MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Extensions of H.264/AVC for Multiview Video Compression

Emin Martinian, Alexander Behrens, Jun Xin, Anthony Vetro, Huifang Sun

TR2006-048   June 2006

Abstract

We consider multiview video compression: the problem of jointly compressing multiple views of a scene recorded by different cameras. To take advantage of the correlation between views, we propose using disparity compensated view prediction and view synthesis and describe how these features can be implemented by extending the H.264/AVC compression standard. Finally, we discuss experimental results on the test sequences from the MPEG Call for Proposals on multiview video.

International Conference on Image Processing

Publication History:–

1. First printing, TR2006-048, May 2006

EXTENSIONS OF H.264/AVC FOR MULTIVIEW VIDEO COMPRESSION

Emin Martinian, Alexander Behrens, Jun Xin, Anthony Vetro, Huifang Sun

Mitsubishi Electric Research Labs
201 Broadway, Cambridge, MA 02139

ABSTRACT

We consider multiview video compression: the problem of jointly compressing multiple views of a scene recorded by different cameras. To take advantage of the correlation between views, we propose using disparity compensated view prediction and view synthesis and describe how these features can be implemented by extending the H.264/AVC compression standard. Finally, we discuss experimental results on the test sequences from the MPEG Call for Proposals on multiview video.

1. INTRODUCTION

Advances in display and camera technology make recording a single scene with multiple video signals attractive. While there are many applications of such multiview video sequences including free viewpoint video [1], three dimensional displays [2], [3], and high performance imaging [4], the dramatic increase in the bandwidth of such data makes compression especially important. Consequently, there is increasing interest in exploiting the inherent correlation in multiview video through disparity compensated prediction [5], mesh-based view prediction [6], wavelet transforms, and related techniques. In response to recent advances in coding technology and the emerging applications for multiview video, MPEG has recently issued a Call for Proposals on multiview video coding [7].

We describe a novel extension of the H.264/AVC standard for multiview video compression. The proposed system achieves gains of up to 2 dB in PSNR over independent coding of all views. In addition to existing temporal prediction and well known disparity compensated prediction, our system adds a novel view synthesis prediction technique. To maintain compatibility with the existing standards and enable reuse of macroblock layer syntax for coding multiple views, we also describe a multiview reference picture management scheme.

The rest of this paper is organized as follows. Disparity compensated view prediction and view synthesis prediction are presented in Section 2. In section 3, we describe how these prediction tools can be incorporated into the existing H.264/AVC compression standard with the proposed multiview reference picture management scheme. Random access for multiview video is discussed in section 4. We present experimental results in Section 5, and close with some concluding remarks in Section 6.

2. PREDICTION TOOLS

This section describes two prediction tools: disparity compensated view prediction as well as view synthesis prediction.

2.1. Disparity Compensated View Prediction

In the following we describe the disparity compensated view prediction (DCVP) method that is used in our system. We define I[c,t,x,y] as the intensity of the pixel in camera c at time t at pixel coordinates (x,y). With conventional temporal prediction for each camera c, frame t in sequence c is typically predicted only from other frames in sequence c. With DCVP, for each c, the value of I[c,t,x,y] may also be predicted from I[$c',t,x\text{-}m_x,y\text{-}m_y$] where ($m_x$,$m_y$) is a disparity vector computed in a blockwise manner and c' is a frame from an already encoded sequence from another camera. One natural camera prediction structure is the sequential structure where I[c,t,x,y] is predicted from I[$c\text{-}1,t,x,y$], which is analogous to the IPPP Group of Pictures (GOP) structure in conventional temporal coding. Other camera prediction structures are also possible and may be better depending on the camera geometry.

