H.264-Based Depth Map Sequence Coding Using Motion Information of Corresponding Texture Video

Han Oh and Yo-Sung Ho

Gwangju Institute of Science and Technology (GIST)
1 Oryong-dong, Buk-gu, Gwangju, 500-712, Korea
{ohhan, hoyo}@gist.ac.kr

Abstract. Three-dimensional television systems using depth-image-based rendering techniques are attractive in recent years. In those systems, a monoscopic two-dimensional texture video and its associated depth map sequence are transmitted. In order to utilize transmission bandwidth and storage space efficiently, the depth map sequence should be compressed as well as the texture video. Among previous works for depth map sequence coding, H.264 has shown the best performance; however, it has some disadvantages of requiring long encoding time and high encoder cost. In this paper, we propose a new coding structure for depth map coding with H.264 so as to reduce encoding time significantly while maintaining high compression efficiency. Instead of estimating motion vectors directly in the depth map, we generate candidate motion modes by exploiting motion information of the corresponding texture video. Experimental results show that the proposed algorithm reduces the complexity to 60% of the previous scheme that encodes two sequences separately and coding performance is also improved up to 1dB at low bit rates.

Keywords: Three-dimensional television, depth-image-based rendering, depth map sequence coding, H.264/AVC video coding standard.

1 Introduction

We have a highly advanced visual sense that can perceive stereoscopic effects and the depth of an object. Three-dimensional television (3D-TV) is one of the promising next-generation multimedia appliances which exploits this visual sense and provides viewers with more realistic and immersive impression. In recent years, considerable research projects have been conducted on 3D-TV. Among various research efforts, a noticeable research project for 3D-TV is the Advanced Three-dimensional Television System Technologies (ATTEST) project [1] that started in March 2002. In ATTEST, a two-dimensional video sequence and its associated per-pixel depth information are recorded and transmitted, instead of transmitting two monoscopic video sequences for the left and right eye viewpoints. Stereoscopic videos are then generated using a depth-image-based rendering (DIBR) technique, as shown in Fig. 1. This data format has some advantages of high coding efficiency and interactivity as well as supporting scalability for various types of receivers.

L.-W. Chang, W.-N. Lie, and R. Chiang (Eds.): PSIVT 2006, LNCS 4319, pp. 898 – 907, 2006.

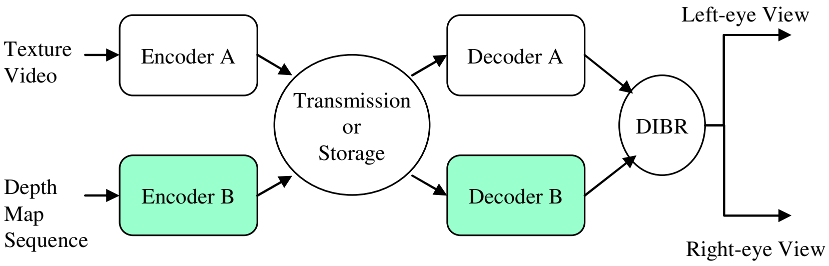


Fig. 1. Structure of 3D-TV for DIBR

In order to utilize limited bandwidth or storage space efficiently, the texture video can be compressed by exploiting its temporal and spatial correlations. In the same manner, the depth map sequence also has a significant amount of redundancies that can be removed for compression. Depth map coding schemes can be categorized into two classes. One class of depth map coding contains coding schemes using adaptive 3D mesh-based interpolation and node tracking. The other class includes coding schemes using conventional video standards, such as MPEG-2, MPEG-4, and H.264.

Mesh-based approaches [2, 3] are based on non-uniform image sampling. While the number of sampling points is reduced in flat areas, more points are coded in high curvature areas. The position and the number of points are determined by an iterative scheme based on the mean-squared-error (MSE) criterion. In order to reduce temporal redundancy, they handle inter frames by moving some nodes from frame to frame and coding the corresponding vectors. Thus, only those changes between two frames are coded. This approach is programmed with the OpenGL library and handled by a graphics hardware; therefore, its rendering time is very fast.

Another approach for depth map coding employs the conventional video coding standards [4]. 8-bit quantized depth values are mapped to a YUV color signal where UV values are set to 128, and compressed as the texture video. This scheme is easy to adapt and very efficient. Up to now, H.264 provides superior compression results compared to the mesh-based approach in terms of PSNR values [5].

