up tables during the coding process in comparison with the Huffman codes or CAVLC in H.264/AVC that is more efficient from the computational complexity point of view.

For compression of zero-run $\underbrace{000..0}_{L}1$ value m is calculated:

$$m = \begin{cases} 0, \text{ if } L = 0 \\ \lfloor \log_2 L \rfloor + 1, \text{ if } L > 0. \end{cases} \quad (1)$$

Then the value L is represented by the following two parts:

1) $\lfloor \log_2(\log_2 L_{max}) \rfloor + 1$ bits for binary representation of m;
2) $m - 1$ bits for binary representation of $L - 2^{m-1}$,

where L_{max} is the maximum possible zero-run value.

As shown in Figure 5, the proposed combined-coder allows to increase the encoding speed of 3-D DWT coder from 77% to 85% on average in comparison with the one using the M-coder.

B. Skipping of 2-D wavelet transform and entropy encoding

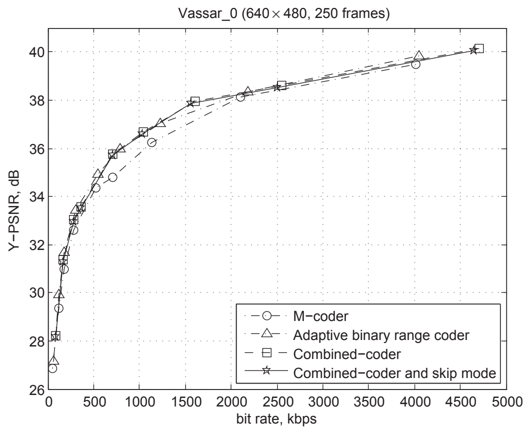
From Algorithm 1, it follows that if

$$D(n_{max}) + \lambda R(n_{max}) > \sum_{(i,j)} x^2[i,j], \quad (2)$$

where $D(n_{max})$ and $R(n_{max})$ are subband distortion and bit stream size after highest significant bit-plane encoding, then this subband is skipped. In this situation the encoder wastes computational resources for calculating the 2-D DWT and for forming the subband bit stream which is not included in the output bit stream. Therefore, it is important to find a criterion which can guarantee with a high probability that the current subband will be skipped *before* processing of this subband.

In this paper we propose the parent-child tree based subband skip criterion. It is well known that the value of the wavelet coefficient in the child subband is correlated with the

[2]A bit-plane entropy encoding with two buffers was proposed first by the authors in [26], but in this work contexts with a high entropy are not compressed at all



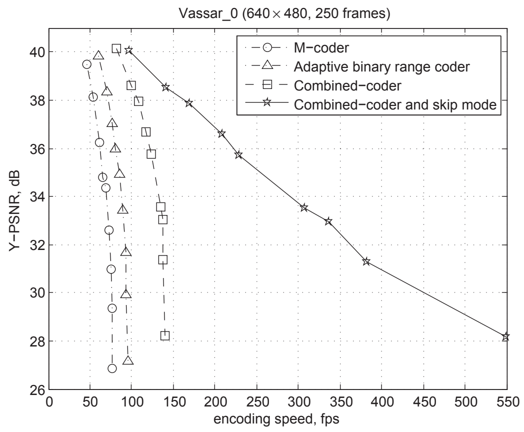


Fig. 5: Rate-distortion-complexity comparison of different bit-plane entropy encoding approaches

corresponding coefficient in the parent subband. This property is widely used in image and video compression algorithms based on inter-subband correlation (see, e.g. [9]). Using the same reasoning, we first process subbands from low-frequency to high-frequency temporal subbands and from low-frequency to high-frequency spatial subbands. Second, we make the following *coding assumption*: if for any subband, inequality (2) holds, then for all temporal-spatial child-subbands, the same inequality also holds. In this case, all child-subbands will not be processed and hence spatial transform calculation and entropy coding steps are skipped. At the decoder side, the corresponding transform coefficients are considered to be zero.

Figure 6 illustrates the proposed approach for a GOF size $N = 4$ with two-level temporal and two-level spatial wavelet decomposition. Frames after temporal wavelet decomposition are denoted as $L3, H2, H1, H0$, where $L3$ is the low-frequency frame. If the spatial subband $HH1$ in the frame $H2$ is skipped then the corresponding child-subbands $HH0$ in the frame $H2$, $HH1$ and $HH0$ in the frame $H1$, and $HH1$ and $HH0$ in the frame $H0$ are skipped without any processing.

Figure 7 shows the fraction of skipped wavelet subbands depending on the Y-PSNR for the different test video sequences. One can easily notice that this fraction is highly dependent on the motion level in the video sequence (compare for instance the results for Football which contains high motion with Vtc1nw which contains low motion). On the other hand, in all cases, as Y-PSNR decreases (or the compression ratio increases), the fraction of skipped subbands increases and reaches 95% in the limit. Therefore, as the compression ration increases, the computational complexity of the proposed algorithm tends to the complexity of the 1-D temporal transform.

As shown in Figure 5, the proposed combined-coder with a parent-child tree skipping allows to increase the encoding speed of 3-D DWT coder from 2 to 7 times in comparison with one using the M-coder without any significant degradation of the rate-distortion performance.

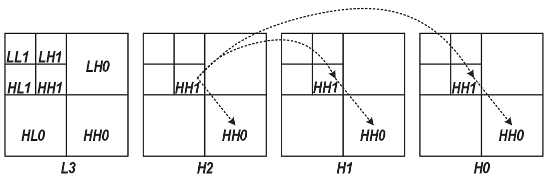


Fig. 6: Example of parent-child skipping tree

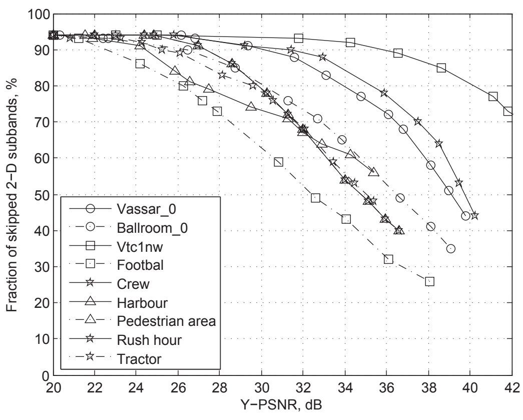


Fig. 7: Efficiency of the proposed parent-child tree based subband skip criterion

V. Proposed low-complexity rate control

A. *Optimal rate allocation based on Lagrangian relaxation*

Consider $\mathbf{n} = \{n_i\}$ to be the *truncation vector*, where n_i signifies that i'th subband in a GOF is truncated after encoding of bit-plane n_i. Denote the overall GOF distortion after the truncation as

$$D(\mathbf{n}) = \sum_{i=0}^{i_{max}-1} w_i \cdot d_i^{n_i}, \tag{3}$$

where quantity $d_i^{n_i}$ corresponds to the distortion value of i-th subband truncated in the bit-plane number n_i, w_i is the weighting distortion coefficient of the ith subband. Similarly, denote the resulting GOF bit stream size as

$$R(\mathbf{n}) = \sum_{i=0}^{i_{max}-1} r_i^{n_i}. \tag{4}$$

Then for each GOF we should determine the truncation vector $\mathbf{n}^*$, so that

$$\begin{cases} \mathbf{n}^* = \arg \min\limits_{\mathbf{n} \in \{\mathbf{n}\}} D(\mathbf{n}) \\ R(\mathbf{n}^*) \leq R_{max}, \end{cases} \tag{5}$$

where R_{max} is the bit budget for each GOF which is defined as

$$R_{max} = \frac{N \cdot C}{f}, \tag{6}$$

where N is the number of frames in the GOF, C is the required bit rate, and f is the video source frame rate.

The rate control task (5) can be solved using the concept of Lagrangian relaxation [27]. It can be proven, that for each $\lambda \geq 0$, the set of truncation vectors $\mathbf{n}_\lambda^*$ minimizing

$$D(\mathbf{n}) + \lambda \cdot R(\mathbf{n}) \tag{7}$$

is the solution of the rate control task (5) for $R_{max} = R(\mathbf{n}_\lambda^*)$.

From this statement it follows that to solve (5) we should find λ, so that $R(\mathbf{n}_\lambda^*) = R_{max}$. It can be proven, see [27], that $R(\mathbf{n}_\lambda^*)$ is a non-increasing function of λ, i.e. λ can be found by bisection method.

For the considered codec, the rate-distortion function for the subband i is independent of the truncation of other subbands; therefore,

$$\min_{\mathbf{n}}\{D(\mathbf{n}) + \lambda \cdot R(\mathbf{n})\} = \sum_i \min_{n_i}\{w_i \cdot d_i^{n_i} + \lambda \cdot r_i^{n_i}\}, \tag{8}$$

It means that a solution of the rate control task (5) can be written as $\mathbf{n}^* = \{n_0^*, n_1^*, ...\}$, where

$$n_i^* = \arg\min_{n_i}\{w_i \cdot d_i^{n_i} + \lambda \cdot r_i^{n_i}\}. \tag{9}$$

B. *One-pass rate control based on adaptive Lagrangian multiplier selection*

To find the necessary λ value with the bisection method, we need to losslessly compress each subband in order to determine all sets of truncation points, which requires significant computational resources. As an alternative to the bisection method the rate control based on progressive construction of the convex hull of the rate-distortion function can be used [28]. In this approach encoder progressively adds bit-planes of all subbands in the GOF (starting from the highest bit-planes) until the required bit budget R_{max} is reached. But in this approach, the calculation of 2-D wavelet transforms for all subbands is needed in order to determine the number of significant bit-planes in each subband. It contradicts the 2-D wavelet transform calculation skipping proposed in Section IV-B.

In this paper we propose the one-pass rate control algorithm which does not require calculation of 2-D wavelet transform for all subbands in the GOF and uses the *virtual buffer* [29] concept to estimate λ without lossless coding. Let us define b_{virt} as the number of bits in a virtual buffer and B_{max} as the virtual buffer size determined as

$$B_{max} = L \cdot C, \tag{10}$$

where L is the required buffering latency [30] at the decoder side for continuous video playback. Then the rate control can be proposed as shown in Algorithm 2.

Algorithm 2 One-pass rate control procedure

```
 1: $b_{virt} \leftarrow b_{virt}(0)$
 2: for $j = 0,..,j_{max} - 1$ do
 3:     $\lambda \leftarrow \lambda_{max} \left( \frac{b_{virt}}{B_{max}} \right)^\gamma$
 4:     for $i = 0,..,i_{max} - 1$ do
 5:         For subband $i$ in GOF $j$ find
 6:         $n_i^* = \arg \min\limits_{n_i \in \{n_i\}} \{w_i \cdot d_i^{n_i} + \lambda \cdot r_i^{n_i}\}$.
 7:         $b_{virt} \leftarrow b_{virt} + r_i^{n_i^*}$
 8:     end for
 9:     $b_{virt} \leftarrow b_{virt} - R_{max}$
10: end for
```

At step 3, Algorithm 2 selects the Lagrange multiplier λ value in the proportion to the virtual buffer fullness, where γ defines the propotion degree. It allows to increase or decrease the compression ratio for the current GOF depending on the buffer fullness. Then for subband i in GOF j the truncation point n_i is selected according to (9). Minimization of $\psi(n_i) = w_i \cdot d_i^{n_i} + \lambda \cdot r_i^{n_i}$ is provided by steps 32–36 of Algorithm 1 with the assumption that $\psi(n_i)$ is a convex function.

Note that if during the encoding process of a video sequence the number of bits in the virtual buffer requires

$$0 \leq b_{virt} \leq B_{max}, \tag{11}$$

then the average bit stream size for each GOF $\bar{R}(\mathbf{n})$ is bounded by

$$R_{max} \leq \bar{R}(\mathbf{n}) \leq R_{max} + \frac{B_{max}}{j_{max}}, \tag{12}$$

where j_{max} is the number of GOF's in a video sequence. From (12) it follows that if (11) holds during the encoding process, then the average bit stream size for each GOF $\bar{R}(\mathbf{n})$ tends to R_{max} with an increasing j_{max}.

To further explain the proposed rate control Algorithm 2, let us suppose that the video encoder compresses the series of identical GOF's. Let us define λ^* as the Lagrange multiplier value computed using the bisection method described in

Section V-A and consider the virtual buffer fullness when the initial buffer fullness is

$$b_{virt}(0) = b^*_{virt}(0) = B_{max}\left(\frac{\lambda^*}{\lambda_{max}}\right)^{\frac{1}{\gamma}}. \qquad (13)$$

Taking into account that for the bisection method $R(\lambda^*) = R_{max}$, after processing each GOF, the buffer fullness will be constant and equal to $b_{virt} = b^*_{virt}(0)$.

Now let us analyze the buffer fullness in the case when the initial buffer fullness $b_{virt}(0) \neq b^*_{virt}(0)$. Let us assume that after compression of the previous GOF the current buffer fullness is $b_{virt} < b^*_{virt}(0)$. It means that λ value calculated by line 3 of Algorithm 2 for the current GOF will be less than the optimal λ^*. Taking into account that $R(\lambda)$ is a non-increasing function of λ, after compression of this GOF the difference $R(\lambda) - R_{max} > 0$. Therefore, a new buffer fullness calculated by line 9 will be higher then previous one. It means that after compression of the next GOF, the current value of λ will be closer to the optimal λ^* and so on, until the current value becomes equal or exceeds λ^*.

Analogically, if the current $b_{virt} > b^*_{virt}(0)$, then after compression of the next GOF the current value of λ will also be closer to the optimal λ^* and so on, until the current λ value becomes equal or smaller than λ^*.

Taking into account the reasoning described above, the value of λ calculated by the proposed algorithm will oscillate around the optimal value λ^*.

As an example, Figure 8 shows the Lagrange multiplier selected by the bisection method and the proposed rate control for different initial virtual buffer fullness. One can see that, in practice, after some adaptation period, the value of the λ approaches the optimal one, computed by the bisection method.

It is important to notice, that in comparison with other heuristic rate control techniques, the proposed algorithm does not require any additional computations and provides close to the optimal Lagrange multiplier as in the bisection method.

VI. Rate-distortion-complexity comparisons

Simulation results were obtained for the test video sequences [32], [33] with different frame resolutions: 640×480 (“Vassar_0“, “Ballroom_0“, “Vtc1nw“ and “Football“), 704×576 (“Crew“ and “Harbour“) and 1920×1080 (“Pedestrian Area“, “Rush Hour“ and “Tractor“).

For our experiments, the proposed video coding algorithm is compared with the x.264 codec [31] which, as shown in [7], provides close to optimum rate-distortion performance for the H.264/AVC standard when computational complexity is significantly restricted. Therefore, this codec can be used as an upper bound of rate-distortion performance which can be achieved by H.264/AVC standard in a low complexity case.

The proposed codec was run with GOF size $N = 16$ using the Haar wavelet transform in the temporal direction and the 5/3 spatial lifting wavelet transform at three-levels of the decomposition[3]. x.264 codec was run in very low complexity mode which corresponds to the Baseline profile of H.264/AVC with intra-frame period 16[4].

[3]The proposed codec can be found at http://www.cs.tut.fi/~belyaev/3d_dwt.htm

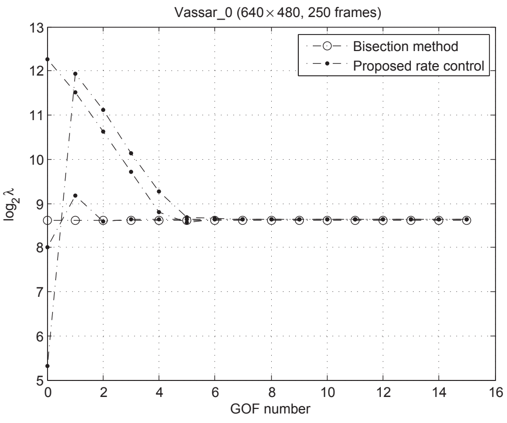


Fig. 8: Lagrange multiplier selection

Additionally, for the first four test video sequences with low frame resolution we have measured the rate-distortion-complexity performance for the reference software of the HEVC standard (HM version 9.1 in low delay configuration), 3-D SPIHT codec, x.264 codec in high complexity mode[5] (which corresponds to the High profile of the H.264/AVC standard) and x.264 codec in very low complexity mode with Intra-frame coding only[6]. In all cases, the codecs were simulated using a constant bit rate mode and the encoding speed was measured without any use of assemblers, threads, or other program optimization techniques.

Simulation results, presented in Figures 9–11, show the rate-distortion-complexity comparison for the considered codecs. Additional computation complexity comparisons based on CPU cycles measurements in Tables II–V are presented.

From Figures 9–10, it can be seen that the highly complex video codecs (HEVC, 3-D SPIHT and x.264 in high complexity mode) provide significantly better rate-distortion performance than low complexity video codecs and, as mentioned in the introduction, the reduction in complexity causes a significant degradation in the rate distortion performance of H.264/AVC, especially for sequences with high level of motion (for example, more than 4.5 dB for Football).

