Synchronization of texture and depth map by data hiding for 3D H. 264 video

Zafar Shahid, William Puech

SYNCHRONIZATION OF TEXTURE AND DEPTH MAP BY DATA HIDING FOR 3D H.264 VIDEO

Z. SHAHID

IRCCyN,UMR CNRS 6597,
Ecole Polytech, University of Nantes,
44306 Nantes, France
Email: zafar.shahid@univ-nantes.fr

W. PUECH

LIRMM,UMR CNRS 5506,
University of Montpellier II,
34095 Montpellier, France
Email: william.puech@lirmm.fr

ABSTRACT

In this paper, a novel data hiding strategy is proposed to integrate disparity data, needed for 3D visualization based on depth image based rendering, into a single H.264 format video bitstream. The proposed method has the potential of being imperceptible and efficient in terms of rate distortion trade-off. Depth information has been embedded in some of quantized transformed coefficients (QTCs) while taking into account the reconstruction loop. This provides us with a high payload for embedding of depth information in the texture data, with a negligible decrease in PSNR. To maintain synchronization, the embedding is carried out while taking into account the correspondence of video frames. Three different benchmark video sequences containing different combinations of motion, texture and objects are used for experimental evaluation of the proposed algorithm.

Index Terms— 3D TV, DIBR, H.264/AVC, data hiding

1. INTRODUCTION

The first black-and-white television has evolved into a high definition digital color TV and now the technology is ready to tackle the hurdles in the way of 3D TV *i. e.* a TV which provides the most natural viewing experience. One of the limitations of 3D TV is the bitrate, since a 3D video consists of more than one disparity files. In this work, we have proposed a method to have a 3D viewing experience without a significant increase in bitrate.

There exist several methods to generate 3D content. It can be in form of stereoscopic video, which consists of two separate video bitstreams: one for the left eye and one for the right eye. This systems has two main limitations. First, it has high bitrate overhead as compared to conventional 2D video. Second, it is not backward compatible. Another example of 3D content generation is proposed in [1], which consists of monoscopic color video (also known as texture) and associated per-pixel depth information. Using this data representation, 3D view of a real-world scene can then be generated at the receiver side by means of depth image based rendering (DIBR) techniques. The system is backward-compatible to conventional 2D TV and is scalable in terms of receiver complexity and adaptability to a wide range of different 2D and 3D displays. There exist algorithms which create 3D content offline. For example, 2D video can also be converted into 3D video using structure from motion techniques [2]. These techniques works in two ways. First, they extract the position of recording camera as well as the 3D structure of the scene can be derived. Second, they infer approximate depth information from the relative movements of automatically tracked image segments. Another example of offline 3D content generation is based on synchronized multi-camera systems, which can be used to have 3D analysis of video sequences [3]. In Section 2, overview of texture and depth based 3D video is presented, accompanied by an introduction of H.264/AVC. We have explained the proposed algorithm in Section 3. Section 4 contains experimental evaluations, followed by the concluding remarks in Section 5.

2. PRELIMINARIES

In texture and depth based 3D content, texture is a regular 2D color video. It is accompanied by depth-image sequence with the same spatio-temporal resolution. Depth information is an 8-bit gray value with the gray level 0 specifying the furthest value and the gray level 255 defining the closest value as shown in Fig. 1. To translate this data representation format to real, metric depth values for virtual view generation the gray values are normalized to two main depth clipping planes. The near clipping plane Z_{near} (gray level 255) defines the smallest metric depth value Z that can be represented in the particular depth-image. Accordingly, the far clipping plane Z_{far} (gray level 0) defines the largest representable metric depth value. In case of a linear quantization of depth, all other values can simply be calculated from these two extremes as:

$$Z = Z_{far} + \Omega \frac{Z_{near} - Z_{far}}{255},\ with\ \Omega \in [0, .., 255] \quad (1)$$

where Ω specifies the respective gray level.