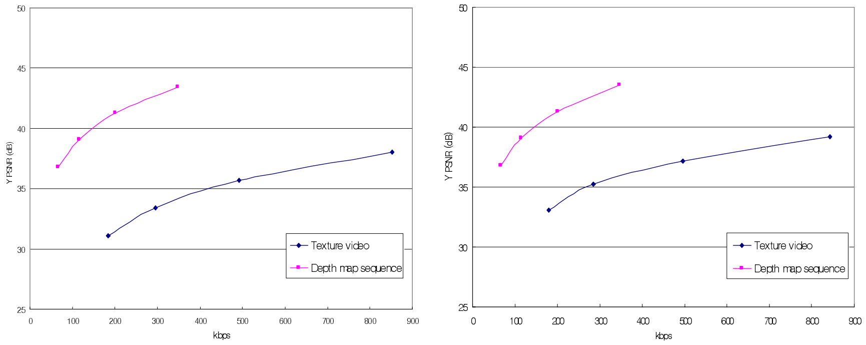
Although H.264 supports various block sizes and rate-distortion optimization and it shows high compression efficiency in depth map coding, it has some disadvantages of requiring long encoding time and high encoder complexity. This drawback is mainly due to the motion estimation operation that is performed for both sequences.

In this paper, we propose an efficient algorithm for depth map coding by sharing the motion information with the corresponding texture video. Since motion estimation in the depth map sequence is skipped and motion information is taken from the texture video, the entire encoding cost is significantly reduced. The idea of sharing motion information can be found in Stefan's paper [6]; however, it is based on MPEG-2 and motion information of the corresponding texture video is merely copied without any additional modifications. Therefore, this approach does not work well for H.264 since the motion information of the depth map sequence is not optimized as in the corresponding texture video. In this paper, we propose a new idea for sharing motion information between two sequences in H.264.

This paper is organized as follows. Section 2 briefly explains the characteristics of depth map sequence coding in H.264. Section 3 analyzes the similarity of motion information between the depth map sequence and the corresponding texture video. After we describe details of the proposed algorithm in Section 4, experimental results are presented in Section 5. Finally, Section 6 concludes this paper.

2 Characteristics of Depth Map Sequence Coding in H.264

While the texture image indicates intensity values of each color component, the depth map represents depth information per pixel. Pixels in one object tend to have similar values in both the texture image and the depth map. Since the depth map is simpler than the corresponding texture image, we can have higher coding efficiency in depth map coding, as shown in Fig. 2.



(a) INTERVIEW (b) ORBI

Fig. 2. Comparison of coding efficiency

In depth map coding with H.264, there are much more skipped macroblocks and 16x16 modes than the case of the texture video. These modes form approximately 95% when the quantization parameter (QP) is 40. This is mainly due to simplicity of the depth map. On the other hand, sub-macroblock modes, such as 8x8, 4x8, 8x4 and 4x4, rarely occur in depth map coding. Coding with sub-macroblock modes increases the bit rate due to numerous motion vectors. In depth map coding, a large number of macroblocks in the inter frame are encoded in 16x16 or 4x4 intra modes. The intra mode is a prediction scheme which exploits the neighboring pixels around the block to be coded. The intra mode works well for smooth images like the depth map and saves the coding bits for motion vectors.

In the inter frame, bits are composed of seven components: header, mode, motion information, coded block pattern (CBP), residual data, delta QP, and stuffing bits. Residual data in the depth map usually takes a relatively small portion, which means that it is predicted well with motion estimation or intra prediction. In addition, bits for motion information take a large portion (more than 40%) even though most macroblocks are encoded in large sizes of blocks and intra coding.

3 Similarity Analysis for Motion Information

In general, the texture video and the depth map sequence have similar characteristics. For example, boundaries of objects coincide and directions of object movements are the same in both sequences. During the motion estimation operation in the texture video, the motion vector is estimated by

$$J_{motion}(MV, REF \mid \lambda_{motion}) = SAD(MV, REF) + MCOST(MV, REF) \tag{1}$$

where *MV* is the motion vector and *REF* is the reference frame for motion estimation. The motion cost (*MCOST*), which indicates the number of coding bits for the motion vector, is defined by

$$MCOST = \lambda_{motion} \frac{(\text{mvbits}[4c_x - p_x] + \text{mvbits}[4c_y - p_y])}{2^{16}} \tag{2}$$

where λ_{motion} is the Lagrangian multiplier of the motion vector and it is determined based on the quantization parameter (QP) by $\sqrt{0.85 \times 2^{(QP-12)/3}}$. In Eq. (2), (c_x, c_y) is the position of the actual candidate motion vector and (p_x, p_y) is the position of the predicted motion vector obtained by the left, upper and upper-right blocks. mvbits[·] is an array which contains expected bits for encoding of the motion vector, as shown in Eq. (3). The farther (c_x, c_y) is away from (p_x, p_y), the more the estimated cost for the motion vector increases.

