Depth Boundary Filtering for View Synthesis in 3D Video

Yunseok Song, Cheon Lee, Woo-Seok Jang, and Yo-Sung Ho
Gwangju Institute of Science and Technology (GIST), Korea
E-mail: {ysong, leecheon, jws, hoyo}@gist.ac.kr

***Abstract*— This paper presents a boundary sharpening method for depth maps to improve synthesis view quality. In general, coded depth maps exhibit noise and artifacts around object boundaries, leading to ineffective view synthesis. In our approach, gradient information is used to extract depth boundary regions. Afterward, filtering based on distance, similarity, and direction is performed on such regions to replace depth values. The proposed algorithm was implemented on 3DV-ATM v0.3 as post-processing to coded depth maps. Experimental results showed 5.23% compared to the anchor results of 3DV-ATM v0.3. Subjective quality was improved as well.**

I. INTRODUCTION

3D video provides natural depth perception that could not be accomplished with 2D video. 3D displays such as auto-stereoscopic displays or free-view point TVs (FTV) are used at the receiver side. Recently, 3D video has received significant interests as one of the new-trend multimedia technologies. Since 2009, box office of 3D films has been highly successful, setting numerous new records. With the surging attention, now 3D contents are produced more than ever in various fields, e.g., documentary, sports broadcasting, and advertisement.

3D video is generated through the 3D video system where texture and depth data of multi-view video sequences are used as input; two or more viewpoints are required to create 3D effects.

Generally, the amount of data increases proportionally to the number of cameras. Moreover, cameras are expensive while consuming space. Due to such limitations, the number of cameras should be kept reasonable in practice. Thus, view synthesis is used to generate data of non-existing viewpoints.

In view synthesis, texture and depth data of neighboring views are utilized. Depth maps contain distance information of camera and objects. These allow for creation of 3D geometry data [1], exploited in the mapping process of corresponding pixels between different viewpoints.

However, in coded depth maps, noise and artifacts occur around object boundaries. Since depth map quality is closely related to the synthesized view quality, depth boundary processing is beneficial [2].

In this paper, we propose a method to effectively refine depth values around boundary regions. The purpose is to improve synthesized view quality specifically in the boundary region. The proposed method is applied as post-processing to coded depth maps.

II. BACKGROUND

In the 3D video system, view synthesis is achieved by interpolation and mapping. In this process, the qualities of texture and depth data of multiview video affect the synthesis view quality [3, 4]. Fig. 1 shows an example of coded depth maps when QPs of 40 and 50 were used. When synthesizing with these data while using the same texture quality, the quality of boundary regions in the higher QP case is much worse; this is represented in Fig. 2. Distortion around depth boundaries are propagated to the resulting synthesis views. Hence, boundary processing on coded depth maps is desirable.

To cope with 3D video, the moving picture experts group (MPEG) has formed 3D video coding (3DVC) group to develop an effective 3D video codec. After evaluating numerous technologies in AVC- and HEVC-compatible categories, test models (3DV-ATM for AVC-compatible, 3DV-HTM for HEVC-compatible) were released in 2011 [5]. 3DV-ATM is the basis in this paper.

Notable algorithms in 3DV-ATM include view synthesis prediction (VSP) and depth-based motion vector prediction (D-MVP) for texture coding. Depth-range-based weighted prediction (DRWP) and inside view motion prediction (IVMP) are used for depth coding.

(a) QP 40 (b) QP 50

Figure 1. Depth views of “Cafe”

(a) QP 40 (b) QP 50

Figure 2. Synthesized views of “Cafe”

III. Proposed Algorithm

The goal of the proposed method is to refine inaccurate depth values which occur around boundary regions. We first estimate these regions to classify where to apply filtering [6]. Then, we use a designed filter to remove noise and sharpen the boundary. The filter is based on similarity, distance, and direction. Weights are calculated for each factor, values being generalized to be in between 0 and 1. Such weights are then multiplied, higher weight meaning more reliability. The pixel within the window possessing the highest weight is chosen as the most dependable depth value; hence this replaces the value of center pixel. Fig. 3 shows the flowchart of the proposed algorithm.

