

CODING TECHNIQUES IN MULTIVIEW VIDEO CODING AND JOINT MULTIVIEW VIDEO MODEL

Ying Chen¹, Miska M. Hannuksela², Ling Zhu³, Antti Hallapuro², Moncef Gabbouj¹, and Houqiang Li³

¹Department of Signal Processing, Tampere University of Technology

²Nokia Research Center

³University of Science and Technology of China

ABSTRACT

Since early 2006, Joint Video Team has been devoting on the development of Multiview Video Coding (MVC) standard as an extension of H.264/AVC. This MVC standard has been finalized in 2008. During the standardization of MVC, there was also a project namely Joint Multiview Video Model (JMVM), which focused on the advanced coding tools that are potentially useful. Those coding tools adopted into JMVM, including illumination compensation and motion skip, have not been added into MVC specification. In this paper, coding techniques in MVC as well as the tools in JMVM are described and discussed, focusing on the coding efficiency.

Index Terms— Multiview Video Coding, Joint Multiview Video Model, Coding Tool, Inter-view Prediction

1. INTRODUCTION

With the advances in acquisition and display technologies, 3D video is becoming a reality in the consumer domain with different application opportunities. 3D video applications can be grouped into two categories: free-viewpoint video [1] and 3D TV [2]. In free-viewpoint video, the viewer can interactively choose his/her viewpoint in 3D space to observe a real-world scene from preferred perspectives [1]. 3D TV refers to the extension of traditional 2D TV displays to displays capable of 3D rendering. Advanced auto-stereoscopic displays can support head-motion parallax, by decoding and displaying multiple views from different view-points simultaneously [2]. The content in above scenarios can be represented by multiple views, which of each is a traditional 2D sequence that was captured by a camera and can be used for 2D digital TV. Multiview video coding (MVC) is a key technology to enable a 3D video transmission system adopting multiview representation with efficient design for both transmission bandwidth and decoder resource consumption.

MPEG-2 and H.264/AVC can code two views by interleaving the left and right views in the temporal domain. In MPEG-2 Multiview Profile, one view, e.g., the left view can be coded in a reduced frame rate and the other view is coded as a temporal enhancement layer [3]. In H.264/AVC,

the stereo video information supplemental enhancement information (SEI) message was adopted to indicate how two views are arranged in one bitstream [4]. The two views can be alternating frames or complementary field pairs.

Recent interests and advances in 3D video display technologies have also driven the requirements of using 3D content of more than two views. To meet this request as well as other requirements described in [5], MPEG issued its “call for proposals” [6] and started the standardization of MVC. Among all the proposed MVC solutions, the one based on H.264/AVC hierarchical B-pictures coding was selected as the basis of the MVC codec [7]. Since early 2006, the standardization efforts of MVC have been continued in Joint Video Team (JVT).

JVT maintains a joint draft of the MVC specification as well as a Joint Multiview Video Model (JMVM). MVC usually refers to the joint draft, which has the latest version in [8]. However, JMVM is a superset of MVC. Specific coding tools that have been demonstrated as useful could be adopted by JVT into JMVM [9]. In MVC, inter-view prediction is realized so-called disparity motion compensation, which follows the H.264/AVC motion compensation processes.

Multiview coding techniques have been summarized in [10] and most of them have also been proposed to JVT. Those techniques include Illumination Compensation (IC), Motion Skip (MS), view interpolation prediction, adaptive reference filtering and asymmetric coding. IC and MS will be further discussed in Section 3. The basic idea of view interpolation prediction is to estimate depth/disparity for synthesize predictions [11][12]. The adaptive reference filtering algorithm filters the integer pixels of the inter-view reference frames, to compromise the potential problem of focus mismatch among views [13]. Asymmetric coding targets on the scenario wherein one view of a stereo pair is coded with a quarter resolution of the other [14][15]. It requires substantially less bandwidth and complexity without noticeably scarifying the subjective quality. IC and MS have been adopted into JMVM. Others were not adopted because they either provide marginal compression efficiency gain or can only be applied for limited use cases.