2.2. View Synthesis Prediction

While DCVP provides improvements over pure temporal prediction, it does not take advantage of some essential features of multiview video. First, while temporal motion can be accurately modeled using translational motion compensation, the differences between multiple views of a scene usually cannot. For example, in moving from one camera to another the disparity in the screen pixel coordinates of an object between cameras will depend on the depth of the object. Objects closer to the camera will move much more than objects that are far from the camera. Also, effects such as rotations, zooms, or different intrinsic camera properties are often difficult to model using pure translational motion compensation. Finally, since many applications of multiview video such as 3D displays or free viewpoint video require accurate camera parameters, this information is often available at encoding time and should ideally be used to improve compression.

As illustrated in Figure 1, we exploit these features of multiview video by synthesizing a virtual view from previously encoded views and then performing predictive coding using the synthesized views. Specifically, for each c, we first synthesize a virtual frame I'[c,t,x,y] based on the on the unstructured lumigraph rendering technique of Buehler *et al.* [8] (described in more detail shortly) and then use disparity compensated view prediction as described in Section 2.1 to predicatively encode the current sequence using the synthesized view.

To synthesize I'[c,t,x,y], we require a depth map D[c,t,x,y] that describes how far the object corresponding to pixel (x,y) is from camera c at time t, as well as an intrinsic matrix $A(c)$, rotation matrix $R(c)$, and a translation vector $T(c)$ describing the location of camera c relative to some global coordinate system. Using these

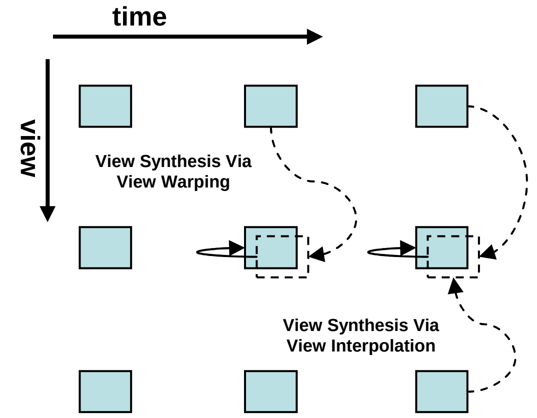

Figure 1 Prediction using view synthesis.

quantities, we can apply the well-known pinhole camera model to project the pixel location *(x,y)* into world coordinates [*u,v,w*] via

$$[u,v,w] = R(c) \cdot A^{-1}(c) \cdot [x,y,1] \cdot \mathrm{D}[c,t,x,y] + T(c) \qquad (1)$$

Next, the world coordinates are mapped into the target coordinates [*x',y',z'*] of the frame in camera *c'* which we wish to predict from via

$$[x',y',z'] = A(c') \cdot R^{-1}(c) \cdot \{[u,v,w] - T(c')\}. \qquad (2)$$

Finally, to obtain a pixel location, the target coordinates are converted to homogenous form [*x'/z',y'/z',1*] and the intensity for pixel location *(x,y)* in the synthesized frame is I'[*c,t,x*,y]=I[*c',t,x'/z',y'/z'*]. Finally, we note that while A(c), R(c), and T(c) must be communicated from the encoder to the decoder, the amount of information required to describe these parameters is very small and thus the associated coding overhead is negligible.

An important issue in view synthesis is computing, coding, and transmitting accurate depth maps. In many scenarios such as free viewpoint video and 3D displays, such depths maps may be required as part of the application itself and can therefore be used in the compression process without requiring any extra coding overhead or computational effort. In general, however, one must both obtain the required depth maps and define a method for the encoder to convey them to the decoder.

For our tests, we used two methods of obtaining and encoding the depth maps. First, some sequences (i.e., the breakdancers test sequences from Microsoft Research [9]) provide depth maps that were extracted using computer vision techniques. For such sequences, we simply use H.264/AVC to compress the depth map. Based on ad hoc testing, we found that devoting 5-10% of the total bit rate to encoding the depth map produced acceptable results.