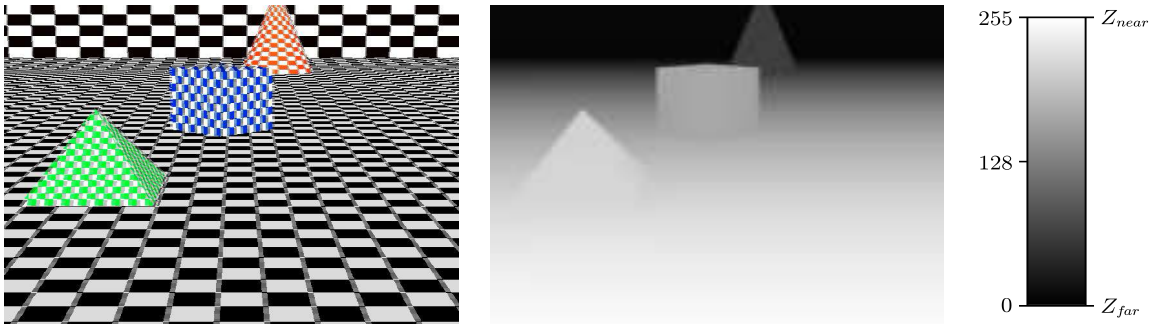


Fig. 1. An example of texture and depth for frame # 0 of *cg* sequences respectively.

Depth image based rendering (DIBR) is used for creating virtual scenes using a still or moving frame along with associated depth information [4]. Original pixel is projected in a 3D virtual world using depth information. It is followed by projection of 3D points in 2D plane of virtual camera.

H.264/AVC [5], let a 4×4 block is defined as $X = \{x(i,j)|i,j\epsilon\{0,3\}\}$. First of all, $x(i,j)$ is predicted from its neighboring blocks and we get the residual block.

This residual block E is then transformed using the forward transform matrix A as $Y = AEA^T$, where $E = \{e(i,j)|i,j\epsilon\{0,3\}\}$ is in the spatial domain and $Y = \{y(i,j)|i,j\epsilon\{0,3\}\}$ is in the frequency domain. Scalar multiplication and quantization are defined as:

$$\hat{y}(u,v) = sign\{y(u,v)\}[(\mid y(u,v) \mid \times Aq(u,v) + Fq(u,v) \times 2^{15+Eq(u,v)})/2^{(15+Eq(u,v))}], \quad (2)$$

where $\hat{y}(u,v)$ is a QTC, $Aq(u,v)$ is the value from the 4×4 quantization matrix and $Eq(u,v)$ is the shifting value from the shifting matrix. Both $Aq(u,v)$ and $Eq(u,v)$ are indexed by QP. $Fq(u,v)$ is the rounding factor from the quantization rounding factor matrix. This $\hat{y}(u,v)$ is entropy coded and sent to the decoder side.

3. THE PROPOSED ALGORITHM

In this paper, we propose to embed the depth information in texture data during H.264/AVC encoding process for 3D video data. In this way, we can avoid the escalation in the bitrate of 3D video. For 3D content with separate file for texture and depth, the overall bitrate is A+B, where A is the bitrate of the original texture and B is the bitrate of its depth map. We have reduced bitrate escalation by reducing the overall bitrate from A+B to A', where A' is very close to A. For this purpose, we have embedded depth information in the texture data in a synchronized manner at frame level. High payload fragile watermarking approach has been used for H.264/AVC, which we have already proposed in literature [6]. In this approach, embedding is performed in the QTCs while taking into account the reconstruction loop to avoid mismatch on encoder and decoder side. Moreover, hidden message is embedded in only those QTCs which are above a certain threshold. This embedding technique has two main advantages. First, it can recognize which QTCs have been watermarked and hence can extract the message on the decoder side. Second, it does not affect the efficiency of entropy coding engine to much extent. The embedding process is performed on QTCs as:

$$\hat{y}_w(u,v) = f(\hat{y}(u,v), M, [K]), \quad (3)$$

where $f()$ is the data hiding process, M is the hidden message and K is an optional key. Moreover $\hat{y}_w(u,v)$ is watermarked QTC while $y_w(u,v)$ is a QTC. For '1' LSB watermark embedding, $f()$ can be given as shown in Algorithm 1:

Algorithm 1 The embedding strategy in 1 LSB.

