Architecture of Algorithmically Optimized MPEG-4 AVC/H.264 Video Encoder

Tomasz Grajek, Damian Karwowski, Adam Łuczak, Sławomir Maćkowiak, and Marek Domański

Chair of Multimedia Telecommunications and Microelectronics
Poznan University of Technology, ul. Polanka 3, 60-965 Poznań, Poland
{tgrajek,dkarwow,aluczak,smack,domanski}@et.put.poznan.pl

Abstract. Architecture of algorithmically optimized MPEG-4 AVC/H.264 video encoder is presented in the paper. The paper reveals details of implementation for the proposed MPEG-4 AVC video encoder. The presented MPEG-4 AVC encoder was tested with test video sequences from the point of view of computational performance and coding efficiency. The runtime of the optimized video encoder is 37 to 132 times smaller relative to runtime of the reference MPEG-4 AVC encoder for comparable encoder compression performance.

Keywords: Video encoder, MPEG-4 AVC, H.264, fast MPEG-4 AVC.

1 Introduction

Digital video compression is of a great importance in many fields of communication and information technology. A large variety of video compression techniques were presented in the literature [13]. Nevertheless, hybrid video technology is mostly used in communication systems and is a cornerstone of all major contemporary video coding standards (MPEG-2, H.263).

Appearance of the high definition television increased requirements for even higher compression performance. It was the motivation to improve existing video coding techniques. Intensive research conducted in this area resulted in a new generation high-performance video encoders like VC-1, AVS, MPEG-4 AVC/H.264 [1–3, 9, 10]

Among video encoders of a new generation, MPEG-4 AVC/H.264 worldwide video compression standard is of a great importance due to its superior compression performance in comparison to other technologies [1–3, 5, 7, 8]. As a matter of fact the works are currently in progress on future HEVC technology. Nevertheless, MPEG-4 AVC is currently the most popular solution and is putting into practice in many areas including limited bandwidth multimedia services, HDTV television, IPTV and videoconference systems.

2 MPEG-4 AVC Video Compression Technology

The main idea behind MPEG-4 AVC remains unchanged comparing to older video compression standards (MPEG-2, H.263). MPEG-4 AVC exploits com-

monly known hybrid video coding scheme with intra- and inter- frame prediction, transform coding, and entropy coding of residual signal. Nevertheless, relative to older video compression stan ards each mechanism of a successive video codec was significantly improved. The mechanisms of MPEG-4 AVC exploit the context-based coding paradigm in which, data of an image block is encoded with respect to data of neighboring blocks. Additionally, some new coding tools were also added to MPEG-4 AVC that had not been used in older standards.

Significant improvements were put to intra- prediction mechanism. In order to efficiently represent intra-predicted blocks of an image, 26 prediction modes were de-fined that allow to perform prediction of an image content in 16x16 luma blocks (4 modes), 8x8 luma blocks (9 modes), 4x4 luma blocks (9 modes) and chroma blocks (4 modes). Each predictor realizes idea of context-based coding by the use of data from neighboring blocks. Depending on local content of an image, one of 26 predictors is chosen by encoder.

Mechanism of inter- frame prediction is also much more sophisticated relative to older encoders. In order to adapt to local content of an image, inter-frame prediction is realized in blocks of variable size. MPEG-4 AVC allows partition of a macroblock into 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 blocks and perform inter-frame prediction independently in each block. Precision (accuracy) of motion-compensated prediction was additionally increased by the use of many reference frames (up to 16). Each 16x16, 16x8, 8x16 and 8x8 block of a macroblock can use its individual reference frame. Concurrent using of many reference frames is also possible in a weighted prediction mode. Apart from that, MPEG-4 AVC realizes motion-compensated prediction with quarter-pixel accuracy, which reduces energy of residual signal.

Residual signal that is a result of intra-/inter- frame prediction is finally processed with transform coding scheme using an integer, DCT-like transformation and quantization of transform coefficients. In order to extra reduce statistical dependency that exists within data after quantization, entropy coding is used. MPEG-4 AVC exploits two adaptive entropy coding methods: Context-based Adaptive Variable Length Coding (CAVLC) together with Exp-Golomb codes and Context-based Adaptive Binary Arithmetic Coding (CABAC) [4]. These techniques make a major step forward in the field of entropy coding.