Furthermore, the proposed 3-D DWT video codec provides from 2 to 6 times lower computational complexity for the same Y-PSNR level compared to the H.264/AVC in the low complexity mode, when the number of frames which can be encoded in one second is used as the complexity metric (see the encoding speed in Figures 9–11), and from 2 to 11

[4]Command line example: x264.exe –output vassar_0.264 vassar_0.avs –preset ultrafast –keyint 16 –bitrate 1000 –no-asm –threads 1

[5]Command line example: x264.exe –output vassar_0.264 vassar_0.avs –preset veryslow –keyint 16 –bitrate 1000 –no-asm –threads 1

[6]Command line example: x264.exe –output vassar_0.264 vassar_0.avs –preset ultrafast –keyint 1 –bitrate 1000 –no-asm –threads 1

times lower computational complexity, when CPU cycles are used as the complexity metric (see Tables II–V). The second complexity metric does not takes into account the time needed for memory access. Therefore the maximum value of the complexity gain measured by this metric is higher than the one measured using the first metric.

For visual assessment we depicted two frames from different video sequences: frame 45 of video sequence "Ballroom_0", when the required bit rate is 500 kbps (see Figures 12–17), and frame 45 of video sequence "Football", when the required bit rate is 1000 kbps (see Figures 18–23). One can see, that the visual quality for the proposed codec is comparable with x.264 in the low-complexity mode (see Figures 14 and 16 or Figures 20 and 22) and significantly better than x.264 in the INTRA mode (see Figures 17 and 23).

VII. Conclusion

In this paper a real-time scalable video codec based on 3-D DWT with a low-complexity bit-plane entropy coding and rate control was presented. Simulations showed that the proposed codec exhibits a lower computational complexity compared to the most efficient software implementation of H.264/AVC with comparable rate-distortion performance.

TABLE II: CPU cycles for "Vassar_0"

Required bit rate, kbps	CPU cycles for x.264 ultrafast	CPU cycles for the proposed 3-D DWT	Computation complexity gain, times
4000	34726795256	13973569441	2.49
1000	24600141863	6374008375	3.86
500	19930517294	4441827624	4.49
250	17903180874	3788785063	4.73

TABLE III: CPU cycles for "Ballroom_0"

Required bit rate, kbps	CPU cycles for x.264 ultrafast	CPU cycles for the proposed 3-D DWT	Computation complexity gain, times
4000	35260887063	15350501412	2.30
1000	25358697723	7136767272	3.55
500	23043466383	5199825170	4.43
250	20842157619	3645741812	5.72

TABLE IV: CPU cycles for "Vtc1nw"

Required bit rate, kbps	CPU cycles for x.264 ultrafast	CPU cycles for the proposed 3-D DWT	Computation complexity gain, times
1000	34040990912	11345717783	3.00
300	24937234723	5135501986	4.86
100	23773185119	3149471200	7.55
50	23717868942	2033751456	11.66

The proposed scheme has a much lower computational complexity than the low complexity profile of H.264/AVC due to the following reasons:

1) It does not require backward loop in encoding scheme, because motion compensation, intra-prediction, deblocking filter and so on are not used and distortion caused by quantization can be estimated in wavelet domain without this loop.

TABLE V: CPU cycles for "Football"

Required bit rate, kbps	CPU cycles for x.264 ultrafast	CPU cycles for the proposed 3-D DWT	Computation complexity gain, times
6000	56263721904	25530553051	2.20
2000	44709404954	12346266905	3.62
1000	40324442231	8264572477	4.88
500	38642897407	6948690816	5.56

2) The proposed entropy encoder uses very simple zero-run coding based on Levenstein codes for low entropy contexts. It helps avoid using any coding tables as in CAVLC used in H.264/AVC. The remaining contexts are compressed by ABRC [16]. Taking into account that ABRC is used approximately only for 12% of the input binary data, its complexity does not affects significantly the total complexity while achieving a high compression performance.
3) The proposed parent-child tree based subband skip criterion allows skipping the 2-D wavelet transform calculation and encoding without any additional computations. The use of this criterion does not lead to a significant degradation of the rate-distortion performance and, with increasing of the compression ratio, more and more spatial subbands are skipped. Therefore, the computational complexity of the proposed codec tends to the complexity of the 1-D temporal transform. It is important to notice that the same idea cannot be effectively implemented in the H.264/AVC which requires extra analysis for motion estimation, coding mode selection, and so on, even after simplifications.
4) The proposed one-pass rate control does not require any additional computations for video signal analysis and provides a near optimal Lagrange multiplier if the statistical properties of the frames in neighboring GOFs are similar.

In this paper the proposed codec is not compared with a scalable extension of the H.264/AVC standard, because the authors have not found any open source real-time software implementation of it. But, taking into account that H.264/SVC has lower rate-distortion performance [34] and higher computation complexity (due to additional inter-layer prediction, input frame downsampling etc.) than H.264/AVC single-layer coding, our scalable video compression scheme can be more preferable than H.264/SVC in many applications where computational complexity and scalability plays a critical role.

The proposed entropy encoding and rate control can be also easily adopted for spatial-domain motion-compensated temporal filtering based video coding and, combined with a complexity scalable motion compensation [35], can achieve better rate-distortion-complexity performance for high complexity encoding modes.

References

[1] H. Devos, P. Lambert, D. De Schrijver, W. Van Lancker, V. Nollet, P. Avasare, T. Clerckx, F. Verdicchio, M. Christiaens, P. Schelkens, R. Van de Walle, D. Stroobandt, "Scalable, Wavelet-Based Video: From Server to Hardware-Accelerated Client", *IEEE Transactions on Multimedia*, Vol.9, pp.1508 – 1519, 2007.

[2] M. van der Schaar, Y. Andreopoulos, and Z. Hu, “Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints“, *IEEE Transactionns on Mobile Compuing*, vol. 5, no. 6, pp. 755–768, 2006.

[3] Advanced video coding for generic audiovisual services, *ITU-T Recommendation H.264 and ISO/IEC 14496-10 (AVC)*, 2009.

[4] Z. Gu,W. Lin, B. Lee and C. Lau, “Low-Complexity Video Coding Based on Two-Dimensional Singular Value Decomposition“, *IEEE Transactions on Image Processing*, pp.674-687, 2012.

[5] O. Lopez, P. Piol, M. Martinez-Rach, M.P. Malumbres, J. Oliver,“Low-complexity 3D-DWT video encoder applicable to IPTV“, *Signal Processing: Image Communication*, Vol.36, pp.358–359, 2011.

[6] J. Xu, F. Wu, S. Li, and Ya-Qin Zhang, “Subband Coupling Aware Rate Allocation for Spatial Scalability in 3-D Wavelet Video Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.17, pp.1311 – 1324, 2007.

[7] X. Li, M. Wien, J.-R. Ohm, “Rate-Complexity-Distortion Optimization for Hybrid Video Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.21, pp.957 – 970, 2011.

[8] High Efficiency Video Coding, http://www.h265.net/

[9] B.J. Kim, Z. Xiong, and W.A. Pearlman,“Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.10, pp.1374–1378, 2000.

[10] S. Lim J. Xu, Z. Xioong and Y. Zhang, “3-D embedded subband coding with optimal truncation (3-D ESCOT)“, Applied and Computational Harmonic Analysis, Vol.10, pp.290–315, 2001.

[11] P. Chen and J. Woods, “Bidirectional MC-EZBC with lifting implementation“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.14, No.10,pp.1183-1194, 2004.

[12] J. Xu, F. Wu, S. Li, Barbell-Lifting Based 3-D Wavelet Coding Scheme, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.9, pp. 1256 – 1269, 2007.

[13] J. Xuesong,L. Maoliu, Z. Zhijie, C. Ming, F. Zhipeng, “Bitrate allocation for 3-D wavelet video coder“, *2011 3rd International Conference on Advanced Computer Control*, pp.309-312, 2011.

[14] D. Taubman, “High performance scalable image compression with EBCOT“, *IEEE Transactions on Image Processing*, Vol.9, pp. 1151-1170, 2000.

[15] F. Yang,S. Wan,E. Izquierdo, “Lagrange Multiplier Selection for 3-D Wavelet Based Scalable Video Coding“, *IEEE International Conference on Image Processing*, pp.309-312, 2007.

[16] E. Belyaev, A. Veselov, A. Turlikov and Kai Liu, “Complexity analysis of adaptive binary arithmetic coding software implementations“, *The 11th International Conference on Next Generation Wired/Wireless Advanced Networking*, 2011.

[17] J. Hua, Z. Xiong, X. Wu, “High-performance 3-D embedded wavelet video (EWV) coding“, *2001 IEEE Fourth Workshop on Multimedia Signal Processing*, pp. 569–574, 2001.

[18] D. Taubman and A. Zakhor, “Multi-rate 3-D subband coding of video“, *IEEE Transactions on Image Processing*, 1994.

[19] J.Y. Tham, S. Ranganath, and A.A. Kassim, “Highly scalable wavelet-based video codec for very low bit-rate environment“, *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 12-27, 1998.

[20] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. Van der Schaar, J. Cornelis, and P. Schelkens, “In-band motion compensated temporal filtering“, *Signal Processing: Image Communication*, vol. 19, no. 7, pp. 653-673, 2004.

[21] D. Zhang, W. Zhang, J. Xu, F. Wu, H. Xiong, “A cross-resolution leaky prediction scheme for in-band wavelet video coding with spatial scalability“, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 4, pp. 516-521, 2008.

[22] N. Mehrseresht and D. Taubman, “A flexible structure for fully scalable motion-compensated 3-D DWT with emphasis on the impact of spatial scalability“, *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 740-753, 2006.

[23] B. Penna, T. Tillo, E. Magli, and G. Olmo, “Progressive 3-D coding of hyperspectral images based on JPEG 2000“, *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 125-129, 2006.

[24] ITU-T and ISO/IEC JTC 1, “JPEG 2000 Image Coding System: Core Coding System, ITU-T Recommendation T.800 and ISO/IEC 15444-1“ *JPEG 2000 Part 1*, 2000.

[25] D. Marpe, T. Wiegand, “A Highly Efficient Multiplication-Free Binary Arithmetic Coder and Its Application in Video Coding“, *IEEE International Conference on Image Processing*, 2003.

[26] E. Belyaev, K. Egiazarian and M. Gabbouj, “Low complexity bit-plane entropy coding for 3-D DWT based video compression“, *The International Symposium on SPIE Electronic Imaging*, 2012.

[27] G.M. Schuster, A.K. Katsaggelos, “Rate-Distortion Based Video Compression, Optimal Video Frame Compression, and Object Boundary Encoding“, *Kluwer Academic Publisher*, 1997.

[28] A. Munteanu, D. C. Cernea, A. Alecu, J. Cornelis, and P. Schelkens, “Scalable L-infinite coding of meshes“, *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, No. 3, pp. 513–528, 2010.

[29] E.Belyaev, “Low bit rate video coding based on three-dimensional discrete pseudo cosine transform“, *International Conference on Ultra Modern Telecommunications*, 2010.

[30] A.R. Reibman and B.G Haskell, “Constraints on variable bit-rate video for ATM networks“, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, Issue 4, pp. 361 – 372, 1992.

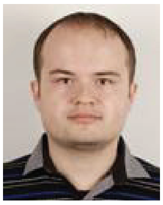
[31] x.264 video codec, http://x264.nl/

[32] MVC test sequences, http://www.merl.com/pub/avetro/mvc-testseq/

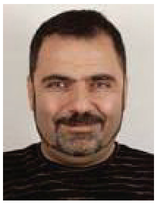
[33] Xiph.org test media, http://media.xiph.org/video/derf/

[34] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the Scalable Video Coding Extension of the H.264 / AVC Standard“, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, Issue 9, pp. 1103 – 1120, 2007.

[35] D. Turaga, M. van der Schaar, and B. Pesquet-Popescu, “Complexity scalable motion compensated wavelet video encoding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No. 8, pp. 982–993, 2005.



Evgeny Belyaev (M'12) received the Engineer degree and the Ph.D. (candidate of science) degree in technical sciences from State University of Aerospace Instrumentation (SUAI), Saint-Petersburg, Russia, in 2005 and 2009, respectively. He is currently a Researcher with the Institute of Signal Processing, Tampere University of Technology, Finland. His research interests include real-time video compression and transmission, video source rate control, scalable video coding, motion estimation and arithmetic encoding.



Karen Egiazarian (SM'96) received the M.Sc. degree in mathematics from Yerevan State University in 1981, the Ph.D. degree in physics and mathematics from Moscow State University, Moscow, Russia, in 1986, and the D.Tech. degree from the Tampere University of Technology (TUT), Tampere, Finland, in 1994. He has been Senior Researcher with the Department of Digital Signal Processing, Institute of Information Problems and Automation, National Academy of Sciences of Armenia. Since 1996, he has been an Assistant Professor with the Institute of Signal Processing, TUT, and from 1999 a Professor, leading the Computational Imaging Group. His research interests are in the areas of applied mathematics, image and video processing.

Moncef Gabbouj (F'11) received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, in 1985, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, in 1986 and 1989,respectively.

He is currently an Academy of Finland Professor with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He was the Head of the department from 2002 to 2007. He was on sabbatical leave at the American University of Sharjah, Sharjah, United Arab Emirates, from 2007 to 2008, and a Senior Research Fellow with the Academy of Finland, Helsinki, Finland, from 1997 to 1998 and in 2008. He is the Co-Founder and Past CEO with SuviSoft Oy, Ltd., Tampere. From 1995 to 1998, he was a Professor with the Department of Information Technology, Pori School of Technology and Economics, Pori, Finland, and from 1997 to 1998 he was a Senior Research Scientist with the Academy of Finland. From 1994 to 1995, he was an Associate Professor with the Signal Processing Laboratory, Tampere University of Technology. From 1990 to 1993, he was a Senior Research Scientist with the Research Institute for Information Technology, Tampere. His current research interests include multimedia content-based analysis, indexing and retrieval, nonlinear signal and image processing and analysis, and video processing and coding.

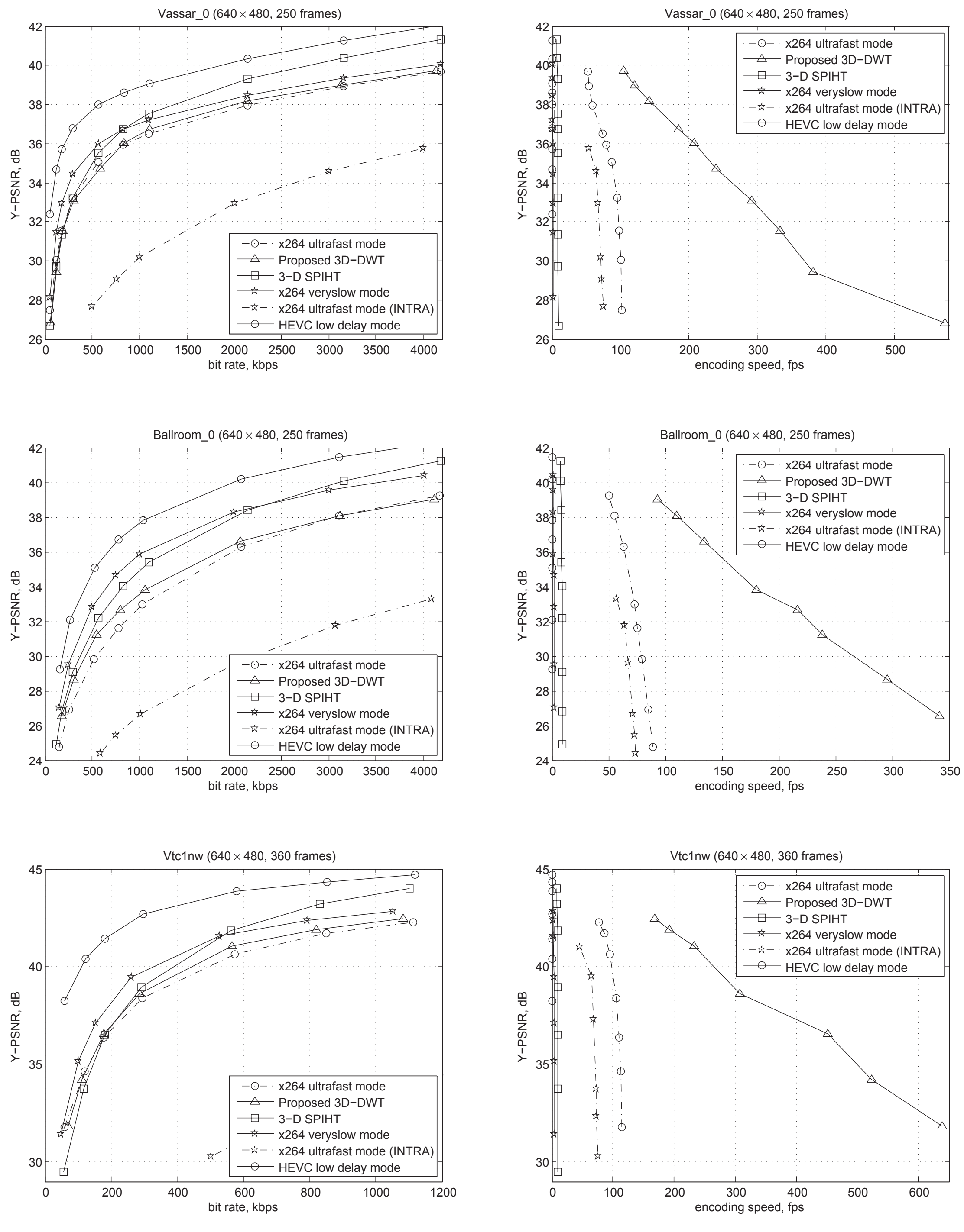


Fig. 9: Rate-distortion-complexity comparison of fast implementation of H.264/AVC standard and proposed 3-D DWT codec