1: **if** $|QTC| > 1$ **then**
2: $\quad |QTC_w| \leftarrow |QTC| - |QTC| \; mod \; 2 + WMBit$
3: **end if**
4: **end**

Fig. 2 shows the block diagram of the proposed technique. The part of the framework related to depth data processing is drawn in red color. The process of embedding depth information in texture data is performed in three steps on a video frame. First, the depth information which is of the same resolution as *luma* is subsampled. Second, this subsampled depth information is compressed using H.264/AVC. For this purpose, depth information is regarded as *luma* component while chroma is treated as skipped. In third and final step, this subsampled and compressed depth information is embedded in the texture data during the encoding process of texture data. In this way we can embed depth information in texture data without a significant compromise on RD trade-off of texture data. On the decoder side, depth information is extracted and decoded using standard H.264/AVC decoder, and up-scaled to resolution of *luma*.

4. EXPERIMENTAL RESULTS

For the experimental results, three benchmark 3D video sequences , namely *interview*, *orbi* and *cg*, have been used for the analysis in resolution of 720×576 [7]. For comparison purposes, we have used PSNR for texture data and RMSE for

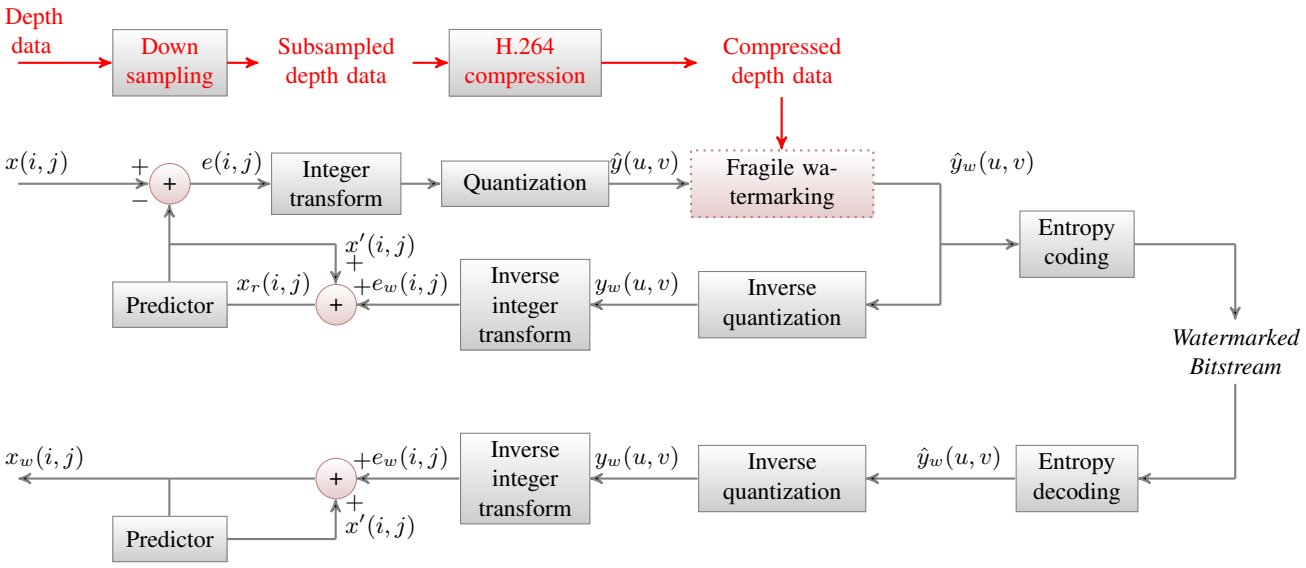


Fig. 2. Embedding of depth information in texture data using fragile watermarking scheme inside the reconstruction loop.

Table 1. Comparison of PSNR original video with embedded video of benchmark 3D video sequences at QP value 18.

	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
Seq.	Orig.	Embed	Orig.	Embed	Orig.	Embed
Interview	46.22	45.50	48.56	48.16	49.72	49.42
Orbi	46.94	46.40	50.06	49.83	49.35	49.00
Cg	37.35	37.46	51.91	51.62	51.44	51.10

Table 2. Average value of RMSE of extracted and up-scaled depth information with that of original depth information of 4 CIF res.