For sequences without depth maps, we used a block based depth search algorithm to extract the optimal depth. Specifically, we define minimum, maximum, and incremental depth values D_{min}, D_{max}, D_{step}, and a block size D_{block}. Then, for each block of *B* pixels in the frame that we wish to predict, we choose the depth to minimize the error for the synthesized block:

$$\mathrm{D}(c,t,x,y) = \operatorname{argmin} \| \mathrm{I}[c,t,x,y] - \mathrm{I}[c',t,x'/z',y'/z'] \| \qquad (3)$$

where the minimization is carried out over the set d = {D_{min}, $D_{min} + D_{step}$, $D_{min}+2D_{step}$, …, D_{max}} and || I [*c,t,x,y*] – I [*c',t,x'/z',y'/z'*] || denotes the average error between the block of size D_{block} centered at (*x,y*) in camera *c* at time *t* and the corresponding block that we are predicting from. Note, that the depth influences the error by affecting the coordinates *(x',y')* of the block we are predicting from.

Figure 2 presents a visual comparison of the two kinds of depth maps for the breakdancers sequence. In general, the depth as computed in (3) yields a smaller error in the synthesized view (and hence a higher PSNR after compression) than depth obtained from classic methods of computer vision, but the depth from (3) is also harder to compress. We believe this is because most depth from stereo algorithms proposed by computer vision researchers incorporate regularization constraints to produce smooth depth maps, while (3) does not include any explicit smoothing and is specifically aimed at minimizing prediction error.

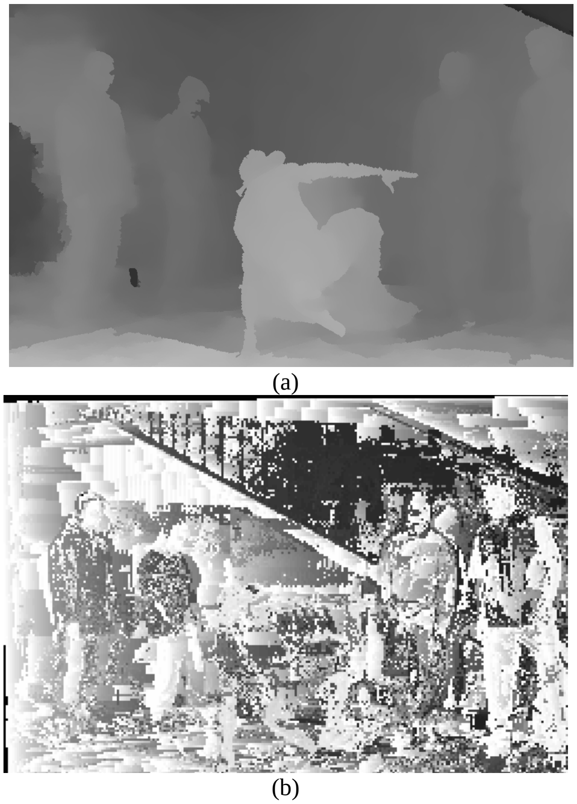

Figure 2 Comparison of depth maps. (a) Depth maps obtained from computer vision algorithms (courtesy of Microsoft Research [9]); (b) Depth maps obtained from block based depth search as defined by equation (3) and using 4x4 block size.

Ideally, one should modify existing depth from stereo algorithms with the goal of producing high quality view synthesis and multiview compression while also making the depth map easy to compress. Due to time constraints, however, we did not implement a depth extraction algorithm, which produced smooth enough depth maps to enable efficient coding of the depth map itself. Instead, for sequences where we used view synthesis and depth maps were unavailable, we compute depth according to (3) as a proxy and code the sequences at 5% below the target bit rate.

Thus assuming that it is possible to produce depth maps that can be compressed at this rate and provide good view synthesis (an assumption that is validated for the breakdancers sequence), we believe our results provide an accurate estimate of the performance of view synthesis.