A. Boundary region estimation

The proposed filter targets object boundaries in depth maps. In general, gradient data are used to represent directional information. We use this property to estimate boundary regions. For each frame, we calculate entire gradient magnitudes and use the standard deviation as the threshold.

If the estimated boundaries are thin, filtering may not be imposed enough. Thus we use a 3x3 mask to expand targets; boundary regions are expanded by 1-pixel, vertically and horizontally. In this case, the trade-off is slight complexity increase for higher gain. Fig. 4 displays an example of estimated boundary regions of “Dancer”. The boundaries of the human and pillars are clearly extracted.

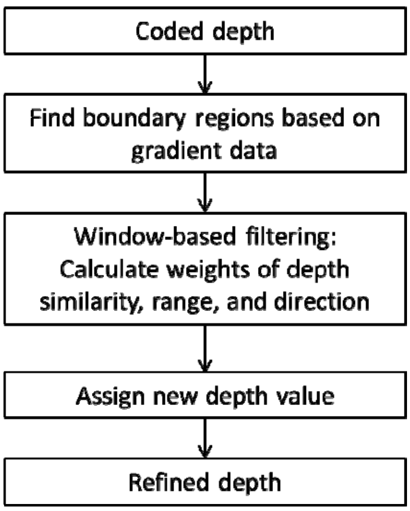


Figure 3. Flowchart of the proposed algorithm

Figure 4. Estimated boundary regions of “Dancer”

B. Depth boundary filter

The proposed depth boundary filter is based on three weights: similarity, distance, and direction. Such weights are represented by (1), (2), and (3), respectively. They are adjusted to possess a value between 0 and 1. The final weight is acquired by multiplying the three weights. In the window, the depth value with the highest weight replaces the depth value of center pixel.

First, the similarity weight function assigns more weight to pixels with depth values similar to the center pixel. Absolute differences are used to measure the distortion.

$$w\_sim_{p,q} = \exp\left(\frac{-\left|D_p - D_q\right|^2}{2\sigma^2}\right) \tag{1}$$

Second, range weight assumes farther pixels are more reliable. In coded depth maps, many inaccurate depth values exist around object boundaries. Thus, we assume that pixels farther from boundaries present less error.

$$w\_range_{p,q} = 1 - \exp\left(\frac{-[(p_x - q_x)^2 + (p_y - q_y)^2]}{2\sigma^2}\right) \tag{2}$$

Four directions are defined: horizontal, vertical, diagonal up-left, and diagonal up-right. Directions are equally partitioned. This is represented in Fig. 5. Fig. 6 shows positions of direction weights in accordance to boundary directions.

Higher weights are assigned to pixels which are located closer to the boundary orthogonal direction. Taking this into consideration, we designed a formula using a cosine function. By forming a line to connect the center pixel and neighboring pixel, the angle between such a line and boundary direction can be estimated. This is denoted as $\theta_{p,q}$. The angle is adjusted to control the range of its cosine value; this keeps the weight in between 0 and 1.

$$w\_direction_{p,q} = 1 - \cos(\theta_{p,q}), \quad 0 \le \theta_{p,q} \le \frac{\pi}{2} \tag{3}$$