Seq.	RMSE
Interview	5.36
Orbi	2.88
Cg	4.20

depth information. To demonstrate our proposed scheme, we have compressed 250 frames of *interview*, and 125 frames of *orbi* and *cg* each, at 25 fps as *intra*. Depth information has been scaled down to 1 depth information for a texture block of 4×4. Table 1 contains a comparison of PSNR of original compressed video and those, which contains depth information for three video sequences at QP value of 18. One can note that the proposed algorithm works well for video sequences having various combinations of motion, texture and objects and is significantly efficient. The average decrease in PSNR for all the three encrypted sequences is $0.39\ dB$, $0.31\ dB$ and $0.33\ dB$, for Y, U and V components respectively. Increase in bitrate ((watermarked - original)/original) is 1.93, 1.88 and 0.62 for *interview*, *orbi* and *cg* respectively.

Framewise analysis of decoded and up-scaled depth information is presented in Fig. 3. Despite subsampling, RMSE value is very little for each frame. Generally, RMSE is lower for simple scences, but it is relatively higher for complex scences. Table 2 contains the RMSE value for extracted and scaled up depth information with that of original depth information of 4CIF resolution. RMSE value for *interview* sequence is 5.36 which is highest among all the three sequences. Hence, RMSE of up-scaled depth information is very controlled and there are not visual degradations in the depth information. For visual analysis, Fig. 4 shows the watermarked video frames along with the depth information which was embedded in them.

Payload capability for each frame is shown in Fig. 5.a for 125 frames of three benchmark sequences. One can note that texture data of each 3D video frame has lot more payload capability than required to embed depth information in it. Framewise rate-distortion (RD) trade-off is presented in Fig. 5.b for PSNR and in Fig. 5.c for bitrate. There is a negligible decrease in PSNR and very small increase in bitrate of texture data. It is evident that we can transmit the depth information in a synchronous manner with negligible overhead.

5. CONCLUSION

In this paper, a novel framework for embedding of depth data in texture is presented. Owing to negligible compromise on bitrate and PSNR, the embedding of depth data in H.264 video is an interesting framework. The experimental results have shown that we can embed the depth information in texture data of respective frame in a synchronous manner, while maintaining a good value of RMSE for depth data, under a minimal set of RD trade-off. In future we will extend our work in two aspects. First, the present algorithm will be

extended for *inter* (P and B frames). Second, the depth data will be compressed in a scalable manner to embed the highest possible amount of depth information.

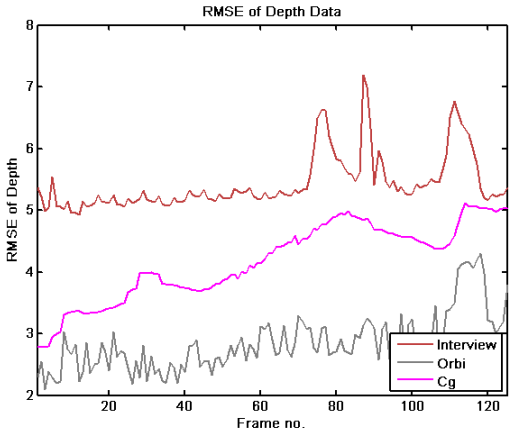


Fig. 3. RMSE of extracted and up-scaled depth information of 4CIF resolution with reference to the original depth information for 125 frames.

(a) Texture (b) Depth

Fig. 4. Texture and depth for frame # 0 of *interview*. Depth information has been embedded in texture in a synchronized manner.

6. REFERENCES

[1] P. Kauff, N. Atzpadin, C. Fehn, M. Mller, O. Schreer, A. Smolic, and R. Tanger, "Depth Map Creation and Image Based Rendering for Advanced 3DTV Services Providing Interoperability and Scalability," *Signal Processing: Image Communication*, vol. 22, pp. 217–234, 2007.