In this paper, the coding techniques in JMVM are introduced. These techniques, when combined in different

scenarios, are compared with MVC and simulcast AVC under MVC common test condition. It can be concluded that those tools provides a relatively low coding efficiency increase. The rest of the paper is organized as follows. In Section 2, background of MVC is given, including the bitstream structure, prediction relationship and illustration of inter-view prediction in. JMVM Coding tools are presented in Section 3. Rate distortion (RD) results and decoding complexity analysis are given in Section 4. Section 5 concludes the paper.

2. MVC CODING STRUCTURE

In an MVC bitstream, a picture in a specific time instance of a specific view is called a view component. The view components are coded into Network Abstraction Layer (NAL) units which are ordered in a time-first coding order. In time-first coding, view components of any temporal location are contiguous in decoding order. All the coded NAL units of one time instance forms an access unit, also referred to as an (coded) picture. Note that the coded view components of an access unit follow the view order, which may not follow the ascending order of view identifiers. All the first view components of access units form a base view, which is decodable by H.264/AVC.

In MVC, view dependencies for inter-view prediction are defined for each coded video sequence. Inter-view prediction is enabled within the view dependencies to remove redundancies among views. With the exception of inter-view prediction, view components of each view are coded with the tools supported by H.264/AVC. In particular, hierarchical temporal scalability was found to be efficient for multiview coding. A typical prediction structure of MVC is shown in Fig. 1. It is noted that the MVC standard provides a greater deal of flexibility than depicted in Fig. 1 for arranging temporal or view prediction references.

There are anchor pictures wherein all view components are anchor view components. Anchor view components and all the view components (in the same view) succeeding in output order (i.e. display order) can be correctly decoded without decoding of previous view components in decoding order (i.e. bitstream order) and thus can be used as random access points. Anchor pictures (e.g., for T0 and T8 in Fig. 1), can be set as required. In MVC common test condition [16], there is an anchor picture for every 0.5 second.

An anchor picture and a non-anchor picture may have different view dependencies. While in a coded video sequence, all anchor or non-anchor pictures have the same view dependency respectively. A view dependency specifies the directly dependent views for each view, and is signaled in the sequence parameter set (SPS). Dependent views are signaled separately for the views that may be used as reference pictures in the two reference picture lists, namely list 0 and list 1. The dependent views corresponding to list 0/list 1 are also called forward/backward dependent views. A view that has both forward and backward

dependent views is called a “B-view”, e.g., view 1, 3, 5 in Fig. 1. A view that has only forward dependent views is called a “P-view”, e.g., view 2, 4, 6, 7 in Fig. 1.

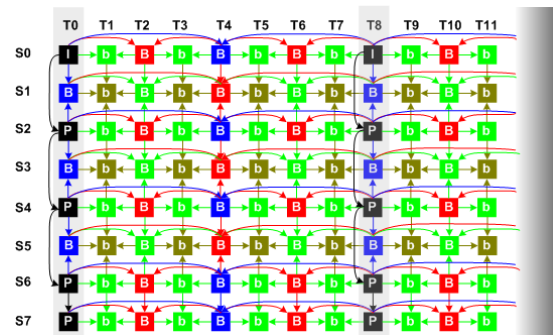


Fig. 1. Typical MVC prediction structure.

In MVC, view components in the same time instance can be used for inter-view prediction. It is utilized by putting the dependent view components into the reference picture lists of the view component that is being coded. In MVC, flexible reference picture list construction enables adding the view components of the time instance into list 0/1 and modifying the order of the inter-view prediction pictures and inter prediction pictures (view components in the same view) [17]. After the reference picture lists are constructed, the decoding processes of an MVC decoder are exactly the same as the processes of an H.264/AVC decoder.

3. CODING TOOLS IN JMVM

In JMVM, two coding tools are adopted on top of the MVC features. Illumination and color inconsistencies can happen due to different lighting conditions for multiple views. Although proper condition settings or preprocessing can also solve the problem, those are not always guaranteed. IC is the technique to solve this problem in the codec level, by subtracting the difference of the means of the reference block and the original block during motion compensation. This difference, namely locally illumination change are signaled for those blocks that use IC mode. At the decoder, an IC block is reconstructed by adding the motion compensation predictor, the residual and the illumination change, as shown in the following equation:

$$I(i, j) = R(i, j) + r(i+x, j+y) + C,$$

wherein $I(i, j)$ represents the reconstructed block, $R(i, j)$ is the residual signal, $r(i+x, j+y)$ is the reference block and C is the illumination change value for this block. At the encoder, each motion search needs an extra calculation of the means of the reference block.