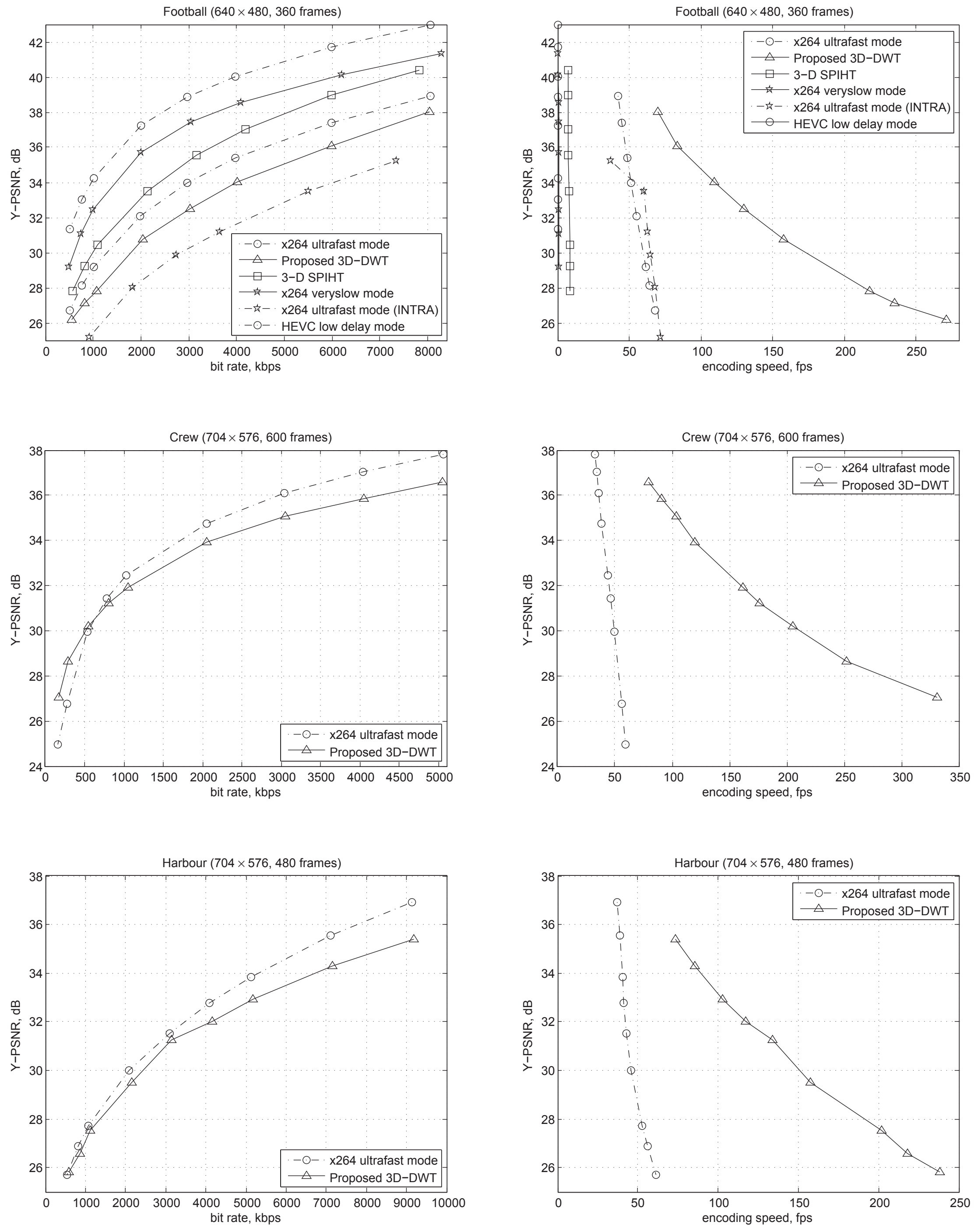


Fig. 10: Rate-distortion-complexity comparison of fast implementation of H.264/AVC standard and proposed 3-D DWT codec

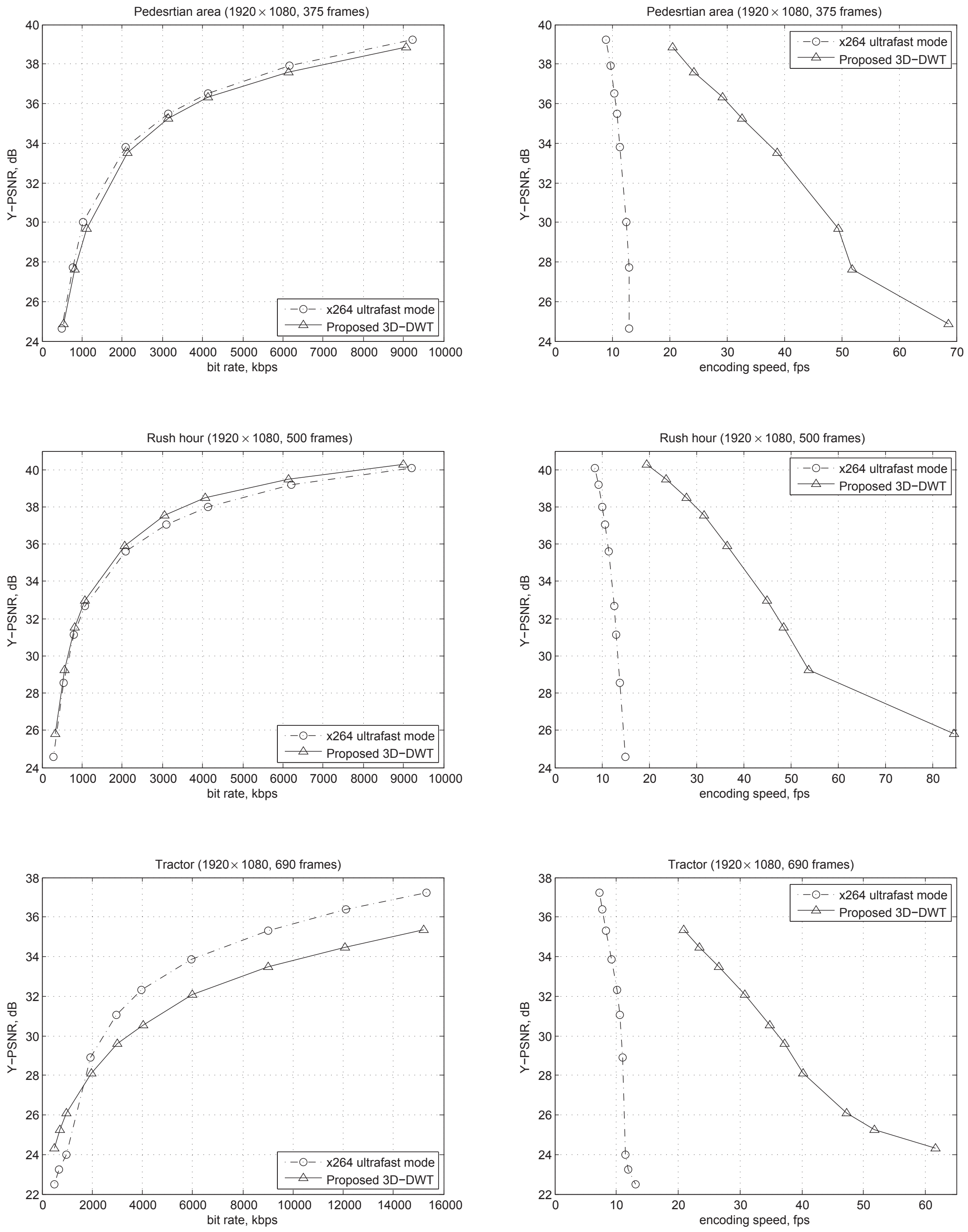


Fig. 11: Rate-distortion-complexity comparison of fast implementation of H.264/AVC standard and proposed 3-D DWT codec

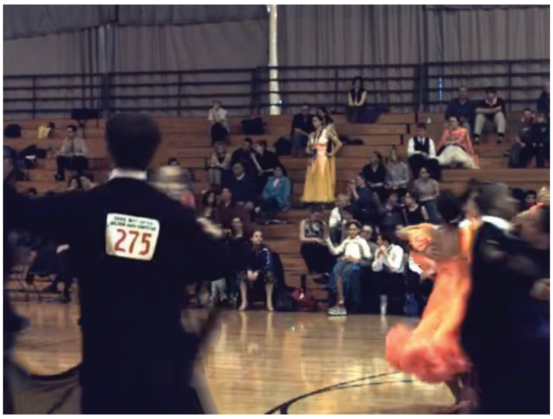


Fig. 12: HEVC, PSNR=35.1 dB, Speed=0.02 fps

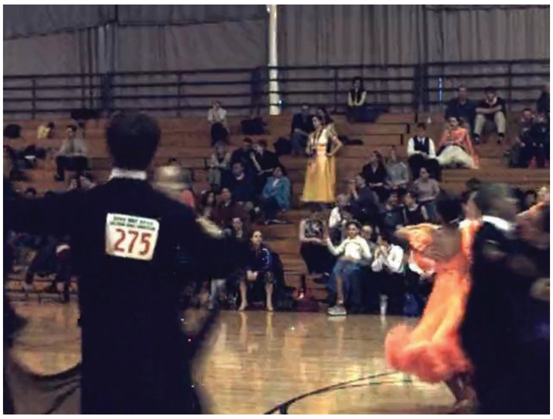


Fig. 15: x.264 veryslow, PSNR=32.8 dB, Speed=1.8 fps

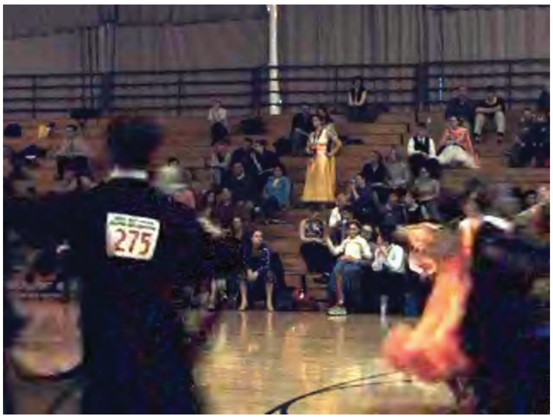


Fig. 13: 3-D SPIHT, PSNR=32.2 dB, Speed=9.2 fps

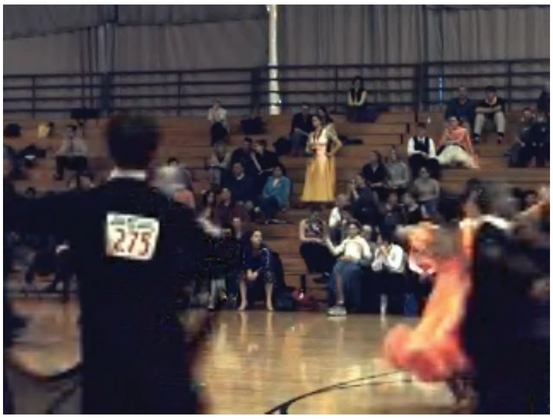


Fig. 16: Proposed 3-D DWT, PSNR=31.2 dB, Speed=238 fps

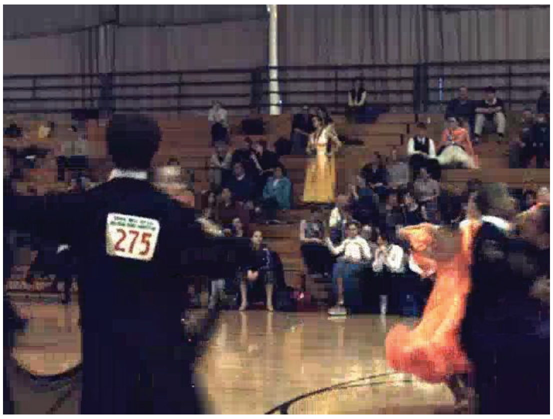


Fig. 14: x.264 ultrafast, PSNR=29.8 dB, Speed=79.6 fps

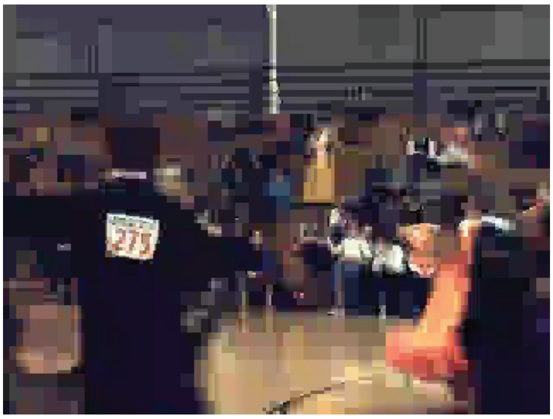


Fig. 17: x.264 INTRA, PSNR=24.4 dB, Speed=74.1 fps

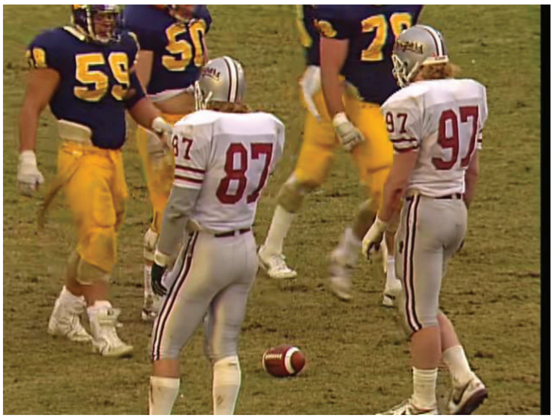


Fig. 18: HEVC, PSNR=34.2 dB, Speed=0.02 fps

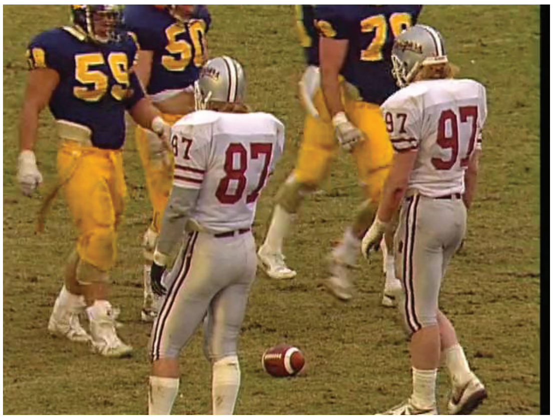


Fig. 21: x.264 veryslow, PSNR=32.5 dB, Speed=1.1 fps

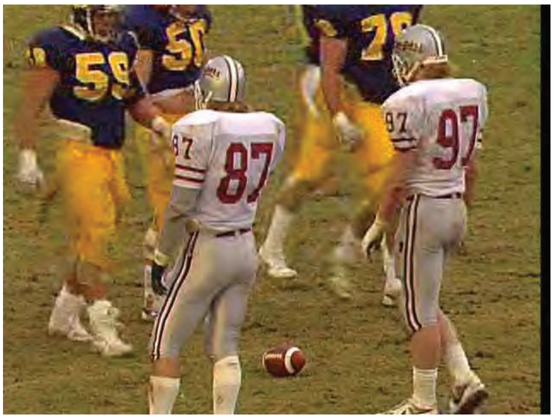


Fig. 19: 3-D SPIHT, PSNR=30.4 dB, Speed = 8.6 fps

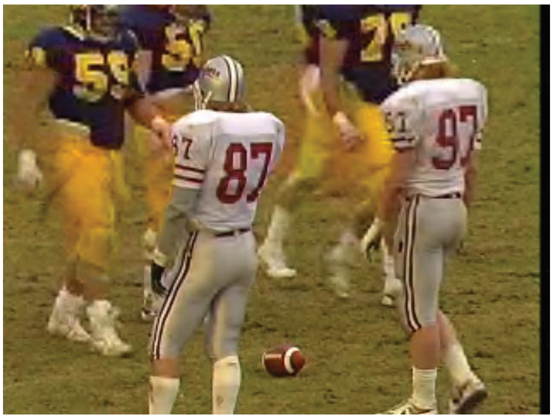


Fig. 22: Proposed 3-D DWT, PSNR=27.8 dB, Speed=217 fps

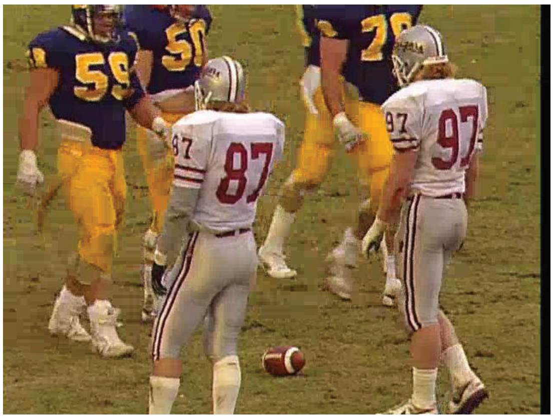


Fig. 20: x.264 ultrafast, PSNR=29.2 dB, Speed=61.4 fps

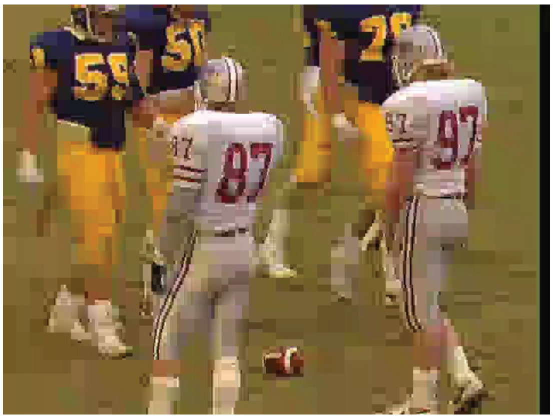


Fig. 23: x.264 INTRA, PSNR=25.2 dB, Speed=72.1 fps

[P3] E.Belyaev, K.Egiazarian, M.Gabbouj and K.Liu, "A Low-complexity joint source-channel video coding for 3-D DWT codec," *Journal of Communications*, vol.8, iss.12, pp.893–901, 2013.