$$\text{mvbits}[-i] = [i] = 2i + 1 \quad i = 0, 1, 2, \cdots, 3 + 2 \times \lceil \log_2(\#\text{ of positions} + 1) \rceil \tag{3}$$

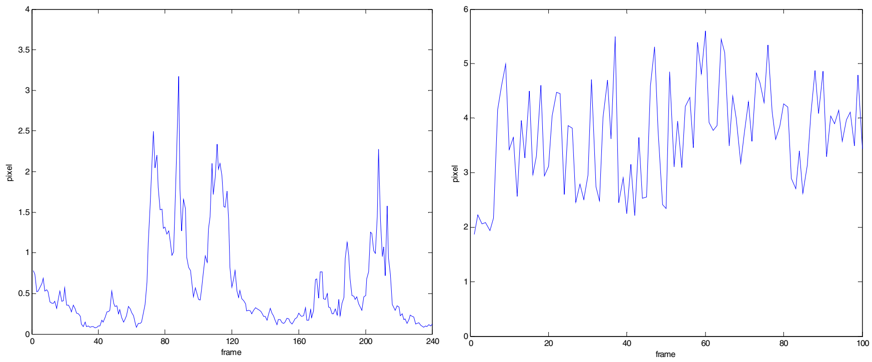
Therefore, the motion cost is carried out as a smoothness constraint. This means that motion vectors are not random and have similar values as neighboring motion vectors even when the size of the block is 4x4. Because of this property of motion vectors, the structure of objects in the depth map is maintained and some blocks may have incorrect motion compensation.

Comparing the similarity of motion vectors between the texture video and the depth map sequence, we define a difference measure by the average distance of two motion vectors in the frame:

$$\text{dist}_{\text{frame}} = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left\| \mathbf{t}_{i,j} - \mathbf{d}_{i,j} \right\| \tag{4}$$

where **t** is the motion vector for each 4×4 block of the texture video and **d** is that of the depth map sequence.

Figure 3 shows the difference of motion vectors per frame of two test sequences, INTERVIEW and ORBI. The maximum search range is ±32. While the average difference of the INTERVIEW sequence with a small motion is about 0.59 pixels, the ORBI sequence whose motion is relatively large because of camera motion has about 3.77 pixels.



(a) INTERVIEW (b) ORBI

Fig. 3. Average difference of motion information per frame

4 Motion Information Sharing Algorithm

When we encode the depth map sequence, we can share the motion information of the corresponding texture video by exploiting the similarity of motion vectors between two sequences. The proposed algorithm consists of three stages, as illustrated in Fig. 4. Initially, we need to decode coding modes and associated motion vectors of the corresponding texture video to use the motion information for depth map sequence coding. From the decoded motion information, we generate various modes and associated motion vectors. Then, we select an optimal mode among the generated candidate modes based on the rate-distortion theory.