Also, we note that not all of the macro-blocks are coded using VSP. In particular, for some macroblocks temporal prediction is best, while for others DCVP is best, and sometimes intra macro-blocks are best. Consequently, a more efficient implementation of VSP would only encode the depth for macro-blocks that use view synthesis prediction. In ad hoc tests, we found that VSP was used in at about 10% of macro-blocks and so the overhead for depth maps could be reduced even further than reported above by sending only a partial depth map from the encoder to the decoder. Of course, some additional syntax may need to be defined for a decoder to properly interpret a partial depth map.

3. MULTIVIEW REFERENCE PICTURE MANAGEMENT

Essentially, VSP can be considered as a special case of DCVP. Specifically, DCVP involves coding the video sequence from camera *c* using predictive coding from another video sequence from camera *c'*. In VSP, we simply synthesize a virtual camera sequence and apply DCVP to predict from the synthesized sequence. Consequently, both VSP and DCVP require an encoder and decoder that can use reference frames outside the current sequence being compressed. While this is conceptually straightforward, some care is required to achieve an efficient implementation of this feature.

As illustrated in Figure 3, we implement DCVP by modifying the H.264/AVC reference software (version JM 9.5) [10]. One of the main advantages of this approach is that we can reuse most of the existing bitstream syntax. Specifically, H.264/AVC defines a Decoded Picture Buffer (DPB) where previously coded frames for the current sequence are stored so they may be used as references for predictive coding. To allow prediction from other cameras, we use a sequence level configuration to define a convention for inserting and deleting previously coded frames from other cameras into the DPB.

Before encoding begins, the user specifies a list of multiview reference sequences. Then for each *t*, before frame *t* is processed, the encoder and decoder read in frame *t* from each multiview reference sequence and place it into a multiview reference picture list. Then, the contents of this list are inserted into the DPB, the usual H.264/AVC coding loop is entered, and after the frame has been processed, each picture in the multiview reference picture list is removed from the DPB. Since both the encoder and decoder modify the DPB in the same way, they remain synchronized and whenever the encoder uses multiview references for predicatively encoding other frames, this information is signaled to the decoder using the existing H.264/AVC syntax. Thus, our use of multiview references frames uses exactly the same motion search, mode decision, entropy coding, etc., of the underlying H.264/AVC compression engine.

The multiview reference picture list is required in this process for a number of reasons. First, since the encoding and decoding processes modify the DPB, the multiview reference picture list provides a way to tell which pictures are multiview references that should be removed. Second, the entropy coding engine in JM 9.5 requires more bits when using references towards the end of the DPB. As a result, we also use the multiview reference picture list manager to reorder the multiview references so that they are ordered by increasing correlation (specified by the user or determined automatically) and come after the temporal reference pictures in the DPB. In ad hoc testing, we observed that properly ordering the reference pictures in the DPB could improve performance by 0.25-0.5 dB in some cases.

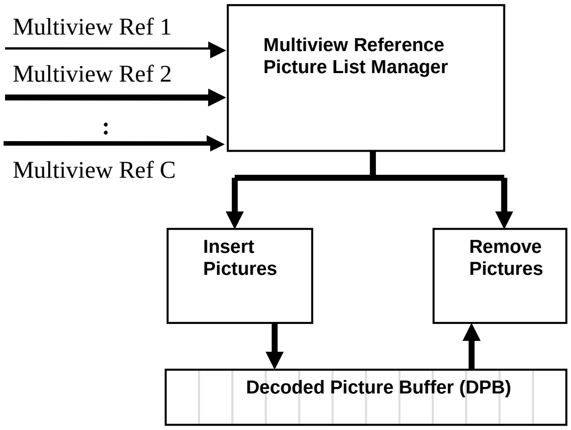

Figure 3 Reference picture management for multiview coding.