A Low-Complexity Joint Source-Channel Video Coding for 3-D DWT Codec

Evgeny Belyaev[1], Karen Egiazarian[1], Moncef Gabbouj[1], and Kai Liu[2]
[1] Tampere University of Technology, Tampere 33720, Finland
[2] Xidian University, 710071 Xian, China
Email: {evgeny.belyaev,karen.egiazarian,moncef.gabbouj}@tut.fi; kailiu@mail.xidian.edu.cn

Abstract—In this paper we propose a low-complexity error-resilient and joint source-channel video coding algorithms for video codec based on three-dimensional discrete wavelet transform (3-D DWT). First, we show that the video stream, generated by 3-D DWT codec with the proposed error-resilient, is significantly more robust to packet losses than video stream generated by scalable extension of the H.264/AVC standard (H.264/SVC). Second, in comparison with H.264/SVC the proposed joint source-channel video coding algorithm demonstrates better performance for high packet loss rates. Finally, the computational complexity of 3-D DWT encoder with error-resilient and joint source-channel video coding is 3-6 times less than H.264/SVC encoder. Therefore, it can be more preferable for video transmission over highly unreliable channels and (or) in systems in which the video encoding computational complexity or power consumption play a critical role.

Index Terms—unequal loss protection, 3-D DWT, H.264/SVC

I. Introduction

Video coding and transmission are core technologies used in numerous applications such as streaming, surveillance, broadcasting and so on. Taking into account packet losses and time-varying bandwidth, the scalable video coding (SVC) is the most preferable compression method for video transmission over packet networks. During the last decade, in a number of papers it was demonstrated that in a combination with unequal loss protection (ULP) of different video stream layers, SVC provides robust transmission. Taking into account that bit errors are usually corrected at the lower network levels, only packet losses (or erasures) can happen at the application layer. Therefore, ULP can be easily implemented at the application layer using inter-packet Reed-Solomon (RS) codes without any modification of other network layers. In this case, the base video stream layer is protected using RS codes with a high redundancy level while the remaining layers are protected with a lower redundancy level or not protected at all.

Manuscript received September 9, 2013; revised November 21, 2013.
This work was supported by Tampere Doctoral Program in Information Science and Engineering (TISE), Finland and the project of National Natural Science Foundation of China (NSFC) for International Young Scientists.
Corresponding author email: evgeny.belyaev@tut.fi.
doi:10.12720/jcm.8.12.893-901

The most widespread ULP techniques, analyzed in the literature, are based on a scalable extension of the H.264/AVC standard (see, for example, [1], [2]) which includes temporal, spatial and quality scalability. H.264/SVC provides high compression efficiency due to motion compensation and inter-layer prediction. However, combined with the end-to-end distortion estimation caused by packet losses, erasure-correction coding and ULP optimization, the full video transmission system based on H.264/SVC has a high computational complexity. Finally, without ULP the video bit stream generated by H.264/SVC is very sensitive to packet losses and even 1% packet loss rate is enough for significant visual quality degradation when reconstructed video becomes not authentic (new objects, which are not present in original video, can appear).

As an alternative to H.264/SVC, scalable video coding based on three-dimensional discrete wavelet transform (3-D DWT) can be used [3]–[8]. Due to intensive investigation of Motion Compensated Temporal Filtering, currently 3-D DWT schemes have a comparable rate-distortion performance with the H.264/SVC. However, as in the previous case, combined with ULP these schemes also require high computational resources. For example, in [9] the encoding speed of 6 frames per second (fps) at CIF/SIF resolution was achieved on a Pentium M 2-GHz processor, which is not enough for real-time transmission (up to the authors knowledge there is no later work in the literature which shows complexity performance of joint compression and loss protection). But it is well known that there are a lot of applications such as video transmission for space missions [10], vehicle-to-infrastructure video delivery [11], underwater wireless video transmission [12], video transmission for wireless sensor networks [13] and so on, where the channel can be very unreliable, all video transmission parts such as video capturing, video encoding, decoding, packet loss protection and playback should operate in real-time, and transmitter computational resources (or power consumption) is very restricted. Therefore, a development of efficient low-complexity scalable video coders with ULP and rate control, which is a particular case of joint source-channel coding problem, is an important practical problem.

In this paper we extend our previous results [14] which were related to low-complexity video compression (without considering any transmission problems, like packet loss protection, packetization and so on) and introduce a novel low-complexity error-resilient and joint source-channel video coding algorithms for video codec based on three-dimensional discrete wavelet transform. First, we show that the video stream, generated by 3-D DWT codec with the proposed error-resilient, is significantly more robust to packet losses than video stream generated by scalable extension of the H.264/AVC standard (H.264/SVC). Second, in comparison with H.264/SVC the proposed joint source-channel video coding algorithm demonstrates better performance for high packet loss rates. Finally, the computational complexity of 3-D DWT encoder with error-resilient and joint source-channel video coding is 3–6 times less than H.264/SVC encoder.

The rest of the paper is organized as follows. Section II introduces the low-complexity joint source-channel video coding algorithm based on 3-D DWT: Section II-A shows the general scheme of the 3-D DWT codec, Section II-B describes the packetization and packet loss protection based on inter-packet Reed-Solomon codes, Section II-C formulates the optimization task for joint source-channel video coding for considered case and describes how this task can be solved by high complex Lagrange relaxation method, Sections II-D and II-E propose how the complexity can be significantly reduced. Section III presents error concealment and error-resilient coding for the 3-D DWT coder as well as comparisons of the proposed codec with scalable extension of H.264/AVC standard. Finally, conclusions are drawn in Section IV.

II. Proposed Algorithm

A. *General Video Coding Scheme Description*

The considered video coding scheme is shown in Fig. 1. First, a group of frames (GOF) of length N are accumulated in the input frame buffer. Then, a one-dimensional multilevel DWT of length N in the temporal direction is applied. All frames in GOF are processed starting from low-frequency to high-frequency frames. For each frame, the spatial subbands are also processed from low-frequency to high-frequency spatial subbands. Depending on the channel rate and motion level in a current GOF, the rate controller chooses one of the following options for each spatial subband:

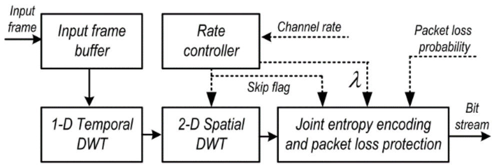


Fig. 1. The considered video coding scheme.

TABLE I. Packet Loss Protection Modes

Protection mode	1	2	3	…	11	12
Protection level	Four-fold repetition	RS (6,2)	RS (7,3)	…	RS (15,11)	No protection

- It permits calculation of 2-D spatial transform, selects the Lagrange multiplier λ and then permits minimization of a Lagrangian sum by selection of the ULP mode and the lowest bit-plane number (truncation point) which is Included into a bit stream;
- It prohibits the calculation of the 2-D spatial DWT, the entropy coding and ULP mode selection for the subband. At the decoder side, the corresponding transform coefficients in the subband are considered to be zero.

B. *Packetization and Packet Loss Protection*

The proposed entropy encoding and ULP scheme is shown in Fig. 2. Each wavelet subband is independently compressed using bit-plane entropy coding with a simplified JPEG2000 context modeling. As it is proposed in [14], the entropy encoder consists of two cores: Levenstein zero-run coder for low-entropy binary contexts and adaptive binary range coder for the remaining binary contexts. Each core has its own output buffer. Comparing to the traditional bit-plane encoders it allows to decrease the complexity of the wavelet encoder up to 2 times [14], without any significant rate-distortion performance degradation.

Compression is starting from the highest bit-plane t_{max} and, when a subband is truncated at the bit-plane t^*, each buffer contains $t_{max} - t^* + 1$ subpackets (each bit-plane fits into two subpackets). ULP is achieved due to the selection of different protection levels for each subband. The list of used packet loss protection levels is illustrated in Table I.

Levels 2-11 are based on inter-packet $RS(n, k)$code in finite field $GF(2^8)$ with the generator polynomial

$$g(x) = x^4 + 30x^3 + 216x^2 + 231x + 166.$$

In this approach k source bytes with the same index are used to form of the source polynomial $m(x)$ and corresponding r parity bytes are generated as:

$$r(x) = -x^r \cdot m(x) \; mod \; g(x).$$

If the source byte with the current index does not exist (stuffing byte), we use zero byte instead of it. Finally, Real-time Transport Protocol (RTP) headers are added to each packet, and transmission is performed.

If k or more packets from n are delivered, then the RS decoder is able to recover all lost source packets. Let p denote the packet loss probability, then the probability that a source packet will be lost and not recovered by the RS decoder is

$$p_{dec}(p,n,k) = \sum_{i=n-k+1}^{n} \frac{i}{n} \cdot \binom{n}{i} \cdot p^i \cdot (1-p)^{n-i}. \tag{1}$$

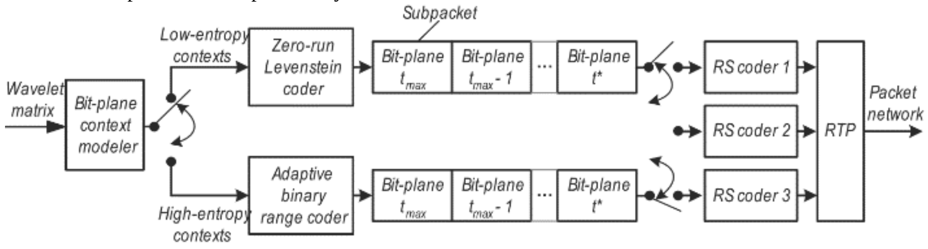


Fig. 2. Proposed joint entropy encoding and unequal packet loss protection scheme.

If the RS code (n_i, k_i) for the protection of two subpackets, containing a bit stream of bit plane i, is used, then the probability that these two subpackets will not be received and not recovered is

$$\pi_i = 1 - \left(1 - p_{dec}(p, n_i, k_i)\right)^2, \tag{2}$$

when these subpackets are placed into the different source packets and

$$\pi_i = p_{dec}(p, n_i, k_i), \tag{3}$$

when the subpackets are placed into the same source packet. It is easy to see, that in the second case, the probability that the subpackets are lost is always less than or equal to that of the first case. Therefore, we are using the second type of packetization.

Let $\psi = \{t^*, n^*, k^*\}$ denote a decision vector for a subband, which is truncated at a bit-plane $t^*$and for each bit-plane RS code (n^*, k^*) is used. Denote by d_i the subband distortion when the bit-planes $t_{max}, .., i$ are received and the bit-plane $i-1$ is not received. Then the expected distortion after truncation at bit-plane t^* is

$$E[d(\psi)] = \sum_{i=t^*+1}^{t_{max}} \prod_{j=i}^{t_{max}} (1-\pi_i) \cdot \pi_{i-1} \cdot d_i + \\ +\pi_{t_{max}} \cdot d_{skip} + \prod_{i=t^*}^{t_{max}} (1-\pi_i) d_{t^*}, \tag{4}$$

where d_{skip} is the distortion, when the subband is not received.

C. Joint Source-Channel Coding by Lagrangian-Relaxation

Let $\mathbf{\Psi} = \{\psi_i\}$ be the set of decision vectors, where ψ_i is the decision vector for the i'th subband in a GOF. The overall expected GOF distortion is

$$E[D(\mathbf{\Psi})] = \sum_i E[d(\psi_i)], \tag{5}$$

and the resulting GOF bit stream size is

$$R(\mathbf{\Psi}) = \sum_i r(\psi_{\mathrm{i}}), \tag{6}$$

where $r(\psi_i)$ is the bit stream size of source and parity packets for the i'th subband when the decision vector ψ_i is applied.

Fig. 3 illustrates the expected visual quality for different loss protection redundancies $\frac{n-k}{n}$ when source and parity packets bit stream size $R(\mathbf{\Psi})$ is equivalently to the channel rate 3000 kbps and packet loss probability is 20%. One can see, that if the redundancy is zero (no any protection is used), then the expected visual quality has a minimum value. With increase of the redundancy the distortion caused by packet loss protection decreases and the expected visual quality increases until some maximum value. A further increase of the redundancy does not lead to increase of the quality, because in order to keep the source and parity packets bit rate equal to the channel rate, it is needed to compress the wavelet subbands with higher and higher compression ratio. In this case the distortion caused by compression increases and distortion caused by losses is not significant, because the video bit stream is overprotected. Therefore, the visual quality decreases.

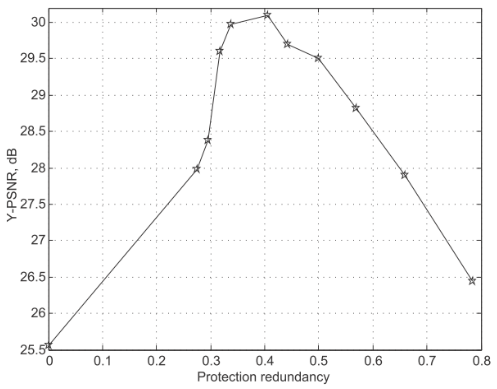


Fig. 3. Expected visual quality in case of protection by Reed-Solomon codes for packet loss probability 20% and channel rate 3000 kbps for "Football" (640x480).

In order to find the minimum overall expected distortion for a given video source, channel rate and packet loss probability the following joint source-channel video coding optimization task can be formulated. For each GOF we should determine the set of decision vectors $\mathbf{\Psi}^*$, such that

$$\begin{cases} \mathbf{\Psi}^* = \arg\min_{\{\mathbf{\Psi}\}} E[D(\mathbf{\Psi})] \\ R(\mathbf{\Psi}^*) \leq R_{max}, \end{cases} \tag{7}$$

where $R_{max} = \frac{N \cdot C}{f}$ is the bit budget for each GOF, C is the channel rate, and f is the video source frame rate. The optimization task can be solved using the Lagrangian relaxation method [15]. To use it the following statements should be proved.

Statement 1. *For each* $\lambda \geq 0$, *the set of decision vectors* $\mathbf{\Psi}_\lambda^* \in \{\mathbf{\Psi}\}$, *which minimizes*

$$E[D(\mathbf{\Psi})] + \lambda \cdot R(\mathbf{\Psi}), \tag{8}$$

is optimal solution of the optimization task, if $R_{max} = R(\mathbf{\Psi}_\lambda^*)$.

Proof. Let us assume that the statement is not hold, and there exist the set of decision vectors $\mathbf{\Psi} \in \{\mathbf{\Psi}\}$, so that $E[D(\mathbf{\Psi})] < E[D(\mathbf{\Psi}_\lambda^*)]$ and $R(\mathbf{\Psi}) \leq R(\mathbf{\Psi}_\lambda^*)$. Then inequality

$$E[D(\mathbf{\Psi})] + \lambda \cdot R(\mathbf{\Psi}) < E\left[D\left(\mathbf{\Psi}_\lambda^*\right)\right] + \lambda \cdot R\left(\mathbf{\Psi}_\lambda^*\right)$$

means that the set of decision vectors $\mathbf{\Psi}_\lambda^*$ is not minimizes (8) that contradicts with the statement formulation.

From Statement 1 it follows that in order to solve (7) one should find λ, such that $R(\Psi_\lambda^*) = R_{max}$.

Statement 2. *Let us assume that for some* λ_1 *and* λ_2, *values* $\mathbf{\Psi}_{\lambda_1}^*$ *and* $\mathbf{\Psi}_{\lambda_2}^*$, *which minimizes (8) are found.Then, if* $R\left(\mathbf{\Psi}_{\lambda_1}^*\right) > R(\mathbf{\Psi}_{\lambda_2}^*)$, *then the following inequality holds:*

$$\lambda_2 \geq -\frac{E\left[D\left(\mathbf{\Psi}_{\lambda_1}^*\right)\right] - E\left[D\left(\mathbf{\Psi}_{\lambda_2}^*\right)\right]}{R\left(\mathbf{\Psi}_{\lambda_1}^*\right) - R\left(\mathbf{\Psi}_{\lambda_2}^*\right)} \geq \lambda_1. \tag{9}$$

Proof. From Statement 1 it follows that

$$E\left[D\left(\mathbf{\Psi}_{\lambda_1}^*\right)\right] + \lambda_1 \cdot R\left(\mathbf{\Psi}_{\lambda_1}^*\right) < E\left[D\left(\mathbf{\Psi}_{\lambda_2}^*\right)\right] + \lambda_1 \cdot R\left(\mathbf{\Psi}_{\lambda_2}^*\right). \tag{10}$$

From (10) and $R\left(\mathbf{\Psi}_{\lambda_1}^*\right) > R\left(\mathbf{\Psi}_{\lambda_2}^*\right)$ follows, that

$$-\frac{E\left[D\left(\mathbf{\Psi}_{\lambda_1}^*\right)\right] - E\left[D\left(\mathbf{\Psi}_{\lambda_2}^*\right)\right]}{R\left(\mathbf{\Psi}_{\lambda_1}^*\right) - R\left(\mathbf{\Psi}_{\lambda_2}^*\right)} \geq \lambda_1, \tag{11}$$

what proves the right side of (9). Analogously, from Statement 1 it follows that

$$E\left[D\left(\mathbf{\Psi}_{\lambda_2}^*\right)\right] + \lambda_2 \cdot R\left(\mathbf{\Psi}_{\lambda_2}^*\right) < E\left[D\left(\mathbf{\Psi}_{\lambda_1}^*\right)\right] + \lambda_2 \cdot R\left(\mathbf{\Psi}_{\lambda_1}^*\right). \tag{12}$$

From (12) and $R\left(\mathbf{\Psi}_{\lambda_1}^*\right) > R\left(\mathbf{\Psi}_{\lambda_2}^*\right)$ follows, that

$$\lambda_2 \geq -\frac{E\left[D\left(\mathbf{\Psi}_{\lambda_1}^*\right)\right] - E\left[D\left(\mathbf{\Psi}_{\lambda_2}^*\right)\right]}{R\left(\mathbf{\Psi}_{\lambda_1}^*\right) - R\left(\mathbf{\Psi}_{\lambda_2}^*\right)}. \tag{13}$$

From Statement 2 it follows, that function $R(\mathbf{\Psi}_\lambda^*)$ is a non-increasing function of λ. It means that needed λ value can be found by the bisection method [15].