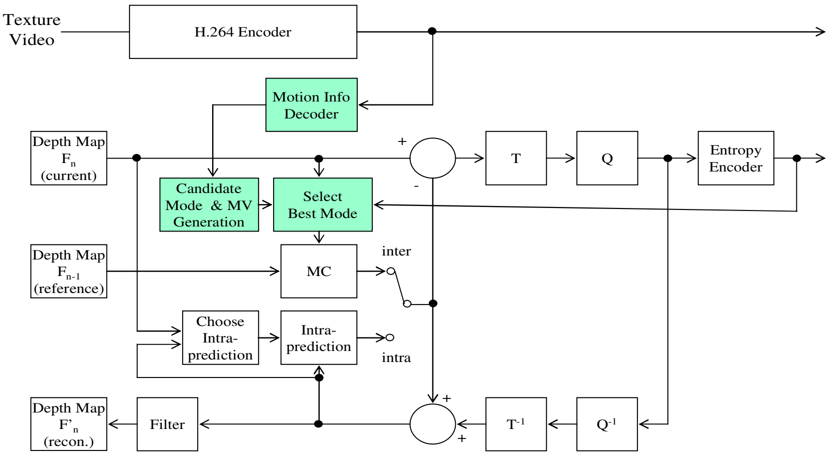


Fig. 4. Block diagram of the proposed system

4.1 Decoding of Motion Information

Inter prediction supports a range of block sizes from 16×16 down to 4×4. In addition to the inter prediction, intra prediction is also available in the inter frame and has two modes: 16×16 mode and 4×4 mode. If prediction with neighboring pixels provides better performance than prediction with motion vectors, the intra mode is selected. However, intra coding rarely occurs in the texture video. For most QPs, the frequency of the intra mode is about 1.0% or less. Since intra prediction does not have motion information, direct use of motion vectors in depth map sequence coding is impossible. Instead, the motion vector can be generated from motion vectors of the neighboring blocks. In our experiment, the median of motion vectors for left(**A**), upper(**B**), and upper right(**C**) blocks has shown a good performance.

$$(p_x, p_y) = MEDIAN(\mathbf{A}, \mathbf{B}, \mathbf{C}) \tag{5}$$

4.2 Candidate Mode Generation

Since modes and motion vectors of the texture video were optimized for the texture video, they should be adjusted to fit into the depth map sequence coding. For this purpose, we need to generate various candidate modes and motion vectors. Generation of candidate modes can be divided into two operations: merge and split operations.

Figure 5 shows an example of generating candidate modes from the decoded mode. If the mode of the texture video consists of smaller partitions, a merge operation is performed and generates larger sizes of modes. If the mode of the texture video is larger than the size of the current mode to be generated, a split operation is performed and generates several smaller sizes of modes using the neighboring blocks.

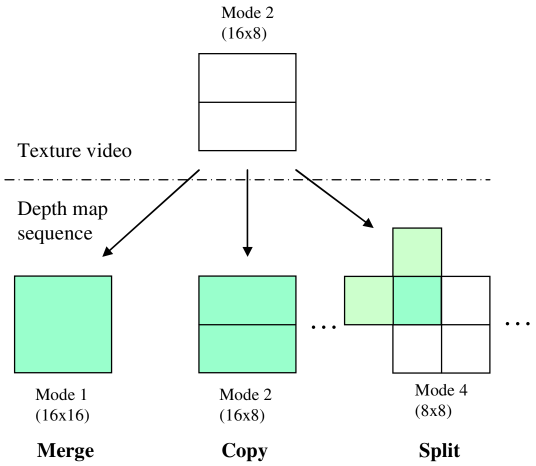


Fig. 5. Example of candidate mode generation

The motion vector for each mode is obtained by

Copy : If the mode is the same as that of the texture video, motion vectors are simply copied from the texture video.

Merge : The average of the motion vectors within the size of the current mode.

Split : The average of the motion vectors within the size of the current mode and those of left and upper blocks to avoid generating duplicate motion vectors.

4.3 Rate-Distortion Optimization

The best mode is selected from the newly generated inter modes, SKIP mode, and intra modes based on the following rate-distortion cost.

$$\mathbf{I}^* = \arg\min_{\mathbf{I}} D(\mathbf{S},\mathbf{I} \mid \lambda) + \lambda \cdot R(\mathbf{S},\mathbf{I} \mid \lambda) \tag{6}$$

where **S** is the set of blocks to be coded, D is the distortion, R is the bit rate, **I** and **I*** are encoding parameters and the best encoding parameters, respectively. In Eq. (6), λ is the Lagrangian multiplier of the mode. R includes the bits of the header, CBP, mode, motion information, and residual data. By skipping the bits for motion information, the best encoding parameters are selected with more weighting on distortion. Thus, some increase in PSNR values is expected. Moreover, when motion compensation is performed with wrong motion vectors, error propagation can be blocked due to the intra mode in the inter frame.

However, the proposed algorithm does not always show better performance at all bitrates. Sharing of motion vectors may be suboptimal in terms of the prediction error criterion. Therefore, residual data can be increased compared to direct coding of the depth map separately. Our approach is advantageous only when coding bits of the residual data are less than bit savings obtained by no coding of motion vectors. In other words, coding performance is good at low bitrates where the residual data is roughly quantized. This scheme is effective even when the depth map image, which is considerably degraded at low bit rates, is used in synthesizing views [7].