Quantization of transform coefficients allows flexible control of the bitrate but also leads to blocking artifacts in reconstructed images. In order to reduce this annoying effect, deblocking filter is used in a prediction loop of both the encoder and the decoder. This is a new tool, that improves subjective quality of the decoded images.

3 Research Problem

Application of the context-based coding methods in MPEG-4 AVC leads to achieving high compression performance of encoder. Nevertheless, the context-based coding scheme makes the algorithms computationally intensive, irregular, and it is extremely difficult to perform computations in parallel in a such case.

Moreover, a great number of different coding modes used in MPEG-4 AVC (26 modes for intra- prediction only, inter prediction performed in blocks of variable size) makes the mechanism of encoder control really complex.

The abovementioned facts cause high computational complexity of MPEG-4 AVC video encoder and great difficulties in realization of real-time compression system even for modern high performance multimedia processors.

An important research problem to solve is the architecture of optimized MPEG-4 AVC video encoder that will be able to perform (near) real-time video compression of standard definition signal when operating on x86 platform. As a matter of fact there already exists fast realizations of MPEG-4 AVC encoder with both the low- and high- level optimizations of encoder program code [14]. Nevertheless, an architecture of the optimized High Profile MPEG-4 AVC encoder with algorithmic optimizations of functional blocks makes the topic of the paper. There is proposed the original structure of such an optimized encoder. Presented MPEG-4 AVC encoder was realized at Chair of Multimedia Telecommunications and Microelectronics as an implementation project.

4 Algorithmically Optimized MPEG-4 AVC Encoder

In order to reduce the number of processor operations when encoding a video, the optimized structure of MPEG-4 AVC video encoder was proposed. The encoder was implemented from scratch in high level C programming language. All computationally complex functional blocks of encoder were algorithmically optimized towards speed. Functional blocks of the encoder were optimized taking into account both the specificity of MPEG-4 AVC technology (hybrid coding scheme, application of context-based coding mechanisms) and general features of x86 target platform (relatively small size of fast cache-memory in a processor and ability to use vector operations). The goal was to propose highly optimized towards speed version of video encoder intended for single processor platforms.

From the encoding time point of view the following parts of MPEG-4 AVC encoder are particularly crucial: access time to context data, the way of motion estimation, the way of encoder mode selection, realization of entropy coding methods. In order to speed up encoding process, all these parts of video encoder were highly optimized in the proposed architecture of MPEG-4 AVC encoder.

4.1 Implementation of Dedicated Data Buffer in Encoder

MPEG-4 AVC exploits context-based coding techniques, in which data of an image block is encoded with respect to information from neighboring blocks (with respect to context data). The location of the neighbors in an image is not fixed. Depending on the coding mode chosen for the current image block (frame mode or field mode) the coordinates of left, upper, upper-left, upper-right neighboring blocks can be different within an image. Therefore, these coordinates must be calculated each time before encoding successive blocks of an image for each syntax element that is encoded using context-based paradigm. Additionally, changing

coordinates of neighbors results in copying the context data from different parts of an image. It makes the mechanism of preparing the context information in encoder very irregular and complex.

In order to optimize this process, dedicated data buffers were implemented in encoder for samples of an image as well as for other context information. The structure of dedicated data buffers were presented in Fig. 1 In Fig. 1 data of a current mackroblock are stored in white blocks whereas context information is stored in grey blocks. Prior to encoding a given mackroblock, the buffers must be filled with context data taken from appropriate parts of an image. This way complex process of calculating the context information is carried out only once per mackroblock and not for individual syntax element separately. Once the buffers are filled in, context data can easily be reached by simple calling within dedicated memory. Besides, due to small size of buffers (data related to one macroblock only) they can be put into processor's cache memory which significantly reduces processing time.