For the considered codec, the rate-distortion function for the subband i is independent of the truncation and RS code parameters selected for other subbands. Therefore,

$$\begin{aligned} &\min_{\mathbf{\Psi}}\{E[D(\mathbf{\Psi})] + \lambda \cdot R(\mathbf{\Psi})\} = \\ &= \min_{\mathbf{\Psi}}\left\{\sum_i E[d(\psi_i)] + \lambda \cdot r(\psi_i)\right\} = \\ &= \sum_i \min_{\psi_\mathrm{i}}(E[d(\psi_i)] + r(\psi_i)). \end{aligned} \tag{14}$$

Therefore, a solution of the task (7) can be written as $\mathbf{\Psi}^* = \{\psi_i^*\}$, where for each subband i the value $\psi_i^* = \{t_i^*, n_i^*, k_i^*\}$ can be found independently as

$$\psi_i^* = \arg\min_{\{\psi_\mathrm{i}\}}\{E[d(\psi_i)] + \lambda \cdot r(\psi_i)\}. \tag{15}$$

D. Skipping of Spatial Subbands for Complexity Reduction

Algorithm 1 Wavelet subband truncation and RS code parameters selection

```
Input: Subband$\{x[i,j]\}$, $\lambda$
 1: $t_{max} \leftarrow \left\lfloor \log_2(\max_{(i,j)} x[i,j]) \right\rfloor$
 2: $E \leftarrow \sum_{(i,j)} x^2[i,j]$
 3: $\Omega^* \leftarrow E$
 4: for $t = t_{max},..,0$ do
 5:     Encode bit-plane $t$ as in [14]
 6:     $r \leftarrow$ get_subband_bit_stream_size()
 7:     $(n_t, k_t) \leftarrow \arg\min_{(n,k)} E[d(t,n,k)] + \lambda \cdot r \cdot \frac{n}{k}$
 8:     $\Omega(t, n_t, k_t) = E[d(t, n_t, k_t)] + \lambda \cdot r \cdot \frac{n_t}{k_t}$
 9:     if $\Omega(t, n_t, k_t) \geq \Omega^*$ then
10:         if $\Omega^* = E$ then
11:             Skip subband
12:         else
13:             Truncate subband at bit-plane $t^*$
14:             Use Reed-Solomon $(k^*, n^*)$ code
15:         end if
16:         exit
17:     else
18:         $\Omega^* \leftarrow \Omega$
19:         $t^* = t, n^* = n_t, k^* = k_t$
20:     end if
21: end for
```

In this paper we propose to use the following Algorithm 1 for ψ_i^* selection. First, all bit planes are compressed from the highest to the lowest bit plane as it is proposed in our previous work [14]. Then, after encoding of each bit plane with number t we calculate the bit stream size r for bit planes $t_{max}, .., t$ of the subband. After that the Reed-Solomon (k^*, n^*) code from Table I which minimizes the Lagrange sum (15) is choosen taking into account that the expected distortion is calculated as in (4) and the full bit stream size includes the compressed subband bit stream size of r bits as well as the bit size of parity packets and calculated as$r \cdot \frac{n}{k}$. If the current Lagrange sum for bit-plane t is less than for bit plane $t-1$, then the bit plane $t+1$ is encoded. Otherwise, the encoding process is stopped, and the subband is truncated at the current bit plane and protected by Reed-Solomon (k^*, n^*) code. If, it is more efficient to not transmit even the highest bit plane t_{max}, then the subband is skipped and at the decoder side all wavelet coefficients of the subband are considered to be zero. One can see, that Algorithm 1 corresponds to (15) with the assumption that $E[d(\psi_i) + \lambda \cdot r(\psi_i)$ is a convex function.

From Algorithm 1, it follows that if

$$\Omega(t_{max}, n_{t_{max}}, k_{t_{max}}) > \sum_{(i,j)} x^2[i,j], \tag{16}$$

then this subband is skipped. In this situation the encoder wastes computational resources for calculating the 2-D DWT and for forming the subband bit stream which is not included in the output bit stream. Therefore, it is important to find a criterion which can guarantee with ahigh probability that the current subband will be skippedbefore processing of this subband.

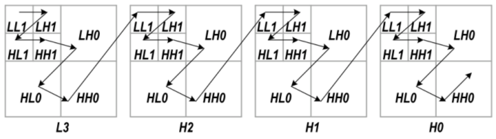


Fig. 4. Example of spatial subbands encoding order.

In our previous work [14] the authors have proposed the following parent-child tree based subband skip criterion, which can be used also for complexity reduction of joint compression and protection scheme considered in this work. It is well known that the value of the wavelet coefficient in the child subband is correlated with the corresponding coefficient in the parent subband. This property is widely used in image and video compression algorithms based on inter-subband correlation (see, e.g. [16]). Using the same reasoning, we first process subbands from low-frequency to high-frequency temporal subbands and from low-frequency to high-frequency spatial subbands (see Fig. 4). Second, we make the following *coding assumption*: if for any subband, inequality (16) holds, then for all temporal-spatial child-subbands, the same inequality also holds. In this case, all child-subbands will not be processed and hence spatial transform calculation, entropy coding and ULP parameters selection are skipped.

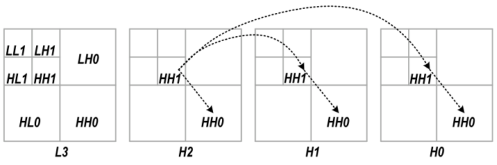


Fig. 5. Spatial subbands skipping.

Fig. 5 illustrates the proposed approach. Frames after temporal wavelet decomposition are denoted as L3, H2, H1, H0, where L3 is the low-frequency frame. If the spatial subband HH1 in frame H2 is skipped then the corresponding child-subbands HH0 in frame H2, HH1 and HH0 in frame H1, and HH1 and HH0 in frame H0 are skipped without any processing.

E. Proposed Joint Source-Channel Coding Algorithm

To find the necessary λ value with the bisection method, one needs to calculate the 2-D DWT and losslessly compress each subband in order to determine all sets of truncation points, which requires significant computational resources. In this paper we propose a one-pass joint source-channel video coding algorithm which uses the *virtual buffer* concept to estimate λ without lossless coding. Let us define b_{virt} to be the number of bits in avirtual buffer and B_{max} the virtual buffer size determined as $B_{max} = L \cdot C$, where L is the required bufferinglatency at the decoder side for continuous video playback, C is the channel rate. The proposed algorithm is presented below.

At step 4, Algorithm 2 selects the Lagrange multiplier λ value in the proportional to the virtual buffer fullness, where γ defines the proportion degree. It allows increasing or decreasing the compression ratio for the current GOF depending on the buffer fullness. At step 5, all frames in the GOF are processed starting from low-frequency to high-frequency frames. For each frame, the spatial subbands are also processed from low-frequency to high-frequency spatial subbands. At step 8, each subband is compressed starting from the highest bit-plane, until the Lagrangian sum decreases. At step 9, if the highest bit-plane is not included into the output bit stream, this subband is skipped together with all of its child-subbands. At steps 10 and 12 the virtual buffer fullness is updated.

Algorithm 2 Proposed joint source-channel coding algorithm

```
1: $b_{virt} \leftarrow b_{virt}(0)$
2: **for** $j = 0,..,j_{max} - 1$ **do**
3:     Calculate 1-D temporal DWT for GOF $j$.
4:     $\lambda \leftarrow \lambda_{max} \left(\frac{b_{virt}}{B_{max}}\right)^{\gamma}$
5:     **for** $i = 0,..,i_{max} - 1$ **do**
6:         **if** parent subband was skipped **then** skip subband $i$;
7:         Calculate 2-D spatial DWT for subband $i$;
8:         Find $\psi_i^* = \arg\min_{\{\psi_i\}}\{E[d(\psi_i)] + \lambda \cdot r(\psi_i)\}$
9:         **if** $r(\psi_i^*) = 0$ **then** skip subband $i$;
10:        $b_{virt} \leftarrow b_{virt} + r(\psi_i^*)$
11:    **end for**
12:    $b_{virt} \leftarrow b_{virt} - R_{max}$
13: **end for**
```

Note that if during the encoding process of a video sequence the number of bits in the virtual buffer requires

$$0 \le b_{virt} \le B_{max}, \tag{17}$$

then the average bit stream size for each GOF $\bar{R}$ is bounded by

$$R_{max} \le \bar{R} \le R_{max} + \frac{B_{max}}{j_{max}}, \tag{18}$$

where j_{max} is the number of GOF's in a video sequence. From (18) it follows that if (17) holds during the encoding process, then the average bit stream size for each GOF $\bar{R}$ tends to R_{max} with an increasing j_{max}.

To further explain Algorithm 2, let us suppose that the video encoder compresses the series of identical GOF's. Let us define λ^* as the Lagrange multiplier value computed using the bisection method described in Section II-C and consider the virtual buffer fullness when the initial buffer fullness is

$$b_{virt}(0) = b^*_{virt}(0) = B_{max}\left(\frac{\lambda^*}{\lambda_{max}}\right)^{\frac{1}{\gamma}}, \qquad (19)$$

Taking into account that for the bisection method $R(\Psi^*_\lambda) = R_{max}$ (see Statement 1), after processing each GOF, the buffer fullness will be constant and equal to

$$b_{virt} = b^*_{virt}(0).$$

Now let us analyze the buffer fullness in the case when the initial buffer fullness $b_{virt}(0) \neq b^*_{virt}(0)$. Let us assume that after compression of the previous GOF the current buffer fullness is $b_{virt} < b^*_{virt}(0)$. It means that λ value calculated by line 4 of Algorithm 2 for the current GOF will be less than the optimal λ^*. Taking into account that $R(\boldsymbol{\Psi}^*_\lambda)$ is a non-increasing function of λ (see Statement 2), after compression of this GOF the difference $R(\boldsymbol{\Psi}^*_\lambda) - R_{max} > 0$. Therefore, a new buffer fullness calculated by line 10 will be higher than previous one. It means that after compression of the next GOF, the current value of λ will be closer to the optimal λ^* and so on, until the current value becomes equal or exceeds λ^*.

Analogically, if the current $b_{virt} > b^*_{virt}(0)$ then after compression of the next GOF the current value of λ will also be closer to the optimal λ^* and so on, until the current λ value becomes equal or smaller than λ^*.

Taking into account the reasoning described above, the value of λ calculated by the proposed algorithm will oscillate around the optimal value λ^* founded by bisection method.

III. Simulation Results

The proposed codec run with a GOF size $N = 16$ using the Haar transform in the temporal direction and the 5/3 spatial wavelet transform at three-levels of the decomposition. At the decoder side we use the following error concealment. First, bit stream of each subband is decoded in a progressive way until a loss is detected. In this case, any losses in stream corresponds to the higher quantization at the encoder side and does not lead to error propagation. Second, if main low-frequency subband, which contains brightness information for all GOF, is lost we copy corresponding subband from the previous GOF. In this case an error propagation can happens. To minimize probability of this event we use the following simple error-resilient coding. At the encoder side for the main subband we use repetition of the highest bit-planes which can be placed into the two additional source packets.

In case of H.264/SVC we have used JSVM 9.8 reference software [17] with two spatial and five temporal scalable layers. GOP size and intra-frame period are set to 16. For error-resilient coding we have used flexible macroblock ordering with two slice groups and loss-aware rate-distortion optimized macroblock mode decision. Frame copy error concealment method is used at the decoder side. For ULP we use Reed-Solomon codes from Table I, so different level of protection can be achieved for each spatial and temporal layer.

For 3-D DWT codec, joint source-channel video coding algorithm was implemented in real-time mode as it was described above, while ULP optimization for H.264/SVC was realized in off-line mode.

Simulation results were obtained for the video sequence "Football" (640 × 480, 360 frames, 30 Hz) [18]. For both codecs the packet length was set to 800 bytes and channel with independent packet losses was simulated. For visual quality measurement we use the following approach. During each measurement with an index i the sum of square errors D_i between luma component of original and reconstructed video is calculated. Then we repeat this experiment $K = 20$ times (7200 frames) and estimate the Expected Peak Signal-to-Noise Ratio ($E[Y-PSNR]$) which we use as the visual quality metric:

$$E[Y-PSNR] = 10\log_{10}\frac{255^2}{\frac{1}{K}\sum_{i=0}^{K-1} D_i}. \qquad (20)$$

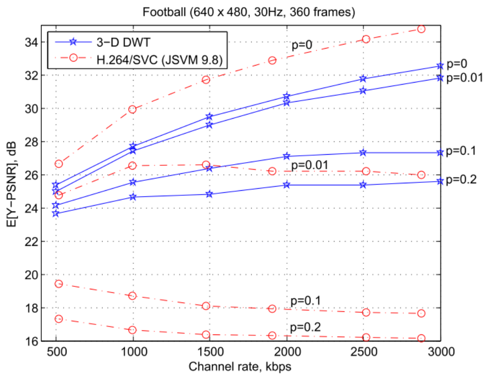


Fig. 6. Expected visual quality comparison for video sequence "Football" with no loss protection.

Fig. 6 and Fig. 7 show the expected PSNR for different packet loss probabilities and channel rates for the proposed codec and for H.264/SVC with and without loss protection (see detailed comparison at http://www.cs.tut.fi/˜belyaev/ULP.htm). One can see that the proposed codec is significantly less sensitive to packet losses. Even if packet loss ratio is 5% it provides much better visual quality (see Fig. 8) while H.264/SVC has frames with unrecognizable objects. In case of ULP, H.264/SVC provides better performance (up to 1.5dB) for low packet loss probabilities, but with increasing of it the proposed codec provides the same or better performance (up to 1.5dB). It can be explained by the following reason.

Since H.264/SVC is more sensitive to packet losses it is needed to add more redundancy by Reed-Solomon coding than 3-D DWT codec, which has less compression performance, but requires less redundancy for protection (see Table II).

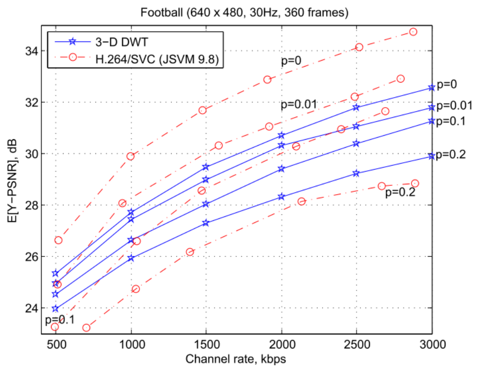


Fig. 7. Expected visual quality comparison for video sequence "Football" with unequal packet loss protection.

For computation complexity measurements we use the encoding speed which is defined as the number of frames which can be encoded in one second by given CPU. The complexity is considered as the inverse value of the encoding speed, which was measured without any use of assemblers, threads, or other program optimization techniques. Table III shows encoding speed for the proposed codec and H.264/SVC depending on the video bit rate. Taking into account that JSVM 9.8 reference software is not optimized for real-time operation we have also used fast implementation of the H.264/AVC standard in single-layer mode (x.264 codec in ultrafast profile [19]) to estimate possible encoding speed of H.264/SVC after optimization. Our estimation shows that the complexity of the proposed codec is 3–6 times less than H.264/SVC at least.