MS is a motion vector (MV) derivation tool to enable reusing of the MVs from a view component in the same time instance but of other views by a given disparity for each macroblock (MB) that utilizes motion skip [19]. It is similar to SVC inter-layer motion prediction while does not enable motion refinement. In JMVM, a global disparity is

maintained and an offset is given for an MB with MS mode to calculate the local disparity, which is of 8-pixel accuracy. An example of motion reuse for a MB using MS mode is shown in Fig. 2. The current MB has a disparity points to four 8x8 blocks in an inter-view reference. The MVs of those blocks are reused for inter prediction motion compensation within the current view. The complexity increase at the decoder is minor, although the increase at encoder is substantially due to local disparity search.

4. SIMULATIONS

Our simulation followed the MVC common test condition defined in [16] and was based on the latest JMVM software [20]. The following scenarios are tested: Simulcast (each view is coded independently), MVC, IC (MVC inter-view prediction plus IC), MS (MVC inter-view prediction plus MS), IC+MS (MVC inter-view prediction plus two tools). For “P-views”, inter-view prediction only applies to anchor pictures in the common tests. Since MS applies only to non-anchor pictures with inter-view prediction enabled, MS is not enabled for the “P-views”. For IC, on the contrary, it works not only for inter-view prediction, but also for inter prediction. So, only the base view is not allowed to use IC, and a view component in a non-base view can be illumination compensated from its inter-view or inter prediction references.

The RD comparison results are shown in Table I, based on Bjontegaard measurement [21]. For each sequence, the compared Peak Signal-to-Noise Ratio (PSNR) and bitrate values are the average ones among all the views. MVC outperforms simulcast coding by an average bit-rate saving of 20%. MS and IC provide respectively 4% and 5% bitrate reduction over MVC, and in total, JMVM coding tools can achieve around 8.5% bit-rate saving, which is equivalent to 0.36 dB PSNR gain. Some typical RD curves are shown in Fig. 3, 4 and 5. It can be concluded that the coding efficiency improvements from simulcast to JMVM are mostly from the inter-view prediction tools defined in MVC. The coding tools, MS and IC, are potentially useful.

For the decoder, IC requires pixel level processing: adding a value to each block; MS requires only parsing and derivation for the MVs. Altogether, JMVM tools do not require significant complexity increase, although for hardware solutions, completely new modules need to be realized. MVC only requires slice header or higher syntax changes based on H.264/AVC, thus needs no new hardware implementations for major processing modules.

5. CONCLUSIONS

During the standardization of MVC, Joint Multiview Video Model has been maintained and it contains coding tools that are not included in the MVC specification. In this paper, those coding tools are reviewed and compared with the inter-view prediction in MVC. The JMVM coding tools,

illumination compensation and motion skip can provide an average bitrate saving of about 8.5%, under MVC common test condition, which is much less than the gain of MVC achieved on top of H.264/AVC simulcast coding. There is neither decoding complexity increase nor extra hardware efforts for a MVC decoder compared to H.264/AVC decoders. The decoder complexity increase of JMVM is minor, but new hardware implementations are required.