5 Experimental Results and Analysis

We have implemented the proposed algorithm into JM reference software 9.7 and our simulation conditions are summarized in Table 1. Two types of test sequences have been used. The depth map sequences of ORBI and INTERVIEW [8] were captured by an infrared range camera, so-called Zcam™ [9]. On the other hand, the depth map sequences of BREAKDANCERS and BALLET [10] were calculated by a state-of-art stereo matching algorithm from multiple scenes. In our experiment, we have compared the proposed scheme to the original coding scheme where the depth map sequence is coded separately. Motion vectors have been taken from various QPs (28, 32, 36, 40) in the corresponding texture video. Since the depth map sequence for 3D-TV generally shows high quality at low bitrates, we have evaluated our proposed algorithm in such high QPs (more than 40).

Figure 6 shows the encoding times of 100 frames of the texture video and the depth map sequence, respectively. Since the proposed algorithm omits the motion estimation operation that has high computational complexity, the encoding time of the proposed algorithm was about 60% of the original scheme.

Figure 7 shows R-D curves of the original scheme and the proposed scheme. At low bitrates, coding efficiency has been improved up to 1dB, which shows the similar tendency on both types of test sequences. In particular, improvement of coding efficiency is easily observed in sequences having a large motion, because sequences with a large motion save a lot of bits by not sending motion information of the depth map sequence. In addition, motion vectors obtained from the texture video that is encoded in higher quality have shown better results. However, at high bitrates, coding performance has significantly fallen due to the precise quantization of increased residual data.

Table 1. Simulation conditions

Number of Frames	100
Search Range	±32
Number of Reference Frames	1
Sequence Type	IPPP
Entropy Coding Method	CABAC
RD Optimization	High Complexity Mode
I Slice Insertion	0.5 sec

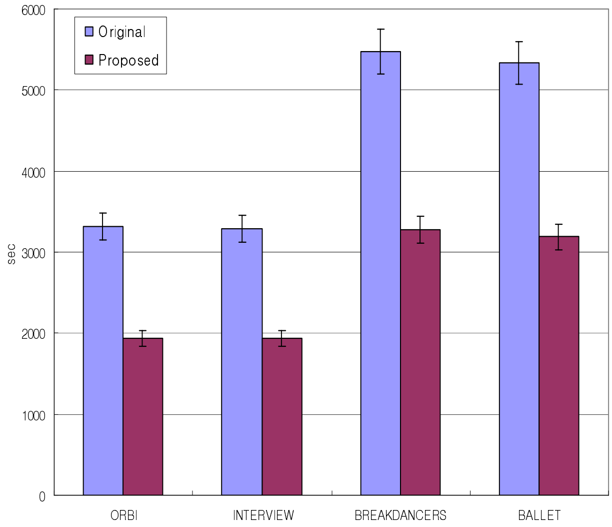
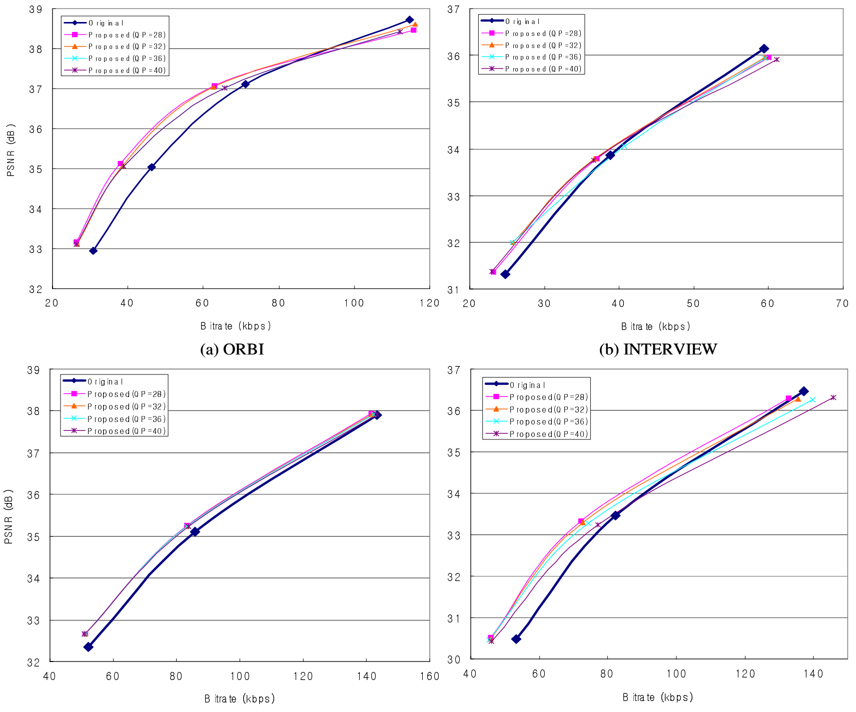