4.2 Fast Motion-Compensated Prediction

Motion estimation is one of the most computationally intensive parts of contemporary video encoders. In MPEG-4 AVC motion-compensated prediction for inter-coded frames can be performed using 1-pel, $\frac{1}{2}$-pel, and $\frac{1}{4}$-pel accuracy. It strongly affects computational complexity of video encoder. In order to speed up the process, fast mechanisms of motion estimation were used in the proposed implementation of encoder [13]. In series of experiments, two such motion estimation methods were chosen: hexagonal motion estimation method for 1-pel accuracy and diamond motion estimation for $\frac{1}{2}$- and $\frac{1}{4}$-pel accuracy. Moreover, interpolation to $\frac{1}{2}$-pel is performed once for a whole image and stored in a memory, whereas interpolation to $\frac{1}{4}$-pel is performed on the fly. Such an approach is a tradeoff between speed, memory requirements, and access time to memory.

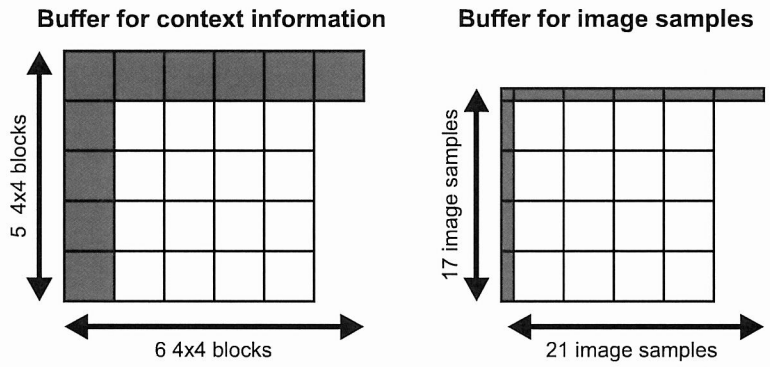

Fig. 1. Dedicated buffers for context data in the proposed optimized video encoder.

4.3 Fast Encoder Control Mechanism

High compression performance of MPEG-4 AVC was obtained by exploiting a huge number of different coding modes that can be adaptively switched depending on local content of an image. Nevertheless, it also strongly affects complexity of video encoder. Optimal solution (from the coding efficiency point of view) is to encode image blocks using each of individual coding mode separately in order to choose the best mode. However, such a solution would result in gigantic complexity of encoder and cannot be used in most applications.

In the proposed architecture of MPEG-4 AVC encoder fast control mechanism was used. This mechanism was worked out during experimental research taking into consideration statistics of selecting the individual coding modes in MPEG-4 AVC for natural video sequences.

In the proposed control mechanism, the idea of 'Early SKIP' mode was used. At the beginning, the encoder tries to encode a block using SKIP coding mode. SKIP mode - mackroblock is encoded with one bit only - there is no residual data for both the transform coefficients and the motion information. If the SKIP mode turns out to be good enough (it does not mean optimal) for the current macroblock, no other coding modes are tested. In the opposite situation, 16x16 and 8x8 coding modes are tested. If 8x8 coding mode does not improve compression (gives higher coding cost than for 16x16 mode), 16x16 coding mode is chosen. In the situation that 8x8 mode gives improvement (relative to 16x16), successive smaller divisions of 8x8 block are checked to test modes that operate on smaller blocks. The criterion for taking a decision is a result of simple filtering of an image using Sobel filter. The Sobel filter masks used to evaluate complexity of a texture were presented in Fig. 2 If the result of filtering is less than a predefined threshold Th, coding modes for blocks smaller than 8x8 are not tested.

4.4 Optimized Entropy Encoders

In the proposed architecture of encoder both variants of entropy encoder defined by MPEG-4 AVC standard were fully optimized and implemented (i.e. CAVLC+Exp-Golomb and CABAC). For the purpose of the paper, CABAC algorithm is more important.