4. RANDOM ACCESS

In order to provide temporal random access to any point in a video sequence, I-frames are usually spaced throughout the sequence at regular intervals, because I-frames can be decoded independently of other frames. In multiview coding, however, it is possible to obtain temporal random access using a new type of frame, which we call a "V-frame" or "V Picture". Specifically, a V-frame is like an I-frame in the sense that it is encoded without any temporal prediction but it differs from an I-frame in that it allows for prediction from other cameras. By placing a V frame at periodic intervals (e.g., every second or half-second) it is possible to obtain the same temporal random as with I-frames, but achieve better coding efficiency since the V-frames can use DCVP or VSP.

5. RESULTS

The rate-distortion performance results for a subset of the test sequences in the MPEG Call for Proposals on multiview video coding [7] are shown in Figure 4. For the Breakdancers sequence, we used view synthesis prediction with the depth maps provided by Microsoft Research compressed at the encoder using H.264/AVC at rates of 30, 50, and 50 kb/s. For the Ballroom, Flamenco2, and Rena sequences, we used view synthesis prediction with our own block based depth maps as a proxy since high quality depth maps were unavailable. Specifically, we added a 5% overhead to the bit rate for each of these sequences to account for the rate that would have been required to compress a depth map if it had been available.

From the plot, we see that view synthesis prediction provides gains over independent coding for each of these sequences. The gains range from about 0.2 dB for the highest bit rate of the Rena sequence to almost 2 dB for the lowest rate of the Ballroom sequence. Evidently, view synthesis can be a useful tool in multiview video compression.