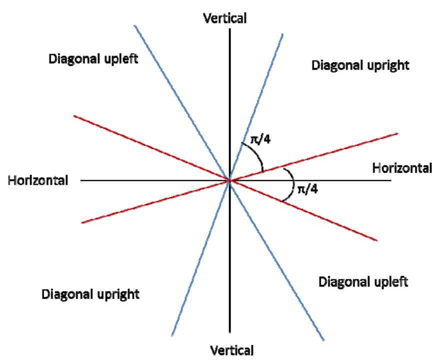
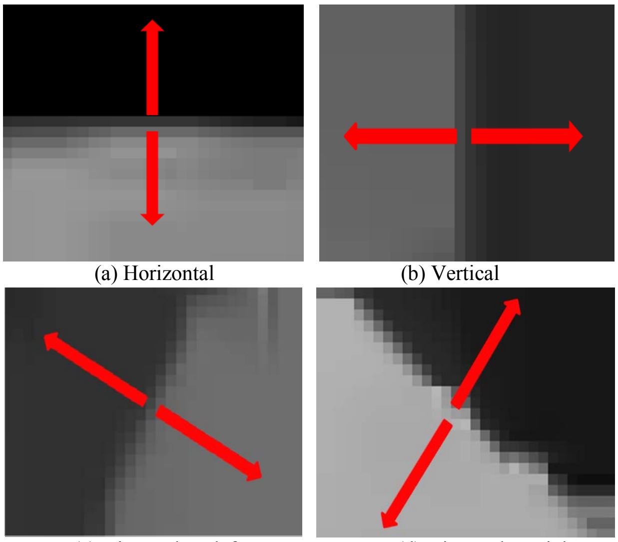


Figure 5. Direction partitions



(a) Horizontal (b) Vertical

(c) Diagonal up-left (d) Diagonal up-right

Figure 6. Direction weights

We precalculate the weights for each direction using (3). In this paper we use a 9x9 window. Table 1 represents the direction weights.

IV. EXPERIMENTAL RESULTS

The proposed method was implemented on 3DV-ATM v0.3. Experiments were conducted on "Dancer", "Kendo", and "Newspaper" sequences.

TABLE 1. DIRECTION WEIGHTS

0.29	0.4	0.55	0.76	1	0.76	0.55	0.4	0.29
0.2	0.29	0.45	0.68	1	0.68	0.45	0.29	0.2
0.11	0.17	0.29	0.55	1	0.55	0.29	0.17	0.11
0.03	0.05	0.11	0.29	1	0.29	0.11	0.05	0.03
0	0	0	0	0	0	0	0	0
0.03	0.05	0.11	0.29	1	0.29	0.11	0.05	0.03
0.11	0.17	0.29	0.55	1	0.55	0.29	0.17	0.11
0.2	0.29	0.45	0.68	1	0.68	0.45	0.29	0.2
0.29	0.4	0.55	0.76	1	0.76	0.55	0.4	0.29

(a) Horizontal boundary (weight: vertical)

0.29	0.2	0.11	0.03	0	0.03	0.11	0.2	0.29
0.4	0.29	0.17	0.05	0	0.05	0.17	0.29	0.4
0.55	0.45	0.29	0.11	0	0.11	0.29	0.45	0.55
0.76	0.68	0.55	0.29	0	0.29	0.55	0.68	0.76
1	1	1	1	1	1	1	1	1
0.76	0.68	0.55	0.29	0	0.29	0.55	0.68	0.76
0.55	0.45	0.29	0.11	0	0.11	0.29	0.45	0.55
0.4	0.29	0.17	0.05	0	0.05	0.17	0.29	0.4
0.29	0.2	0.11	0.03	0	0.03	0.11	0.2	0.29

(b) Vertical boundary (weight: horizontal)

0	0.01	0.05	0.14	0.29	0.49	0.68	0.86	1
0.01	0	0.02	0.11	0.29	0.55	0.8	1	0.86
0.05	0.02	0	0.05	0.29	0.68	1	0.8	0.68
0.14	0.11	0.05	0	0.29	1	0.68	0.55	0.49
0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
0.49	0.55	0.68	1	0.29	0	0.05	0.11	0.14
0.68	0.8	1	0.68	0.29	0.05	0	0.02	0.05
0.86	1	0.8	0.55	0.29	0.11	0.02	0	0.01
1	0.86	0.68	0.49	0.29	0.14	0.05	0.01	0

(c) Diagonal up-left boundary (weight: diagonal up-right)