CABAC algorithm realizes context-based entropy encoding of syntax elements. Previous research of the author on this topic revealed, that data statistics modeling is a bottleneck of contemporary adaptive arithmetic encoders (more

X			
	-1	-2	-1
	0	0	0
	1	2	1

Y			
	-1	0	1
	-2	0	2
	-1	0	1

Fig. 2. Sobel filter masks.

than 60% computations in entropy encoder). Therefore, the main optimization technique applied in this part of encoder was reducing the access time to all context data. It was achieved by adapting the implementation of CABAC to exploit the dedicated buffers of context information that were presented in section 4.1. As a result, the procedure of calculating the context data was accelerated several times. Other parts of CABAC (binarizarion and arithmetic encoder core) were implemented on the basis of C pseudo-code presented in the MPEG-4 AVC standard recommendation [1].

5 Methodology of Experiments

Computational complexity of the optimized MPEG-4 AVC encoder was thoroughly investigated with set of test video sequences. The goal was to explore the influence of algorithmic optimizations that were made in the video encoder on its encoding speed. For that reason, commonly available JM reference software of the MPEG-4 AVC standard with no algorithmic optimizations of encoder functional blocks was used as the anchor (version JM 13.2 of the software) [12]. Average encoding times of a frame were measured for both the optimized and the reference encoders. In order to obtain reliable experimental results, equivalent coding tools were activated in both versions of encoders. Experiments were done according to the following encoding scenario:

- Full HD test video sequences were used: BasketballDrive, BQTerrace, Cactus, Kimono1, ParkScene. The sequences were recommended by groups of experts ISO/IEC MPEG and ITU VCEG as a test material for research on new video compression technologies [6].
- Structure of group of pictures (GOP) was set to IBBPBBPBBPBBPBBP
- Experiments were done for a wide range of bitrates using different values of quantization parameter ($QP = 22, 27, 32, 37$). This results in the quality of a reconstructed video from excellent (QP=22) to very poor ($QP = 37$).
- CABAC entropy encoder was used.
- Threshold Th for fast control mechanism was set to 20000 for macroblocks on image border and 25000 for macroblocks on inside the image.
- The following platform was used: Intel(R) Core(TM) i7 CPU 950@3.07 GHz, 12 GB RAM, Windows 7 64-bit.

The two encoders produce two slightly different bitstreams (from the viewpoint of their size and quality of reconstructed videos). The degree of bitstreams variation was measured with Bjøntegaard metric [11]. The metric allows to compare the RD curves of two encoders in terms of bitrate reduction and PSNR gain based on 4 RD points (for $QP = 22, 27, 32, 37$ in experiments). Such tests were done for luma (Y) component.

6 Experimental Results

Experiments revealed, that it is possible to significantly speed-up computations in encoder using algorithmic optimizations of methods. Detailed results of encod-

ing times for none-optimized (reference) and optimized encoders were presented in Table 1. Results show extraordinary computational performance of the optimized encoder relative to the reference version. The optimized encoder is 37 to 132 times faster relative to the reference version. Higher run time ratios were noticed for higher value of QP (lower bitrate cases) due to smaller contribution of CABAC entropy encoder run time in total encoding time. The optimized software decreases the encoder run time 77 times on average in comparison to the reference encoder. What is very important, encoder speed-up was achieved with no quality degradation of the encoded sequence and with no essential increase of the bitstream size. For *BasketballDrive*, *BQTerrace* and *Cactus* relatively small increase of bitrate (1.07% - 2.3%) was observed for optimized encoder - this is equivalent to really small decrease of PSNR measure (0.02dB - 0.05dB) relative to the reference encoder. In the case of *Kimono1* and *ParkScene* sequences the optimized encoder produced bitstream of a size smaller by 1.7% - 2.01%

Table 1. Average encoding times of a frame for the reference and the optimized MPEG-4 AVC encoders in a function of QP value. The factor ratio is a ratio of run times of the reference and the optimized encoders respectively. **BD-Rate** represents average percentage difference of bitstream sizes for optimized and reference encoders respectively. **BD-PSNR Y** represents average difference for PSNR measure for sequences decoded with optimized and reference encoders respectively.