TABLE II: PROTECTION REDUNDANCY

Packet loss probability, p	0	0.01	0.1	0.2
H.264/SVC (JSVM 9.8)	0	0.276	0.464	0.617
3-D DWT	0	0	0.271	0.412

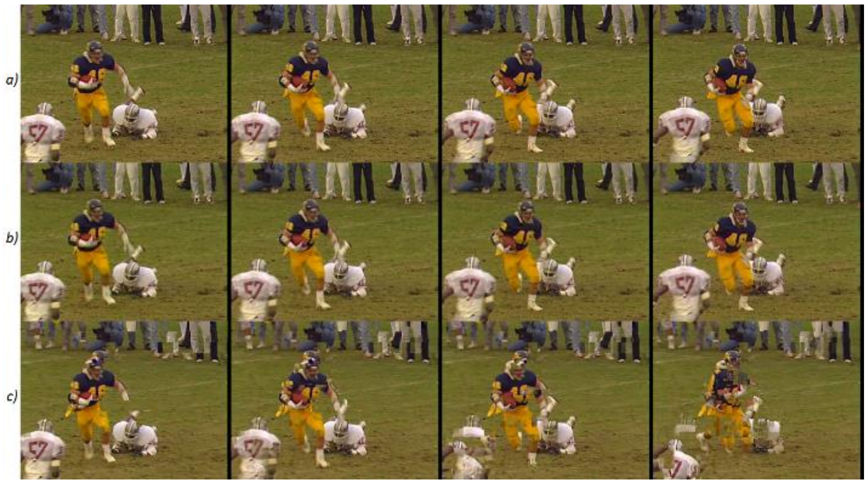


Fig. 8. Visual comparison for 5% packet loss. a) Original frames b) Proposed 3-D DWT c) H.264/SVC (JSVM 9.8)

TABLE III: ENCODING SPEED FOR CPU INTEL CORE 2.1 GHZ, FPS

Video bit rate, kbps	500	1000	2000	2500	3000
H.264/SVC (JSVM 9.8)	<1	<1	<1	<1	<1
x.264	49	44	40	37	36
3-D DWT	279	215	150	133	120
3-D DWT + ULP (p = 0.1)	288	185	160	137	130
3-D DWT + ULP (p = 0.2)	315	198	177	160	145

Additionally, we have added the full complexity for the proposed codec with ULP for different packet loss probabilities. One can see that the complexity of the proposed codec decreases with increasing of the packet loss probability or with decreasing of the bit rate. This can be explained as follows. An increase of a packet loss probability leads to an increase of the portion of parity packets in the output bit stream. In order to keep the source and parity data bit rate equal to the channel rate, the video compression ratio is increased, and condition $r(\psi_i^*) = 0$ in Algorithm 2 holds for more and more subbands which are skipped together with the corresponding child-subbands. As a result, 2-D spatial DWT, entropy coding and ULP mode selection are not performed for a significant portion of spatial subbands.

IV. CONCLUSIONS

A novel low-complexity error-resilient and joint source-channel video coding algorithms for video codec based on three-dimensional discrete wavelet transform were presented. We have shown that in comparison with

H.264/SVC, the proposed codec has the following advantages:

- It is much less sensitive to packet losses and provides robust video transmission and authentic reconstruction for 1–5% packet loss rates without any packet loss protection. In case of H.264/SVC even 1% packet loss rate is enough for significant visual quality degradation. Reconstructed video of H.264/SVC can be not authentic, i.e. new objects, which are not present in original video, can appear.
- The proposed joint source-channel video coding algorithm demonstrates better performance for high packet loss rates.
- The computational complexity of 3-D DWT encoder with error-resilient and joint source-channel video coding is 3–6 times less than H.264/SVC encoder. High complexity of H.264/SVC is caused by high complexity of motion estimation, backward loop at the encoder side, input frame down sampling, and end-to-end distortion estimation caused by packet losses. It is difficult to use of the H.264/SVC on mobile devices, personal computers and other systems, which compress one or more high resolution video sources in real-time. Therefore, H.264/SVC is a solution for systems based on pre-encoded video.

Therefore, the 3-D DWT codec with proposed error-resilient and joint source-channel video coding algorithms can be more preferable for video transmission over highly unreliable channels and (or) in systems in which the video encoding computational complexity or power consumption play a critical role.

REFERENCES

[1] E. Maani and A. Katsaggelos, “Unequal error protection for robust streaming of scalable video over packet lossy networks,” *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 20, no. 3, pp.407–416, 2010.

[2] Y. Huo, M. El-Hajjar, and L. Hanzo, “Inter-layer fec aided unequal error protection for multilayer video transmission in mobile TV,” *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 20, no. 3, pp. 1622–1634, 2013.

[3] M. van der Schaar and D. Turaga, “Cross-layer packetization and retransmission strategies for delay-sensitive wireless multimedia transmission,” *IEEE Transaction on Multimedia*, vol. 9, no. 1, pp. 185–197, 2007.

[4] D. Taubman and A. Zakhor, “Multi-rate 3-D subband coding of video,” *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, 1994.

[5] J. Tham, S. Ranganath, and A. Kassim, “Highly scalable wavelet-based video codec for very low bit-rate environment,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 12–27, 1998.

[6] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. Van der Schaar, J. Cornelis, and P. Schelkens, “In-band motion compensated temporal filtering,” *Signal Processing: Image Communication*, vol. 19, no. 7, pp. 653–673, 2004.

[7] D. Zhang, W. Zhang, J. Xu, F. Wu, and H. Xiong, “A cross-resolution leaky prediction scheme for in-band wavelet video coding with spatial scalability,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 4, pp. 516–521, 2008.

[8] N. Mehrseresht and D. Taubman, “A flexible structure for fully scalable motion-compensated 3-D DWT with emphasis on the impact of spatial scalability,” *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 740–753, 2006.

[9] J. Fowler, M. Tagliasacchi, and B. Pesquet-Popescu, “Video coding with wavelet-domain conditional replenishmentand unequal error protection,” in *Proc. IEEE International Conference on Image Processing*, Atlanta, USA, 2006, pp. 1869-1872.

[10] R. Viola, L. Chiariglione, and R. Russo, “Video compression for manned space missions,” in *Proc. IEEE Global Telecommunications Conference*, Dallas, USA, 1989.

[11] A. Vinel, E. Belyaev, O. Lamotte, M. Gabbouj, Y. Koucheryavy, and K. Egiazarian, “Video transmission over IEEE 802.11p: Real-world measurements,” presented at the IEEE ICC-2013(Workshop on Emerging Vehicular Networks), Budapest, Hungary, June 9-13, 2013.

[12] J. Ribas, D. Sura, and M. Stojanovic, “Underwater wireless video transmission for supervisory control and inspection using acoustic OFDM,” in *Proc. OCEANS*, Seattle, USA, 2010, pp. 1-9.

[13] I. Politis, M. Tsagkaropoulos, and S. Kotsopoulos, “Optimizing video transmission over wireless multimedia sensor networks,” in *Proc. IEEE Global Telecommunications Conference*, New Orleans, USA, 2008, pp. 117-122.

[14] E. Belyaev, K. Egiazarian, and M. Gabbouj, “A low-complexity bit-plane entropy coding and rate control for 3-D DWT based video coding,” *IEEE Transactions on Multimedia*, 2013.

[15] G. Schuster and A. Katsaggelos, “Rate-distortion based video compression, optimal video frame compression, and object boundary encoding,” *Kluwer Academic Publisher*, Norwell, MA, USA, 1997.

[16] B. Kim, Z. Xiong, and W. Pearlman, “Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT),” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 8, pp. 1374–1378, 2000.

[17] JSVM 9.8 software package. [Online]. Available: http://iphome.hhi.de/

[18] Xiph.org test media. [Online]. Available: http://media.xiph.org/video/derf/

[19] x.264 Video Codec. [Online]. Available: http://x264.nl/

Evgeny Belyaev received the Engineer degree and the Ph.D. (candidate of science) degree in technical sciences from State University of Aerospace Instrumentation (SUAI), Saint-Petersburg, Russia, in 2005 and 2009, respectively. He is currently a Researcher with the Institute of Signal Processing, Tampere University of Technology, Finland.
His research interests include real-time video compression and transmission, video source rate control, scalable video coding, motion estimation and arithmetic encoding.

Karen Egiazarian received the M.Sc. degree in mathematics from Yerevan State University in 1981, the Ph.D. degree in physics and mathematics from Moscow State University, Moscow, Russia, in 1986, and the D.Tech. degree from the Tampere University of Technology (TUT), Tampere, Finland, in 1994.
He has been Senior Researcher with the Department of Digital Signal Processing, Institute of Information Problems and Automation, National Academy of Sciences of Armenia. Since 1996, he has been an Assistant Professor with the Institute of Signal Processing, TUT, and from 1999 a Professor, leading the Computational Imaging Group.
His research interests are in the areas of applied mathematics, image and video processing.

Moncef Gabbouj received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, in 1985, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, in 1986 and 1989, respectively.
He is currently an Academy of Finland Professor with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He was the Head of the department from 2002 to 2007. He was on sabbatical leave at the American University of Sharjah, Sharjah, United Arab Emirates, from 2007 to 2008, and a Senior Research Fellow with the Academy of Finland, Helsinki, Finland, from 1997 to 1998 and in 2008. He is the Co-Founder and Past CEO with SuviSoftOy, Ltd., Tampere. From 1995 to 1998, he was a Professor with the Department of Information Technology, Pori School of Technology and Economics, Pori, Finland, and from 1997 to 1998 he was a Senior Research Scientist with the Academy of Finland. From 1994 to 1995, he was an Associate Professor with the Signal Processing Laboratory, Tampere University of Technology. From 1990 to 1993, he was a Senior Research Scientist with the Research Institute for Information Technology, Tampere. His current research interests include multimedia content-based analysis, indexing and retrieval, nonlinear signal and image processing and analysis, and video processing and coding.

Kai Liu received the B.S. and M.S. degrees in computer science and the Ph.D. degree in signal processing from Xidian University, Xian, China, in 1999, 2002, and 2005, respectively. Currently, he is an Associate Professor of computer science and technology with the Xidian University.
His major research interests include VLSI architecture design and image coding.

[P4] E.Belyaev, A.Veselov, A.Turlikov and Kai Liu, "Complexity analysis of adaptive binary arithmetic coding software implementations," Proceedings of *The 11th International Conference on Next Generation Wired/Wireless Advanced Networking*, Saint-Petersburg, Russia, August 23–25, 2011.

Complexity analysis of adaptive binary arithmetic coding software implementations

Evgeny Belyaev[1], Anton Veselov[2], Andrey Turlikov[2] and Liu Kai[3]

[1]Tampere University of Technology, Korkeakoulunkatu 10, 33720 Tampere, Finland
{evgeny.belyaev@tut.fi}
[2]Saint-Petersburg State University of Aerospace Instrumentation, Bolshaya Morskaya 67, 190000 St. Petersburg, Russia
{felix,turlikov}@vu.spb.ru
[3]School of Computer Science and Technology, Xidian University, NO.2 Taibai South Road, MailBox 161, 710071 Xi'an, China
{kailiu@mail.xidian.edu.cn}

Abstract. This paper is dedicated to the complexity comparison of adaptive binary arithmetic coding integer software implementations. Firstly, for binary memoryless sources with known probability distribution, we prove that encoding time for arithmetic encoder is a linear function of a number of input binary symbols and source entropy. Secondly, we show that the byte-oriented renormalization allows to decrease encoding time up to 40% in comparison with bit-oriented renormalization. Finally, we study influence of probability estimation algorithm for encoding time and show that probability estimation algorithm using "Virtual Sliding Window" has less computation complexity than state machine based probability estimation algorithm from H.264/AVC standard.

Key words: binary arithmetic coder, range coder, probability estimation, complexity analysis

Introduction

Adaptive binary arithmetic coding is included in well known image and video compression standards and state of the art codecs like JPEG [1], JPEG2000 [2], H.264/AVC [3], Dirac [4] etc. Arithmetic coders implemented in these codecs are based on Q-coder [5] which is multiplication free adaptive binary arithmetic coder with bit renormalization and look-up tables used for probability estimation. Q-coder was introduced in 1988 and since that time the relative computational complexity of different arithmetical operations changed significantly. For example, table look-up operation takes more CPU clock cycles than a multiplication [7]. Thus, these changes should be considered for designing of a new video compression standards (especially for High Efficiency Video Coding (HEVC) [8]) or state of the art codecs.

In this paper we compare adaptive binary range coder introduced in [6] with arithmetic coder from H.264/AVC standard. At first, we show that the byte-oriented renormalization allows to decrease encoding time up to 40% in comparison with bit-oriented renormalization. Then we investigate the influence of probability estimation algorithm for encoding time and show that using look-up table free "Virtual Sliding Window" (VSW) algorithm [14] allows to decrease the encoding time up to 10% in comparison with probability estimation algorithm from H.264/AVC standard.

Other actual topic for arithmetic encoding is complexity or power consumption modeling which is needed for power-rate-distortion analysis. Previous works [9, 10] use assumption that entropy encoder complexity is approximately proportional to the output bit rate. In this paper we prove that in case of binary memoryless sources with known probability distribution encoding time for binary arithmetic encoder is a linear function of a number of input binary symbols and source entropy. If probability is not known, then encoding time is a linear function of a number of input binary symbols and output bit rate.

The rest of this paper is organized as follows. Section 1 describes the main idea of arithmetic encoding and its two integer implementations with different renormalizations. Section 2 is dedicated to integer implementations of probability estimation based on sliding window approximations. Section 3 introduces the linear model for complexity of binary arithmetic encoder and show the comparative results for described adaptive binary arithmetic coding software implementations.

1 Integer implementations of binary arithmetic encoder

Let us consider stationary discrete memoryless binary source with ones probabilities p. In binary arithmetic encoding codeword for sequence $\mathbf{x}^N = \{x_1, x_2, ..., x_N\}$, $x_i \in \{0,1\}$ is represented as $\lceil -\log_2 p(\mathbf{x}^N) + 1 \rceil$ bits of number

$$\sigma(\mathbf{x}^N) = q(\mathbf{x}^N) + p(\mathbf{x}^N)/2, \tag{1}$$

where $p(\mathbf{x}^N)$ and $q(\mathbf{x}^N)$ are probability and cumulative probability of sequence $\mathbf{x}^N$ accordingly which can be calculated by using following recurrence formulas. If $x_i = 0$, then

$$\begin{cases} q(\mathbf{x}^i) \leftarrow q(\mathbf{x}^{i-1}) \\ p(\mathbf{x}^i) \leftarrow p(\mathbf{x}^{i-1}) \cdot (1-p). \end{cases} \tag{2}$$

If $x_i = 1$, then

$$\begin{cases} q(\mathbf{x}^i) \leftarrow q(\mathbf{x}^{i-1}) + p(\mathbf{x}^{i-1}) \cdot (1-p) \\ p(\mathbf{x}^i) \leftarrow p(\mathbf{x}^{i-1}) \cdot p. \end{cases} \tag{3}$$

In practice, integer implementation of arithmetic encoder is based on two v-size registers: `low` and `range` (see Algorithm 1). Register `low` corresponds to $q(\mathbf{x}^N)$, register `range` corresponds to $p(\mathbf{x}^N)$. The precision required to represent registers `low` and `range` grows with the increase of N. For decreasing coding latency and avoiding registers underflowing the *renormalization* procedure is used for each output symbol (see lines 8–26 in Algorithm 1).

As an alternative to arithmetic coders, *range coders* use bytes as output bit stream element and do byte renormalization at a time [16–18] (see lines 8–15 in Algorithm 2). In this paper the binary range coder with Subbotin's [19] renormalization is analyzed.

Algorithm 1 Binary symbol x_i encoding procedure in binary arithmetic coder

```
Input: $x_i$, low, range, counter
 1: len := range·p
 2: len := max{1,len}
 3: range := range − len
 4: if $x_i$ =1 then
 5:    low := low+range
 6:    range := len
 7: end if
 8: while range < QUARTER do
 9:    if low ≥ HALF then
10:       PUTBIT(1)
11:       for i=1,...,counter do
12:          PUTBIT(0)
13:       end for
14:       low := low − HALF
15:    else if low < QUARTER then
16:       PUTBIT(0)
17:       for i=1,...,counter do
18:          PUTBIT(1)
19:       end for
20:    else
21:       counter := counter + 1
22:       low := low − QUARTER
23:    end if
24:    low := low/2
25:    range := range/2
26: end while
```

2 Integer implementations of probability estimation

2.1 Sliding window and its approximations

Algorithms of adaptive data encoding based on *sliding window* are widely known. The probability of source symbol is estimated by analysis of special buffer contents [11]. It keeps W previous encoded symbols, where W is the length of the buffer. After encoding of each symbol the buffer's content is shifted by one position, new symbol is written to the free cell and the earliest symbol in buffer is erased. This buffer is called sliding window after the method of buffer content manipulation.

Algorithm 2 Binary symbol x_i encoding procedure in binary range coder

```
Input: $x_i$, low, range
 1: len := range·p
 2: len := max{1,len}
 3: range := range − len
 4: if $x_i$ =1 then
 5:    low := low+range
 6:    range := len
 7: end if
 8: while (low ⊕ (low+range))<TOP ∨ (range<BOTTOM) do
 9:    if range<BOTTOM ∧ ((low ⊕ (low+range)))≥TOP then
10:       range:= −low ∧ BOTTOM−1
11:    end if
12:    PUTBYTE(low·$2^{-24}$)
13:    range:=range·$2^{-8}$
14:    low:=low·$2^{-8}$
15: end while
```

For binary sources probability of ones is estimated by Krichevsky-Trofimov [12] formula

$$\hat{p}_{t+1} = \frac{S_t + 0.5}{W + 1}, \tag{4}$$

where S_t is the number of ones in the window before encoding symbol with the number t.

The advantage of using the sliding window is the opportunity of precise evaluation of source statistics and fast adaptation to changing statistics. However, the window has to be stored in the encoder and decoder memory, which is a serious disadvantage of this algorithm. To avoid it the *Imaginary Sliding Window* technique (ISW) proposed for a binary source [13] and for non-binary source [11]. The ISW technique does not require window content storage and estimates count of symbols from source alphabet stored in the window.

Let us consider the ISW method for a binary source. Define $x_t \in \{0, 1\}$ as source input symbol with number t, $y_t \in \{0, 1\}$ as symbol deleted from the window after addition of x_t. Suppose at every time instant a symbol in a random position is erased from the window instead of the last one. Then the number of ones in the window is recalculated by the following recurrent randomized procedure.