6. REFERENCES

- [1] A. Vetro, W. Matusik, H. Pfister, and J. Xin, “Coding approaches for end-to-end 3D TV systems,” in *Proceedings of the 23rd Picture Coding Symposium (PCS '04)*, pp. 319–324, San Francisco, Calif, USA, December 2004.
- [2] A. Smolic and P. Kauff, “Interactive 3-D video representation and coding technologies,” *Proceedings of the IEEE*, vol. 93, no. 1, pp. 98–110, 2005.
- [3] J.R. Ohm, “Stereo/Multiview Encoding Using the MPEG Family of Standards,” In *Proceedings of Electronic Imaging*, San Diego, USA, Jan. 1999.
- [4] ITU-T Rec. H.264 | ISO/IEC IS 14496-10, “Advanced video coding for generic audiovisual services, v3: 2005.”
- [5] ISO/IEC JTC1/SC29/WG11, “Requirements on Multi-view Video Coding v.5,” Doc. N7539, Nice, France, October 2005.
- [6] ISO/IEC JTC1/SC29/WG11, “Call for Proposals on Multi-view Video Coding”, Doc. N7327, Poznan, Poland, July 2005.
- [7] P. Merkle, K. Mueller, A. Smolic, and T. Wiegand, “Efficient Compression of Multi-view Video Exploiting Inter-view Dependencies Based on H.264/MPEG4-AVC”, *IEEE, International Conference on Multimedia and Expo (ICME) 2006*, Toronto, Canada, July 9-12 2006.
- [8] “Joint draft 9.0 on multi-view video coding,” *JVT-AB204*, Hannover, Germany, July 2008.
- [9] “Joint multiview video model (JMVM) 8.0,” *JVT-AA207*, Geneva, Switzerland, April 2008.
- [10] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G.B. Akar, G.A. Triantafyllidis and A.Koz. “Coding Algorithms for 3DTV — A Survey,” *IEEE Trans. on Circuits and Systems for Video Technology*, Vol 7, Issue 11, pp. 1606-1621, November 2007.
- [11] S. Yea, A. Vetro, “Report on CE10 on View Synthesis Prediction,” *JVT-U063*, Hanzhou, China, Oct. 2006.
- [12] H. Kimata, S. Shimizu, M. Tanimoto, T. Fuji, K. Yamamoto, “CE10: Proposal on View Interpolation Prediction for MVC,” *JVT-U093*, Hanzhou, China, Oct. 2006.
- [13] J. H. Kim, P. Lai, J. Lopez, A. Ortega, Y. Su, P. Yin, and C. Gomila, “New Coding Tools for Illumination and Focus Mismatch Compensation in Multiview Video Coding,” *Trans. Circuits Syst. Video Tech.*, vol. 17, no. 11, pp. 1519–1535, Nov 2007.
- [14] C. Fehn, P. Kauff, S. Cho, H. Kwon, N. Hur, J. Kim, “Asymmetric coding of stereoscopic video for transmission over T-DMB,” *Proc. 3DTV-CON 2007*, Kos Island, Greece, May 2007.
- [15] Y. Chen, Y.-K. Wang, M. M. Hannuksela, M. Gabbouj, “Single-Loop Decoding for Multiview Video Coding,” *IEEE ICME*, Hannover Germany, June 2008.
- [16] “Common Test Conditions for Multiview Video Coding,” *JVT-T207*, Klagenfurt, Austria, Jul. 2006.
- [17] Y. Chen, Y. -K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema, M. Gabbouj, “The Emerging MVC Standard for 3D

Video Services,” *EURASIP Journal on Advances in Signal Processing*, Volume 2009, Article ID 786015, 2008.

[18] Y.-L. Lee, J.-H. Hur, Y.-K. Lee, K.-H. Han, S. Cho, N. Hur, J. Kim, J.-H. Kim, P.-L. Lai, A. Ortega, Y. Su, P. Yin, and C. Gomila, “CE11: Illumination Compensation,” *JVT-U052*, Hangzhou, China, October, 2006.

[19] H. Yang, Y. Chang, J. Huo, S. Lin, S. Gao, L. Xiong “CE1, Fine motion matching for motion skip mode in MVC,” *JVT-Z021*, Antalya, Turkey, January, 2008.

[20] P. Pandit, A. Vetro, Y. Chen, “JMVM 8 software,” *JVT-A4208*, Geneva, Apr. 2008

[21] G. Bjontegaard, “Calculation of average PSNR differences between RD-curves,” *VCEG-M33*, March, 2001.

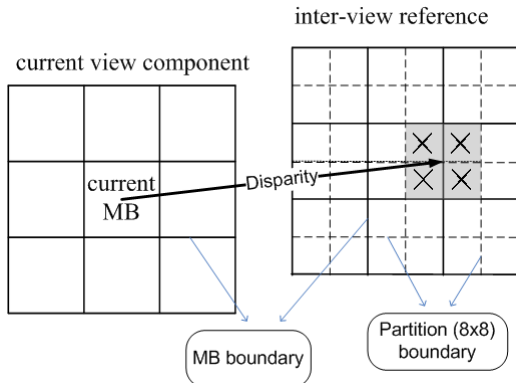


Fig. 2. Motion Vector Derivation in Motion Skip.