[2] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

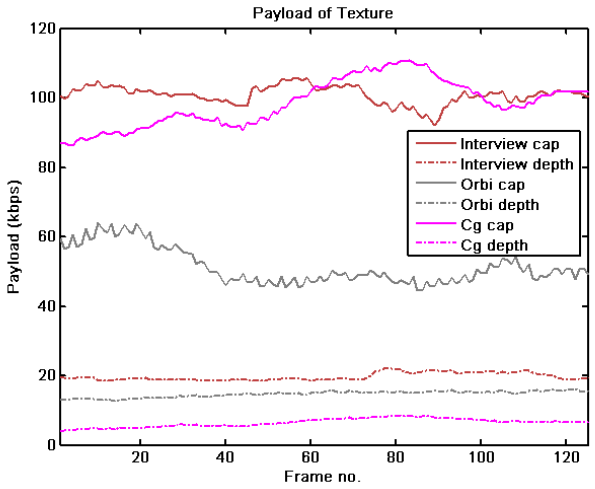
[3] J. Mulligan and K. Daniilidis, "View-Independent Scene Acquisition for Tele-Presence," Tech. Rep., Computer and Information Science, University of Pennsylvania,, 2000.

[4] L. McMillan, *An Image Based Approach for Three-Dimensional Computer Graphics*, Ph.D. thesis, University of North Carolina at Chapel Hill, 1997.

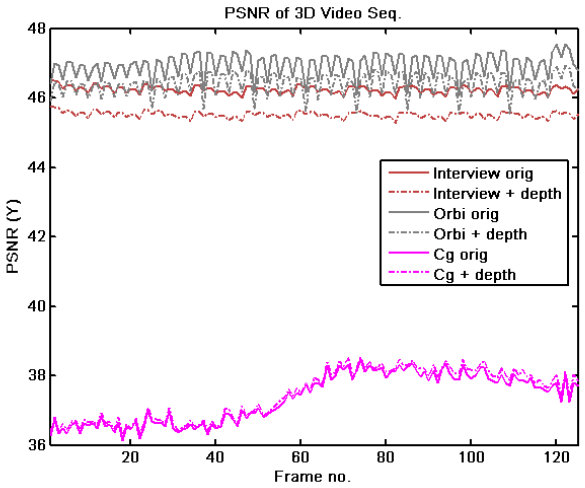
[5] H264, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 ISO/IEC 14496-10 AVC)," Tech. Rep., Joint Video Team (JVT), Doc. JVT-G050, March 2003.

[6] Z. Shahid, M. Chaumont, and W. Puech, "Considering the Reconstruction Loop for Data Hiding of Intra and Inter Frames of H.264/AVC," *Springer: Signal Image and Video Processing*, vol. 5, no. 2, 2011.

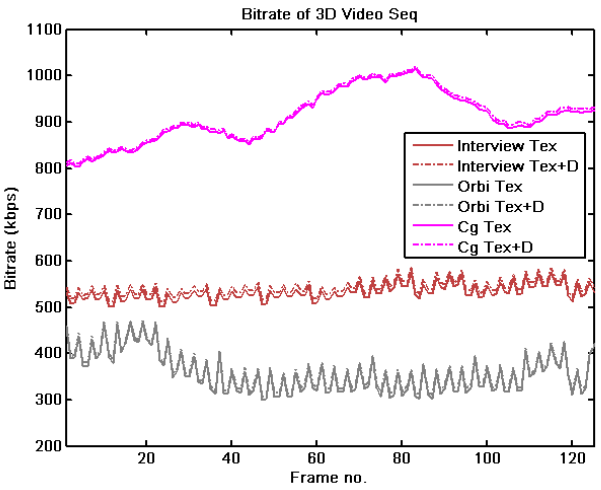
[7] C. Fehn, K. Schr, I. Feldmann, P. Kauff, and A. Smolic, "Distribution of ATTEST Test Sequences for EE4 in MPEG 3DAV," Tech. Rep. M9219, Heinrich-Hertz-Institut (HHI), 2002.



(a) Payload



(b) PSNR



(c) Bitrate

Fig. 5. Frame-wise analysis of 125 frames for 1 LSB embedding mode for texture data: (a) Available payload capacity along with actual payload of depth information of respective frame, (b) PSNR of original and watermarked video frames, (c) Bitrate of original and watermarked video frames.