Step 1. Delete a random symbol from the window

$$S_{t+1} = S_t - y_t, \tag{5}$$

where y_t is a random value generated with probabilities

$$\begin{cases} Pr\{y_t = 1\} = \dfrac{S_t}{W}, \\ Pr\{y_t = 0\} = 1 - \dfrac{S_t}{W}. \end{cases} \tag{6}$$

Step 2. Add a new symbol from the source

$$S_{t+1} = S_{t+1} + x_t. \tag{7}$$

For implementation of ISW algorithm a random variable must be generated. This random variable should take the same values at the corresponding steps of encoder and decoder. However, there is a way to avoid generating a random variable [14]. At step 1 of the algorithm let us replace random value y_t with its probabilistic average. Then the rule for recalculating number of ones after encoding of each symbol x_t can be presented in two steps.

Step 1. Delete an average number of ones from the window

$$S_{t+1} = S_t - \frac{S_t}{W}. \tag{8}$$

Step 2. Add a new symbol from the source

$$S_{t+1} = S_{t+1} + x_t. \tag{9}$$

By combining (8) and (9), the final rule for recalculating number of ones can be given as follows:

$$S_{t+1} = \left(1 - \frac{1}{W}\right) \cdot S_t + x_t. \tag{10}$$

2.2 Probability estimation based on state machine

One way for implementation of probability estimation can be based on the *state machine* approach. Each state of this machine corresponds to some probability value. Transition from state to state is defined by the value of the input symbol. This approach does not require multiplications or divisions for probability calculation. In addition, the fixed set of states allows to implement the multiplication-free arithmetic encoding [5].

For example, let us consider state machine based probability estimation in H.264/AVC standard [15]. Input symbols are divided into two types: Most Probable Symbols (MPS) and Least Probable Symbols (LPS). State machine contains 64 states and is based on equation (10). Each state defines probability estimation for Least Probable Symbol. Set of probability values $\{\hat{p}_0, \hat{p}_1, ..., \hat{p}_{63}\}$ is defined as:

$$\begin{cases} \hat{p}_i = (1-\gamma)\hat{p}_{i-1}, \text{ where } i = 1, ..., 63, \hat{p}_0 = 0.5, \\ \gamma = 1 - \left(\frac{\hat{p}_{min}}{0.5}\right)^{\frac{1}{63}}, \hat{p}_{min} = 0.01875. \end{cases} \tag{11}$$

Probability estimation for symbol x_{t+1} is calculated as

$$\hat{p}_{t+1} = \begin{cases} (1-\gamma)\hat{p}_t + \gamma, \text{ if } x_t = \text{LPS}, \\ \max\{(1-\gamma)\hat{p}_t, \hat{p}_{62}\}, \text{ if } x_t = \text{MPS}. \end{cases} \tag{12}$$

2.3 Probability estimation based on virtual sliding window

Probability estimation using “Virtual Sliding Window“ [14] is also based on equation (10), but it does not use state machine for probability calculation. For this algorithm probability estimation that x_{i+1} is equal to one is defined as

$$\hat{p}_{i+1} = \frac{s_i}{2^{2w}}, \tag{13}$$

where 2^{2w} is a window length and s_i is a virtual sliding window state which is recalculated by the following rule:

$$s_{i+1} = \begin{cases} s_i + \left\lfloor \dfrac{2^{2w} - s_t + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_i = 1 \\ \\ s_i - \left\lfloor \dfrac{s_i + 2^{w-1}}{2^w} \right\rfloor, \text{ if } x_i = 0. \end{cases} \tag{14}$$

For stationary memoryless sources window length expansion increases the probability estimation precision and improves compression rate. For arbitrary source window length expansion may reduce estimation precision. Therefore, optimal window length selection is a complex problem because statistical properties of a binary source are unknown. In [14] the following simple heuristic algorithm of window length selection is proposed. Let us define $L = \{2^{2w_1}, 2^{2w_2}, ..., 2^{2w_l}\}$ as a set of window lengths. The output of the binary source is encoded and then window length is selected from the set L. During encoding probability estimations $\hat{p}_i(w_1), \hat{p}_i(w_2), ..., \hat{p}_i(w_l)$ are calculated. After encoding, bit stream length estimation is calculated by equation: $\hat{R}(w_k) = \sum_i \hat{r}_i(w_k)$, where

$$\hat{r}_i(w_k) = \begin{cases} -\log_2 \hat{p}_i(w_k), \text{ if } x_i = 1, \\ -\log_2 (1 - \hat{p}_i(w_k)), \text{ if } x_i = 0, \end{cases} \tag{15}$$

and window length w^* is assigned by equation

$$w^* = \arg\min_k \hat{R}(w_k). \tag{16}$$

Thus, compression gain is reached by assigning specific window length selected by statistical properties of corresponding source. Therefore, Virtual Sliding Window provides better compression efficiency [14] in comparison to adaptation mechanism in H.264/AVC standard [15].

3 Computation complexity analysis

From Algorithm 1 follows, that lines 1–8 are used for each input binary symbol. On the other hand, amount of using of lines 9–25 is in direct proportion to number of bits in the output bit stream. Let use define N as a number of the input binary symbols, R as a size of the output bit stream. Therefore encoding time for binary arithmetic coder T_{arith} includes two main parts:

$$T_{arith} = \alpha_{arith} \cdot N + \beta_{arith} \cdot R, \tag{17}$$

where α_{arith} is the computation complexity of lines 1–8 per one input binary symbol, β_{arith} is the computation complexity of lines 9–25 per one output binary symbol.

Using the reasoning described above, encoding time for binary range coder (see Algorithm 2) can be written as:

$$T_{range} = \alpha_{range} \cdot N + \beta_{range} \cdot \frac{1}{8} \cdot R, \tag{18}$$

where α_{range} is the computation complexity of lines 1–8 per one input binary symbol, β_{range} is the computation complexity of lines 9–14 per one output byte.

It is known [20], that redundancy of integer implementation of arithmetic encoder depends on the number of bits for probabilities representation τ and bit size v of registers *low* and *range*. Therefore, the size of the output bit stream

$$R \approx N \cdot \left(h(p) + 2 \cdot (\tau + \log e) \cdot 2^{-(v-2)} \right), \tag{19}$$

where $h(p)$ is entropy of binary memoryless source with ones probabilities p, $v \geq \tau + 2$.

Equations (17), (18) and (19) show, that if probability of ones is known, then for given arithmetic coder implementation encoding time is the linear function of a number of input binary symbols and source entropy $h(p)$.

Values α_{arith}, β_{arith}, α_{range} and β_{range} depend on processor architecture. For simplification let us assume that $\alpha_{arith} \approx \beta_{arith} \approx \alpha_{range} \approx \beta_{range}$ and $v \gg \tau$, then from (17), (18) and (19) follows that

$$\frac{T_{arith} - T_{range}}{T_{arith}} \approx \frac{\frac{7}{8} \cdot h(p)}{1 + h(p)} \in [0, ..., 0.4375] \, . \tag{20}$$

Figure 1 shows the encoding time for 10^8 input binary symbols using Processor Intel Core 2 DUO, 3GHz. These results show that byte-oriented renormalization allows to decrease encoding time up to 40% in comparison with bit-oriented renormalization. In addition this figure shows that proposed linear model is fits for encoding time representation for Algorithms 1-2.

In real applications the probability of ones is not known. In this case for input binary symbol x_i the probability estimation of ones $\hat{p}_i$ is calculated and used in line 1 of Algorithms 1-2 instead p. In this case, the size of output bit stream can be calculated as

$$R \approx \sum_{i=0}^{N-1} r_i, \tag{21}$$

where

$$r_i = \begin{cases} -\log_2 \hat{p}_i, \text{ if } x_i = 1, \\ -\log_2 (1 - \hat{p}_i), \text{ if } x_i = 0, \end{cases} \tag{22}$$

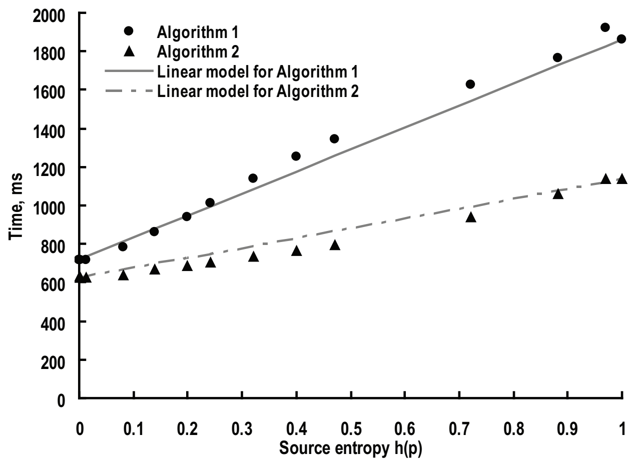


Fig. 1. Encoding time for $N = 10^8$ in case when probability is known

Equations (17), (18) and (21) show, that the encoding time for adaptive binary arithmetic coder is the linear function of a number of input binary symbols and the output bit rate which depends on precision of the probability estimation $\hat{p}_i$.

Figure 2 shows the encoding time for Algorithm 1 in case of binary arithmetic encoder with probability estimation using state machine (as in H.264/AVC standard) and "Virtual Sliding Window" with parameters $w = 4$ and $w = 8$. This figure shows that both probability estimation algorithms require additional computation complexity. At the same time, "Virtual Sliding Window" allows to decrease the encoding time up to 10% (for $w = 8$) in comparison to probability estimation algorithm from H.264/AVC standard.

4 Conclusion

In this paper we have proved that in case of binary memoryless sources with known probability distribution encoding time for binary arithmetic encoder is a linear function of a number of input binary symbols and source entropy. We have shown that the adaptive binary arithmetic encoder implementation based on byte-oriented renormalization and probability estimation using "Virtual Sliding Window" has significantly less computational complexity than binary arithmetic

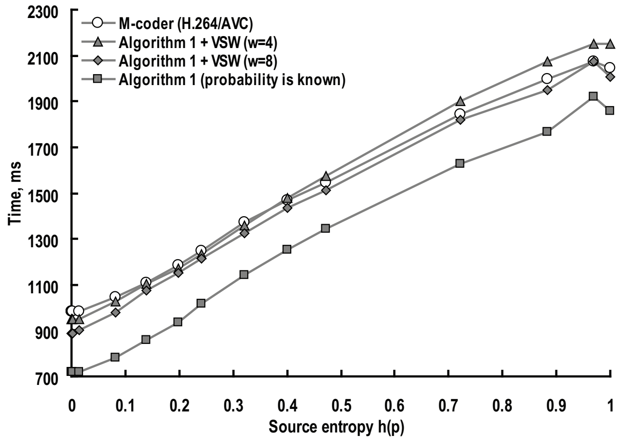


Fig. 2. Encoding time for $N = 10^8$ in case of binary arithmetic encoder with probability estimation using state machine and "Virtual Sliding Window"

encoder from H.264/AVC standard. Therefore, it is more preferable as an entropy encoding method for future video compression standards or state of the art codecs.

This work is supported by the Russian Foundation for Basic Research, project 10-08-01071-a and by the project of NSFC International Young Scientists.

References

1. ITU-T and ISO/IEC JTC1, "Digital Compression and cod- ing of continuous-tone still images", ISO/IEC 10918-1 ITU-T Recommendation T.81 (JPEG), 1992.
2. ITU-T and ISO/IEC JTC 1, "JPEG 2000 Image Coding System: Core Coding System, ITU-T Recommendation T.800 and ISO/IEC 15444-1" *JPEG 2000 Part 1*, 2000.
3. Advanced video coding for generic audiovisual services, *ITU-T Recommendation H.264 and ISO/IEC 14496-10 (AVC)*, 2009.
4. H. Eeckhaut, B. Schrauwen, M. Christiaens, J. Campenhout "Speeding up Dirac's entropy coder", *Proc. 5th WSEAS Int. Conf. on Multimedia, Internet and Video Technologies*, pp. 120–125, 2005.
5. W.B. Pennebaker, J.L. Mitchel, G.G. Langdon, R.B. Arps, "An overview of the basic principles of the q-coder adaptive binary arithmetic coder", *IBM J. Research and Development.* V.32. pp.717–726, 1988.

6. E. Belyaev, "Low bit rate video coding based on three-dimensional discrete pseudo cosine transform", *International Conference on Ultra Modern Telecommunications*, 2010.
7. A. Said, "Comparative analysis of arithmetic coding computational complexity", Hewlett-Packard Laboratories Report, HPL-2004-75, 2004.
8. High Efficiency Video Coding, `http://www.h265.net/`
9. X. Lu, Y. Wang, and E. Erkip, "Power efficient H.263 video transmission over wireless channels", *International Conference on Image Processing*, pp. 533-536, 2002.
10. Z. He, Y. Liang, L. Chen, I. Ahmad, D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints", *IEEE Transactions on Circuits and Systems for Video Technology*, vol.15, pp.645-658, 2005.
11. B. Ryabko, "Imaginary sliding window as a tool for data compression", *Problems of Information Transmission*, pp. 156-163, 1996.
12. E. Krichevski and V. Trofimov, "The performance of universal encoding", *IEEE Transactions on Information Theory*, vol. IT-27, pp. 199-207, 1981.
13. T.Leighton and R.L.Rivest, "Estimating a probability using finite memory", *IEEE Transactions on Information Theory*, vol. IT-32, pp. 733-742, 1986.
14. E. Belyaev, M. Gilmutdinov, A. Turlikov, "Binary arithmetic coding system with adaptive probability estimation by Virtual Sliding Window", *Proceedings of the 10th IEEE International Symposium on Consumer Electronics*, pp.194–198, 2006.
15. Marpe D., Schwarz H., Wiegand T., "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard", *IEEE Transactions on Circuits and Systems for Video Technology*, vol.7. pp.620–636, 2003.
16. Schindler M.A., "Byte oriented arithmetic coding", *Proceedings of Data Compression Conference*, 1998.
17. D. Vatolin, "Data compression methods", *Dialog-MIFI Publisher, Moscow*, 2002. (in Russian)
18. P. Lindstrom, M. Isenburg, "Fast and Efficient Compression of Floating-Point Data", *IEEE Transactions on Visualization and Computer Graphics*, Vol.12, Iss.5, pp.1245 – 1250, 2006.
19. D.Subbotin, "Carryless Rangecoder", 1999. `http://search.cpan.org/src/SALVA/Compress-PPMd-0.10/Coder.hpp`.
20. B.Y. Ryabko, A. N. Fionov, "An efficient method for adaptive arithmetic coding of sources with large alphabets", *Problems of Information Transmission*, vol.35, No. 4, pp. 95–108, 1999.

A real-time simulcast multi-view wavelet video coding based on skipping of spatial subbands

Evgeny Belyaev, Karen Egiazarian and Moncef Gabbouj
Department of Signal Processing
Tampere University of Technology, Finland
Email: {evgeny.belaev, karen.egiazarian, moncef.gabbouj}.tut.fi

***Abstract*—In this paper a real-time simulcast multi-view video coding based on three-dimensional discrete wavelet transform (3-D DWT) is considered. An efficient rate-distortion criterion of skipping spatial subbands is proposed. A processing of subbands is done in a group of frames from low frequency to high-frequency temporal subbands and from low frequency to high-frequency spatial subbands. If for the processed subband in the current view it appears to be more efficient not to include the highest significant bit-plane into the output bit stream, then all the corresponding temporal and spatial child subbands are skipped without any calculations of 2-D wavelet transforms and entropy encoding. Moreover, all corresponding spatial subbands in sequel views (with its child subbands) are skipped as well. Simulations results have demonstrated that the 3-D DWT codec with the proposed skipping rule has much lower computational complexity (from 2 up to 8 times) for the same quality level compared to the H.264/AVC standard in the low complexity mode.**

I. Introduction

A real-time multi-view video capturing is an important component of new technologies such as 3D television, free viewpoint video and so on. In this case a scene is captured simultaneously by several video cameras generating a huge amount of data which cannot be stored or transmitted without compression. In the current capturing systems the M-JPEG standard, which is usually embedded in most of video cameras, is used for video coding (see, e.g., [1]). However, M-JPEG exploits only intra-frame redundancy and is not efficient from the compression point of view.

Since all the cameras capture the same scene, a high interview similarity can be exploited for an efficient compression. In the multi-view extension of the H.264/AVC standard (or H.264/MVC) this is achieved by using a combined temporal/inter-view prediction, which allows to reduce the video bit rate up to 50% [2] in comparison with simulcast case when all the videos are compressed independently. In the wavelet-based video coding a similar compression improvements are achieved due to an additional disparity-compensated view filtering utilizing the inter-view correlations [3], [4]. However, due to high computational complexity, these coding techniques can be used only for off-line coding or require a special equipment like multi-core platforms, GPU's etc.

In this paper we extend our previous work related to a single video compression based on the three-dimensional discrete wavelet transform (3-D DWT) [5], [6] and introduce a new real-time simulcast multi-view wavelet video coding with a rate-distortion efficient skipping of the spatial subbands. Simulation results show that the proposed approach significantly outperforms in compression efficiency an intra-frame coding and, on the other hand, has a much lower computational complexity (from 2 to 8 times) for the same quality level compared to the inter-frame H.264/AVC coding in the low complexity mode.