Fig. 6. Comparison of encoding times



(a) ORBI

(b) INTERVIEW

(c) BREAKDANCERS

(d) BALLET

Fig. 7. Performance comparison

6 Conclusions

In this paper, we have proposed a new H.264-based coding algorithm for the depth map sequence using motion information of the corresponding texture video. Although pixel values in both sequences are different, boundaries of objects in the scene coincide and directions of object movements are very similar. Besides, when estimating the motion vectors in the texture video, H.264 considers the cost for coding motion vectors. Hence, the structure of objects tends to be maintained. These features allow the motion vectors of two sequences to be similar. In order to share motion vectors in a proper way, we have generated various candidate modes and motion vectors from the decoded modes and motion vectors of the texture video. We then select one among those candidates based on the rate-distortion optimization. Our experimental results have demonstrated that the proposed scheme reduces the complexity up to 60% on average of the original scheme where the depth map sequence and the texture video are encoded separately. Coding efficiency has been improved up to 1dB at low bitrates. However, the proposed scheme does not always provide improved performance at high bit rates. Sometimes at high bitrates, coding performance has been rather reduced due to precise coding of increased residual data. Therefore, the proposed scheme is effective when fast encoding is required at low bit rates.

Acknowledgements

This work was supported in part by the Information Technology Research Center (ITRC) through the Realistic Broadcasting Research Center (RBRC) at Gwangju Institute of Science and Technology (GIST), and in part by the Ministry of Education (MOE) through the Brain Korea 21 (BK21) project.

References

[1] Redert, A., Op de Beeck, M., Fehn, C., IJsselsteijn, W., Pollefeys, M., Van Gool, L., Ofek, E., Sexton, I., Surman, P.: ATTEST–Advanced Three-Dimensional Television System Technologies. Proc. of International Symposium on 3D Data Processing (2002) 313–319

[2] Chai, B., Sehuraman, S., Hatrack, P.: Mesh-based Depth Map Compression and Transmission for Real-time View-based Rendering. Proc. of International Conference on Image Processing (2001)

[3] Grewatsch, S., Muller, E.: Fast Mesh-based Coding of Depth Map Sequences for Efficient 3D-Video Reproduction Using OpenGL. Visualization, Imaging, and Image Processing (2005)

[4] Fehn, C.: Depth-image-based Rendering (DIBR), Compression and Transmission for a New Approach on 3D TV. Proc. of SPIE Conf. Stereoscopic Displays and Virtual Reality Systems XI Vol. 5291 (2004) 93–104

[5] Grewatsch, S., Muller, E.: Evaluation of Motion Compensation and Coding Strategies for Compression of Depth Map Sequences. 49th SPIE's Annual Meeting (2004)

[6] Grewatsch, S., Muller, E.: Sharing of Motion Vectors in 3D Video Coding. Proc. of International Conference on Image Processing (2004)

[7] Fehn, C.: A 3D-TV Approach Using Depth-Image-Based Rendering (DIBR). Visualization, Imaging, and Image Processing (2003) 482-487

[8] Fehn, C., Schuur, K., Feldmann, I., Kauff, P., Smolic, A.: Distribution of ATTEST test sequences for EE4 in MPEG 3DAV. ISO/IEC JTC1/SC29/WG11 M9219 (2002)

[9] Iddan, G., Yahav, G.: 3D Imaging in the Studio. SPIE's Videometrics and Optical Methods for 3-D Shape Measurement Vol. 7 (2003) 48-55

[10] Zitnick, C., Kang, S., Uyttendaele, M., Winder, S., Szeliski, R.: High-quality Video View Interpolation Using a Layered Representation. ACM Transaction on Graphics Vol. 23 No. 3 (2004) 598-606

[11] Wiegand, T., Sullivan G., Bjontegaard, G., Luthra, A.: Overview of the H.264 Video Coding Standard. IEEE Transaction on Circuits and Systems for Video Technology Vol. 13 No. 7 (2003) 560-576

[12] Wedi, T.: Motion Compensation in H.264. IEEE Transaction on Circuits and Systems for Video Technology Vol. 13 No. 7 (2003) 577-586

[13] Fehn, C., Hopf, K., Quante, Q.: Key Technologies for an Advanced 3D-TV System. Proceedings of SPIE Three-Dimensional TV, Video and Display, (2004) 66-80