1	0.86	0.68	0.49	0.29	0.14	0.05	0.01	0
0.86	1	0.8	0.55	0.29	0.11	0.02	0	0.01
0.68	0.8	1	0.68	0.29	0.05	0	0.02	0.05
0.49	0.55	0.68	1	0.29	0	0.05	0.11	0.14
0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
0.14	0.11	0.05	0	0.29	1	0.68	0.55	0.49
0.05	0.02	0	0.05	0.29	0.68	1	0.8	0.68
0.01	0	0.02	0.11	0.29	0.55	0.8	1	0.86
0	0.01	0.05	0.14	0.29	0.49	0.68	0.86	1

(d) Diagonal up-right boundary (weight: diagonal up-left)

The performance of the proposed algorithm was compared to the 3DV-ATM v0.3 under the common test conditions [7]. Notable test conditions include four-QP (26, 31, 36, 41) set usage and synthesizing via the latest rendering software—VS1D-fast. For each sequence, six views were synthesized. Fig. 7, Fig. 8, and Fig. 9 represent RD-curves for "Dancer", "Kendo", and "Newspaper", respectively. On average, -5.23% BD-BR and 0.16 dB BD-PSNR was achieved.

TABLE 2. "DANCER" RESULTS

QP	Bitrate (kbit/s)	PSNR (dB)	BD-BR (%)	BD-PSNR (dB)
26	6313.61	34.91	-8.43	0.25
31	3153.22	33.15		
36	1613.55	31.21		
41	863.72	29.32		

TABLE 3. "KENDO" RESULTS

QP	Bitrate (kbit/s)	PSNR (dB)	BD-BR (%)	BD-PSNR (dB)
26	2391.21	41.44	-2.02	0.09
31	1335.19	39.40		
36	746.87	36.92		
41	434.59	34.11		

TABLE 4. "NEWSPAPER" RESULTS

QP	Bitrate (kbit/s)	PSNR (dB)	BD-BR (%)	BD-PSNR (dB)
26	2090.24	37.90	-3.69	0.14
31	1114.24	36.16		
36	613.95	34.03		
41	362.28	31.62		

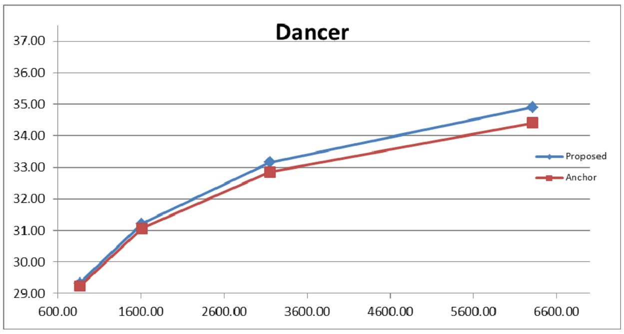


Figure 7. “Dancer” RD-curve

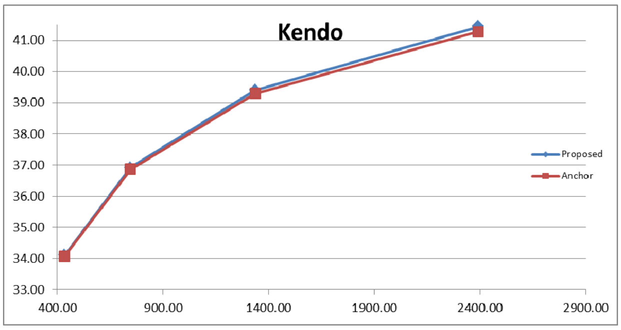


Figure 8. “Kendo” RD-curve

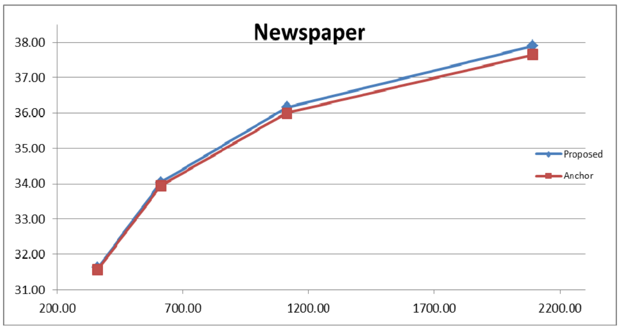
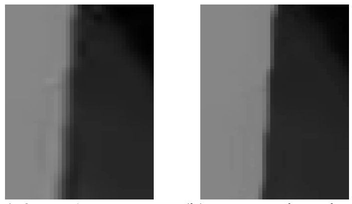