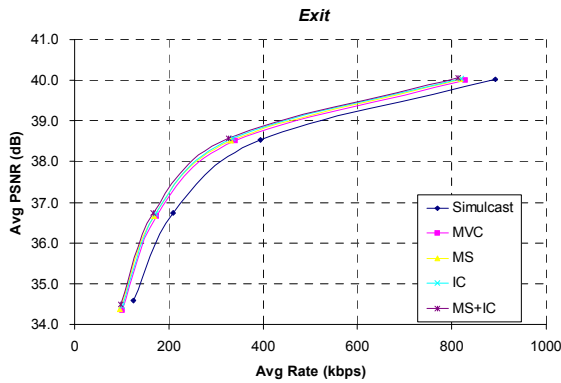


Fig. 4. RD Curves for “Exit”.

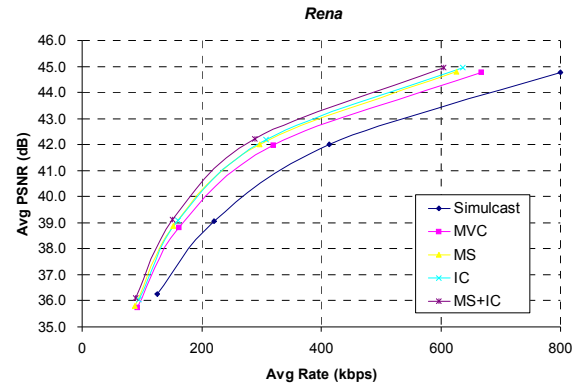


Fig. 3. RD Curves for “Rena”.

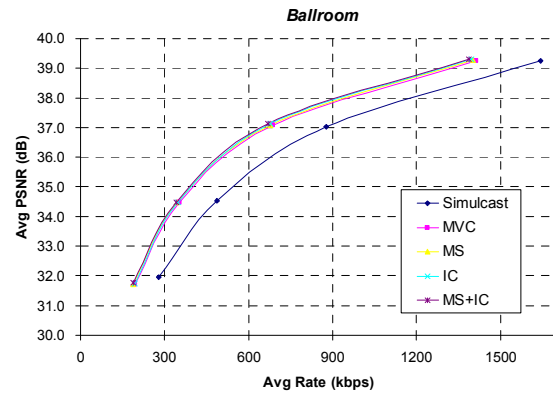


Fig. 5. RD Curves for “Ballroom”.

TABLE I. Comparison between MVC and various scenarios

Sequence	MVC vs Simulcast		MVC vs MS		MVC vs IC		MVC vs MS+IC	
	Bit-rate	Δ PSNR	Bit-rate	Δ PSNR	Bit-rate	Δ PSNR	Bit-rate	Δ PSNR
<i>Akyo&Kayo</i>	34.23%	-1.430	-5.74%	0.294	-6.17%	0.320	-11.48%	0.598
<i>Ballroom</i>	32.29%	-1.105	-1.86%	0.073	-1.90%	0.072	-3.68%	0.144
<i>Exit</i>	16.05%	-0.407	-3.13%	0.087	-2.95%	0.079	-6.12%	0.168
<i>Race1</i>	27.00%	-1.015	-4.88%	0.208	-10.67%	0.462	-14.58%	0.652
<i>Rena</i>	26.94%	-1.051	-7.17%	0.339	-7.07%	0.341	-12.89%	0.635
<i>Breakdancers</i>	12.80%	-0.279	-3.06%	0.066	-6.43%	0.141	-8.69%	0.191
<i>Flamenco2</i>	12.42%	-0.526	-4.53%	0.211	-2.89%	0.135	-7.33%	0.354
<i>Uli</i>	-0.54%	0.022	-1.26%	0.050	-1.43%	0.056	-2.88%	0.114
Average	20.15%	-0.724	-3.96%	0.166	-4.94%	0.201	-8.46%	0.357