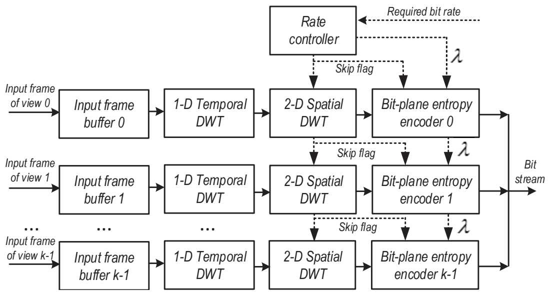


Fig. 1: The proposed multi-view video coding scheme

II. The proposed subband skipping approach

The proposed multi-view video coding scheme based on 3-D DWT is shown in Figure 1. First, a group of frames (GOF) of length N of each view are accumulated in the input frame buffers. Then, for each view, a one-dimensional multilevel DWT of length N in the temporal direction is applied. All frames in the GOF of each view are processed starting from low-frequency to high-frequency frames. For each frame, the spatial subbands are also processed from low-frequency to high-frequency spatial subbands (see Figure 2). Depending on a required bit rate, the rate controller chooses one of the following options for each spatial subband:

1) It permits a calculation of 2-D spatial transform, selects the Lagrange multiplier (which defines the rate-distortion trade-off for the subband) and then permits an entropy encoding for the subband;
2) It prohibits a calculation of the 2-D spatial DWT and entropy coding for the subband. At the decoder side, the corresponding transform coefficients in the subband are considered to be zero.

If the current wavelet subband is not skipped, then the bit-plane entropy encoder is applied, the Lagrangian sum is minimized by a selection of the lowest bit-plane number which is included into a bit stream taking into account a current Lagrange multiplier value λ defined by a virtual buffer fullness (see details in [5], [6])).

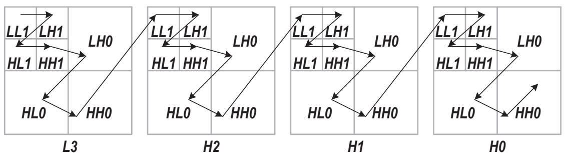


Fig. 2: Example of wavelet subbands encoding order

From the rate-distortion point of view it can be more efficient to not include the highest significant bit-plane into the output bit stream. In this situation the encoder is wasting computational resources to calculate the 2-D DWT and to form a subband bit stream which is not included in the output bit stream. Therefore, it is important to find a criterion which can guarantee with a high probability that the current subband will be skipped *before* processing of this subband. Similarly to our previous work [6], for subband skipping we use the following *coding assumptions*. If for any subband of the current view, the Lagrange sum is minimized when the current subband bit stream is not included in the output bit stream, then

1) All corresponding temporal-spatial child-subbands in this view are skipped.
2) All corresponding spatial subbands in sequel views (with its child subbands) are skipped as well.

Figure 3 illustrates the proposed approach for three view and a GOF size $N = 4$ with two-level temporal and two level spatial wavelet decomposition. Frames after temporal wavelet decomposition are denoted as L3, H2, H1, H0, where L3 is the low-frequency frame. In this example, if the spatial subband HH1 in the frame H2 of view 0 is skipped then the corresponding child-subbands HH0 in the frame H2, HH1 and HH0 in the frame H1, and HH1 and HH0 in the frame H0 are skipped without any processing. Additionally, corresponding HH1 subbands in views 1 and 2 are skipped together with their child subbands.

III. Simulation results

Simulation results were obtained for the test multi-view video sequences “Vassar“, “Ballroom“ and “Exit“ [7] which consist of 8 views. Each view has a frame resolution 640×480, and a frame rate 30 frames per second. For our experiments, the proposed video coding algorithm is compared with the x.264 codec [8] which, as shown in [9], provides close to optimum rate-distortion performance for the H.264/AVC standard when a computational complexity is significantly restricted. Therefore, this codec can be used as an upper bound of the rate-distortion performance which can be achieved by H.264/AVC standard in a low complexity case.

The proposed codec was run with GOF size $N = 16$ using the Haar wavelet transform in the temporal direction and the 5/3 spatial lifting wavelet transform at three decomposition levels[1]. x.264 codec was run in a very low complexity mode which corresponds to the Baseline profile of H.264/AVC with intra-frame period 16. Additionally, for “Vassar“ we have measured the rate-distortion-complexity performance for x.264 codec in a very low complexity mode with an Intra-frame coding only. In all cases the codecs compress test sequences in simulcast way, i.e. each view can be decoded independently. The encoding speed in this paper is defined as the number of frames which can be encoded in one second on the hardware platform with processor Intel Core 2 DUO CPU 3.0GHz. The computational complexity is considered as an inverse value of the encoding speed. In all cases, the codecs were simulated using a constant bit rate mode and the encoding speed was measured without any use of assemblers, threads, or other program optimization techniques.

[1]The proposed codec can be found at http://www.cs.tut.fi/~belyaev/3d_dwt.htm

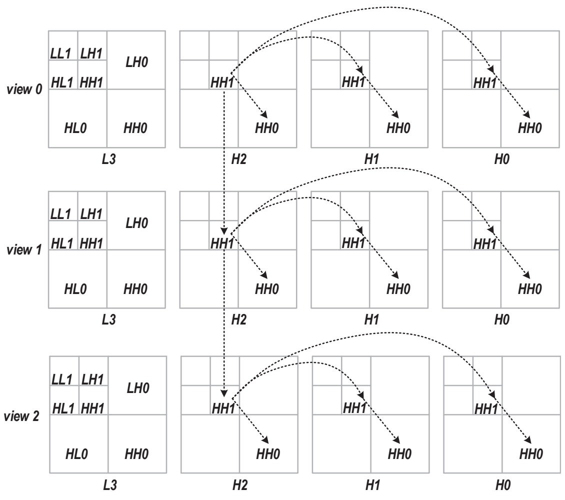


Fig. 3: The proposed skipping approach

TABLE I: Encoding speed (in fps) for Y-PSNR=35dB

Test video	x.264 ultrafast (INTRA)	x.264 ultrafast	3-D DWT with the proposed skipping
“Vassar“	6.5	10.8	35.2
“Ballroom“	6.4	8.2	22.5
“Exit“	6.5	10.3	34.8

Simulation results, presented in Figure 7, show the rate-distortion-complexity comparison for the considered codecs. One can see, that the proposed skipping approach allows to significantly reduce the computation complexity of 3-D DWT based scheme. Furthermore, 3-D DWT video codec with the proposed skipping provides from 2 to 8 times lower computational complexity for the same Y-PSNR level compared to the H.264/AVC in the low complexity mode. Finally, it is easy to notice that intra-frame coding only is the worst in terms of quality and complexity. Table I shows the encoding speed for case, when the codecs are used for multi-view video coding with visual lossless quality which is typically equal to Y-PSNR=35dB. One can see, that x.264 does not provide real-time multi-view video coding for the considered processor, while 3-D DWT codec is able to encode in real-time (“Vassar“ and “Exit“) or close to real-time (“Ballroom“) mode.

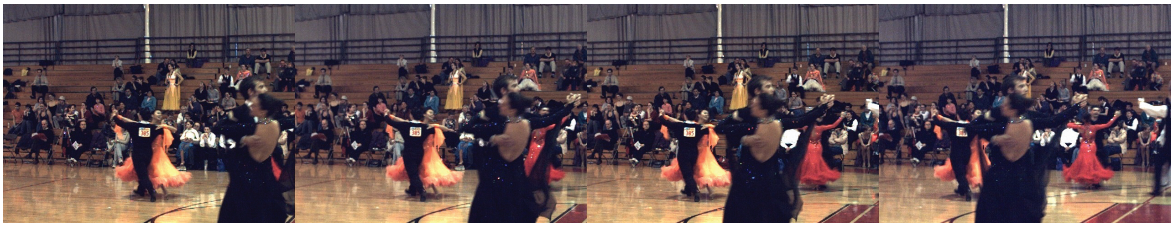


Fig. 4: Original views

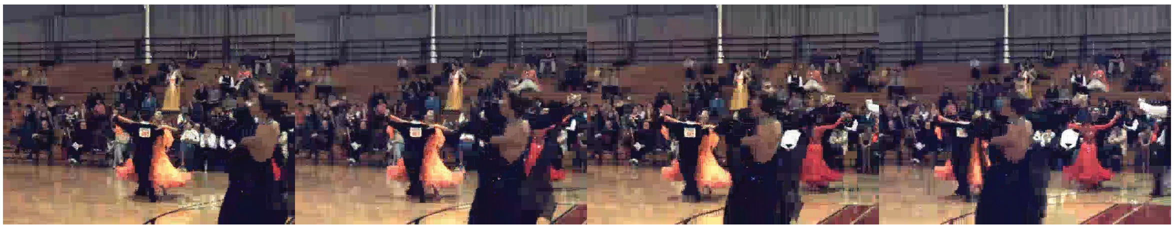


Fig. 5: x.264 ultrafast mode, Bit rate = 2000 kbps, Y-PSNR=26.422, Encoding speed = 10.6 fps

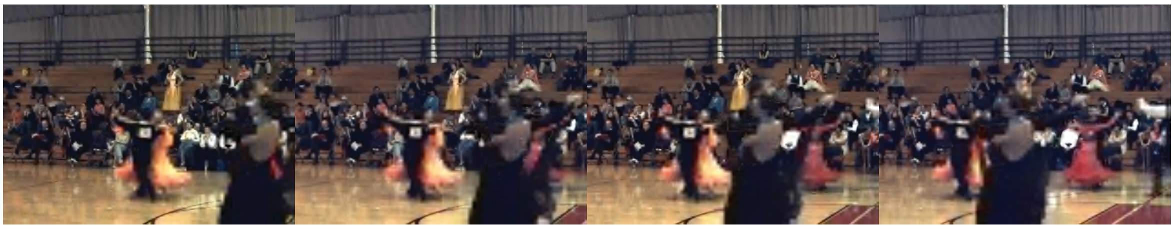


Fig. 6: The proposed codec, Bit rate = 2000 kbps, Y-PSNR=27.866, Encoding speed = 45.8 fps

For a visual assessment we depicted frame 90 of views 0,2,4, and 6 of video sequence “Ballroom“, when the required bit rate is 2000 kbps (see Figures 4–6). One can see, that the visual quality for the proposed codec is comparable with x.264 in the low-complexity mode.

IV. Conclusion

In this paper we have presented the 3-D DWT based video codec with a subband skipping rule. Our simulations show that the proposed codec has much lower computation complexity than H.264/AVC standard in a low complexity mode. Additionally, in contrast to H.264/AVC, the proposed codec provides temporal and spatial scalability due to natural properties of a wavelet transform. Therefore, our video compression scheme can be more preferable than H.264/AVC in applications where real-time multi-view coding is needed. The proposed codec is not compared with a multi-view extension of the H.264/AVC standard, because the authors have not found any open source fast software implementation of it. However, taking into account that H.264/MVC uses additional inter-view prediction, its computation complexity should be higher that H.264/AVC in the simulcast mode.

References

[1] F.Marton, E.Gobbetti, F.Bettio, J.A.I.Guitian, R.Pintus, “A real-time coarse-to-fine multi-view capture system for all-in-focus rendering on a light-field display“, *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, 2011.

[2] P.Merkle, A.Smolic, K.Muller, T.Wiegand, Member, “Efficient Prediction Structures for Multi-view Video Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.17, Is.11, pp. 1461 – 1473, 2007.

[3] Y.Liu, K. Ngan, “Fully scalable multi-view wavelet video coding“, *IEEE International Symposium on Circuits and Systems*, 2009.

[4] W.Yang, Y.Lu, F.Wu, J.Cai, K.Ngan, S.Li,“4-D Wavelet-Based Multiview Video Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.16, Is.11, pp.1385–1396, 2006.

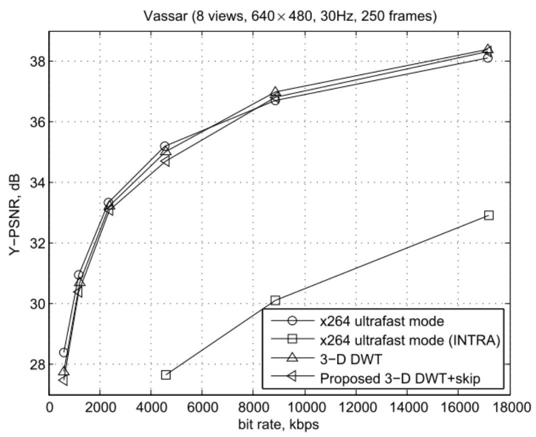
[5] E. Belyaev, K. Egiazarian and M. Gabbouj, “Low complexity bit-plane entropy coding for 3-D DWT based video compression“, *The International Symposium on SPIE Electronic Imaging*, 2012.

[6] E. Belyaev, K. Egiazarian and M. Gabbouj, “A low-complexity bit-plane entropy coding and rate control for 3-D DWT based video coding“, *IEEE Transactions on Multimedia*, 2013 (accepted).

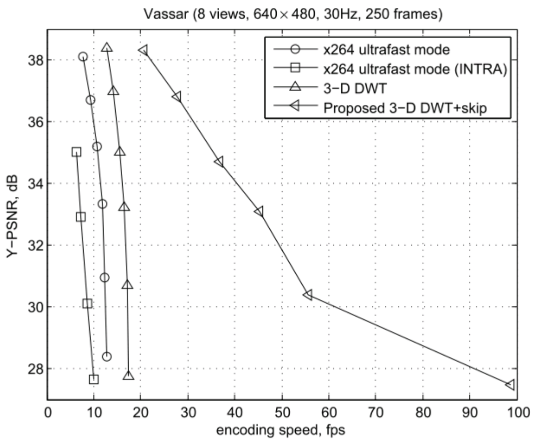
[7] MVC test sequences, http://www.merl.com/pub/avetro/mvc-testseq/

[8] x.264 video codec, http://x264.nl/

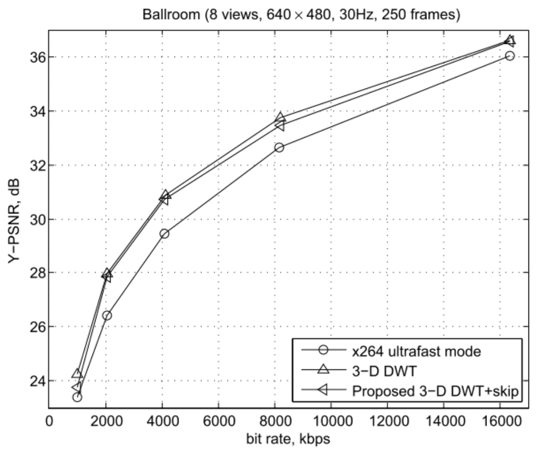
[9] X. Li, M. Wien, J.-R. Ohm, “Rate-Complexity-Distortion Optimization for Hybrid Video Coding“, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.21, pp.957 – 970, 2011.



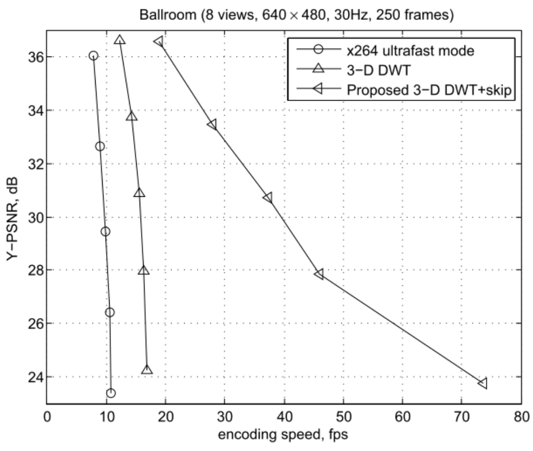
(a) Rate-distortion comparison for "Vassar"



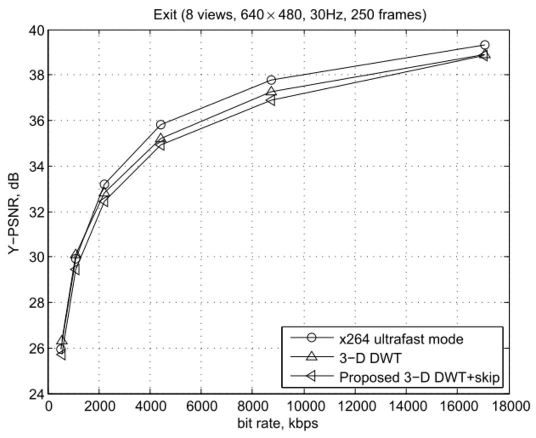
(b) Complexity-distortion comparison for "Vassar"



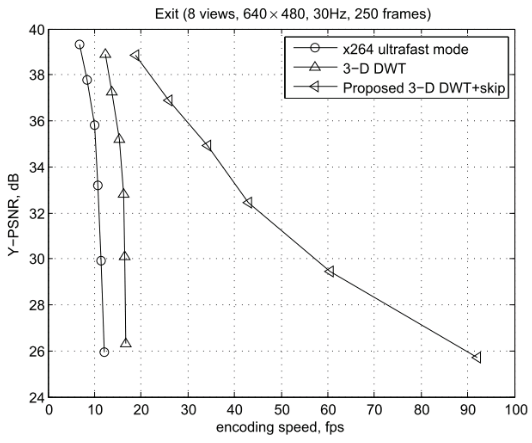
(c) Rate-distortion comparison for "Ballroom"



(d) Complexity-distortion comparison for "Ballroom"



(e) Rate-distortion comparison for "Exit"



(f) Complexity-distortion comparison for "Exit"

Fig. 7: Rate-distortion-complexity comparison of different low complexity multi-view video coding schemes

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O.B. 527
FI-33101 Tampere, Finland

ISBN 978-952-15-3463-8
ISSN 1459-2045