Figure 9. “Newspaper” RD-curve

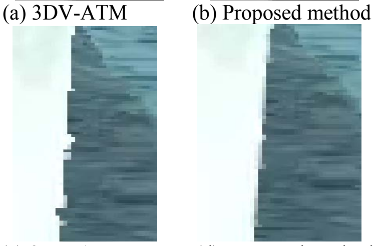
The highest gain was achieved in “Dancer” simulation. This is due to the highest quality of pre-coded depth data among the three. Depth data of “Dancer” are computer-generated while that of “Kendo” and “Newspaper” are estimated by using depth estimation reference software (DERS). Generally, computer-generated depth data show higher depth accuracy than DERS.

As shown in the above RD-curves, the proposed algorithm performs better in lower QPs, meaning better quality of coded data. Note that the filter is window-based. If the number of inaccurate depth values within the window is large, the proposed algorithm may wrongfully assign high weights to meaningless depth values.

Fig. 7 displays subjective quality comparison of “Dancer” where QP 31 was used. Compared to 3DV-ATM, the proposed algorithm produces more accurate depth values at the boundary region; the boundary is sharpened with reduced blurring. This creates smoothly connected boundary in the synthesized view.



(a) 3DV-ATM (b) Proposed method



(c) 3DV-ATM (d) Proposed method

Figure 7. Subjective quality comparison, “Dancer” (QP 31)

V. CONCLUSIONS

In this paper, we introduced a boundary sharpening algorithm for depth data. The method was implemented on 3DV-ATM v0.3. The proposed filtering is applied to depth boundary regions. First we calculate gradient magnitudes of each pixel and use the standard deviation as the threshold for depth boundary region extraction. Afterward, window-based depth boundary filtering based on similarity, range, and direction allows selecting the best depth value to replace the center pixel. High weights are assigned to similar, farther pixels located in the orthogonal direction of the boundary direction. Experimental results show -5.23% BD-BR and 0.16 dB BD-PSNR compared to the anchor results of 3DV-ATM v0.3. Subjective quality improvement was confirmed as well.

ACKNOWLEDGMENT

This research is supported by MCST and KOCCA in the CT Research & Development Program 2012.

REFERENCES

[1] E.K. Lee and Y.S. Ho, “Generation of multi-view video coding using a fusion camera system for 3D displays,” *IEEE Trans. on Consumer Electronics*, vol. 56, no. 4, pp.2797-2805, Nov. 2010.

[2] Y. Song, C. Lee, and Y.S. Ho, “Adaptive depth boundary sharpening for effective view synthesis,” *Picture Coding Symposium (PCS)*, pp. 73-76, May 2012.

[3] S. Liu, P. Lai, D. Tian, C. Gomila, and C.W. Chen, “Joint trilateral filtering for depth compression,” *Proc. of SPIE Visual Communications and Image Processing*, July 2010.

[4] C. Lee, K.J. Oh, and Y.S. Ho, "View Interpolation Prediction for Multi-view Video Coding," in *Picture Coding Symposium (PCS)*, pp. 1-4, Nov. 2007.

[5] ISO/IEC JTC1/SC29/WG11, “Test model under consideration for AVC-based 3D video coding (3DV-ATM),” n12349, Dec. 2011.

[6] ISO/IEC JTC1/SC29, “3D-CE4.a related: Depth boundary filtering,” m24947, Apr. 2012.

[7] ISO/IEC JTC1/SC29, “Common test conditions for AVC and HEVC-based 3DV,” n12560, Feb. 2012.