Sequence	BD-Rate [%]	BD-PSNR Y [dB]	QP	Avg. encoding time [ms/frame]		Ratio
				Ref	Fast AVC	
BasketballDrive	1.07	-0.02	22	18 029	406	44.41
			27	16 869	277	60.90
			32	16 489	203	81.23
			37	16 314	158	103.25
BQTerrace	1.63	-0.05	22	17 619	475	37.09
			27	16 427	293	56.06
			32	15 866	176	90.15
			37	15 779	119	132.60
Cactus	2.30	-0.05	22	17 143	401	42.75
			27	15 923	242	65.80
			32	15 592	169	92.26
			37	15 414	129	119.49
Kimono1	-1.70	0.05	22	16 911	346	48.88
			27	16 359	252	64.92
			32	16 094	185	86.99
			37	15 895	140	113.54
ParkScene	-2.01	0.07	22	16 760	386	43.42
			27	16 266	272	59.80
			32	15 989	183	87.37
			37	15 657	129	121.37
					Average	**77.61**

for equivalent video quality (which corresponds to 0.05db - 0.07dB increase of PSNR for equivalent bitrate).

7 Conclusions

Computational performance of MPEG-4 AVC video encoder can be significantly improved when applying algorithmic optimizations for encoder functional blocks. Optimization of encoding mechanisms allows to speed up encoder runtime by a factor of 37 to 132, depending on encoding scenario. On average, the throughput of optimized encoder is 77 times higher comparing to performance of an none-optimized version of encoder. What is very important, the optimized encoder has virtually the same compression performance as the reference encoder. Authors see the possibilities of further improving the computational performance of encoder when doing low-level optimizations of encoder program code together with multi-thread programming techniques.

References

1. ISO/IEC 14496-10 (MPEG-4 AVC) / ITU-T Rec. H.264: Advanced Video Coding for Generic Audiovisual Services (2010)
2. Special issue on H.264/AVC video coding standard. IEEE Trans. on Circuits and Systems for Video Technology, Vol. 13 (July 2003)
3. I. E. G. Richardson: H.264 and MPEG-4 Video Compression. Video Coding for Next-generation Multimedia, Wiley (2003)
4. D. Marpe, H. Schwarz, and T. Wiegand: Context-based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard. IEEE Trans. on Circuit and Systems for Video Technology, Vol. 13, No. 7, pp. 620–636 (2003)
5. J. Golston, A. R. Rao: Video Compression: System Trade-Offs with H.264, VC-1 and Other Advanced CODECs. Texas Instruments (2006)
6. ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6: Joint Call for Proposals on Video Compression Technology. MPEG doc. N11113, Kyoto, Japan (2010)
7. N. Kamaci and Y. Altunbasak: Performance Comparison of the Emerging H.264 Video Coding Standard with the Existing Standards. IEEE International Conference on Multimedia and Expo (ICME), Vol. 1 (6-9), pp. 345–348, Baltimore, USA (2003)
8. G. Sullivan and T. Wiegand: Video Compression - From Concepts to the H.264/AVC Standard. Proceedings of the IEEE, Special Issue on Advances in Video Coding and Delivery, Vol. 93, No. 1, pp. 18–31 (2005)
9. Society of Motion Picture and Television Engineers: VC-1 Compressed Video Bitstream Format and Decoding Proces. SMPTE 421M-2006 (2006)
10. Audio Video Coding Standard Workgroup of China (AVS): The Standards of People's Republic of China GB/T 20090.2-2006, Information Technology - Advanced Coding of Audio and Video - Part 2:Video (2006)
11. Gisle Bjøntegaard: Calculation of Average PSNR Differences between RD curves. ITU-T SG16/Q6, 13th VCEG Meeting, Doc. VCEG-M33, Austin, USA (2001)
12. H.264/AVC software coordination site: http://iphome.hhi.de/suehring/tml
13. J. W. Woods: Multidimensional Signal, Image, and Video Processing and Coding, Academic Press (2012)
14. x264 video codec - http://www.videolan.org/developers/x264.html.