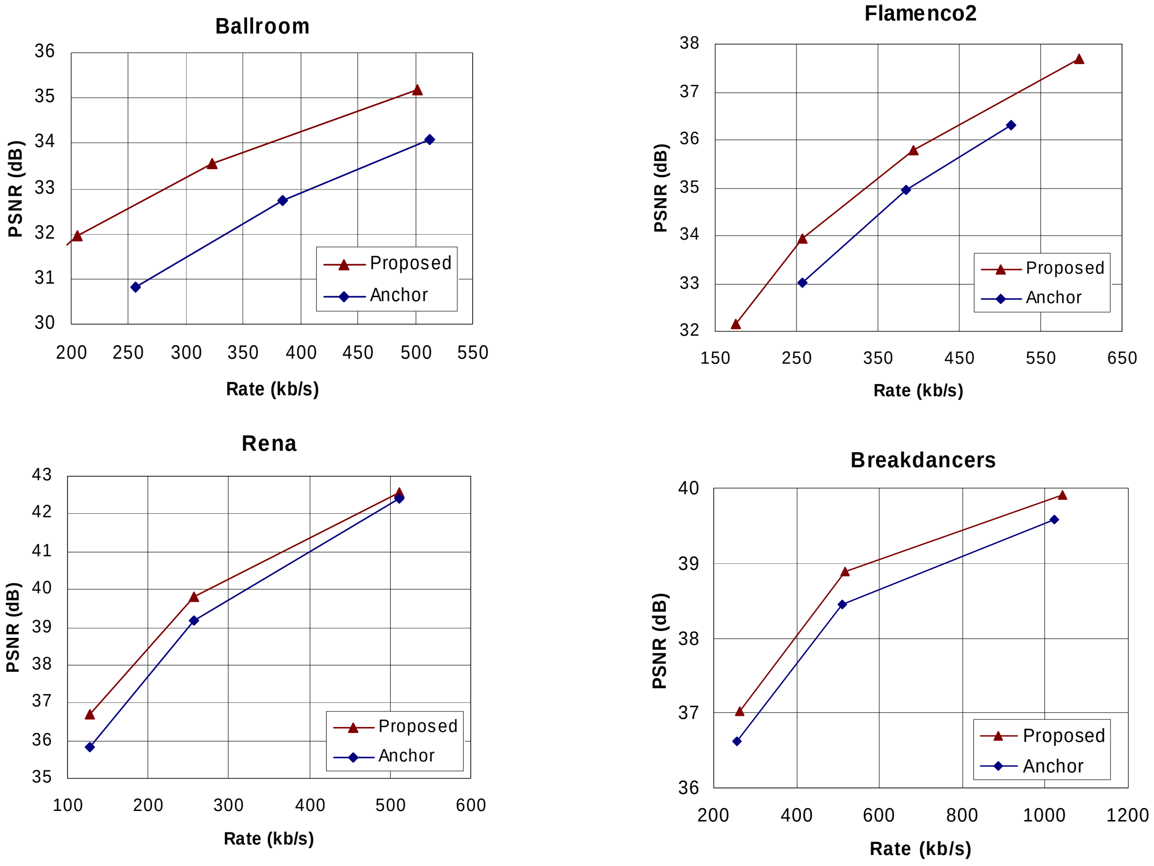

Figure 4 Performance results for several MPEG test sequences.

6. CONCLUDING REMARKS

We proposed a new multiview video compression system (which has also been submitted in response to MPEG Call for Proposal on Multiview Video Coding) and showed that it achieves gains of up to 2 dB in PSNR over independent coding of all views. Our main contributions include a novel view synthesis prediction technique, a buffer management method that extends the existing H.264/AVC compression standard to allow disparity compensated view prediction and view synthesis prediction, and the new V-frame picture type.

There are a variety of opportunities for future work. First, high quality, compressible depth maps are essential for view synthesis. Classical methods of computing depth maps do not take advantage of the fact that a multiview encoder always has a real version of the view to synthesize and can use this ground truth to produce more accurate depth maps. Thus, new depth extraction algorithms could potentially yield significantly better performance. We are currently developing such depth extraction algorithms as well as associated compression techniques to efficiently communicate depth. Second, we have observed that adding a “synthesis correction vector” to the view synthesis process can compensate for inaccuracies in camera parameters resulting in better view synthesis [11]. Finally, the proposed multiview compression system can be further improved by taking tools that provide gains in single view video compression, e.g., Open-GOP and hierarchical B-frames, and adapting them to exploit the properties of multiview video.

REFERENCES

[1] A. Smolic, P. Kauff, “Interactive 3-D video representation and coding technologies”, *Proceedings of the IEEE*, vol. 93, no. 1, pp 98-110, Jan. 2005.

[2] N. A. Dodgson, “Autostereoscopic 3D Displays”, *IEEE Computer*, vol. 38, no. 8, pp. 31-36, Aug. 2005.

[3] M. Tanimoto, “FTV (Free Viewpoint Television) Creating Ray-Based Image Engineering”, *International Conference on Image Processing*, vol. 2, pp. 25-28, Sept. 2005.

[4] B. Wilburn, et al., “High Performance Imaging Using Large Camera Arrays,” *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 765-776, July 2005.

[5] B. Girod, Chuo-Ling Chang, P. Ramanathan, Xiaoqing Zhu, “Light field compression using disparity-compensated lifting”, *Proc. International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 760-763, April 2003.

[6] R.-S. Wang and Y. Wang, “Multiview video sequence analysis, compression, and virtual viewpoint synthesis”, *IEEE Transactions on Circuits and Systems for Video Technolgy*, vol. 10, no. 3, pp. 397-410, April 2000.

[7] ISO/IEC JTC1/SC29/WG11, “Updated Call for Proposals on Multi-view Video Coding”, MPEG Doc. N7567, Nice, France, Oct. 2005.

[8] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, “Unstructured Lumigraph Rendering”, *Proceedings of ACM SIGGRAPH*, pp. 425-432, Aug. 2001.

[9] C.L. Zitnick, et al., "High-quality video view interpolation using a layered representation," *ACM SIGGRAPH and ACM Trans. on Graphics*, Los Angeles, CA, Aug. 2004, pp. 600-608.

[10] H.264/AVC Reference Software JM 9.5, available online at http://iphome.hhi.de/suehring/tml/doc/lenc/html/index.html

[11] E. Martinian, A. Behrens, J. Xin, A. Vetro, “View Synthesis for Multiview Video Compression”, *Picture Coding Symposium* 2006.