A New In-Loop Filter for Depth Map Coding in HEVC

Hyunsuk Ko and C.-C. Jay Kuo
Ming Hsieh Department of Electical Engineering, University of Southern California, CA 90089-2564, USA
E-mail: kohyunsu@usc.edu and cckuo@sipi.usc.edu

***Abstract*—A depth-map is used to synthesize virtual texture views in the multi-view plus depth (MVD) format. In conventional video coding, a coded depth-map often suffers from compression artifacts along object boundaries, which have a negative effect on the quality of rendered images in the view synthesis process. To address this problem, we propose a depth-map boundary filtering technique to eliminate coding artifacts while preserving sharp edges. This can be mathematically formulated as a L0-norm minimization problem. This filtering process is cascaded with the de-blocking filter in the emerging HEVC video coding standard to result in a new in-loop filter. Experimental results are given to show that the subjective and objective quality of the synthesized views is enhanced by the introduction of the new in-loop fitler.**

Keywords- Boundary filtering, depth-map coding, view synthesis, L0-norm minimization

I. INTRODUCTION

With improved 3D display technologies and great market success of 3D films, the interest in 3D video is increasing rapidly in recent years. The 3D video technology is expected to expand to home applications such as the Free-Viewpoint TV (FTV) and 3D television (3DTV). The former allows a viewer to navigate a given 3D scene by his/her own choice while the latter offers stereo views to viewers at multiple angles. In order to meet the quality requirements of these applications, there has been a great amount of research on proper 3D video data formats and processing such as compression.

As the simplest form of 3D video representation, a stereoscopic image pair, which consists of the left and the right views, is able to provide a realistic 3D scene. However, a viewer should be located at a proper position to enjoy the 3D visual experience, which is usually the center position in front of the display. The multi-view video (MVV) was introduced to allow a more flexible range of viewing angles. It provides multiple stereo pairs targeting at the same scene but observed from different view angles. In order to compress the MVV format efficiently, the Multi-view Video Coding (MVC) standard [1] was already finalized by the Joint Video Team (JVT) as an extension of the H.264/AVC standard. Although MVC can offer both 3D perception and view navigation, it has limitations. First, 3D video acquisition with a large number of cameras is usually difficult and expensive. Second, although MVC is more efficient than the simulcast coding, its coding rate is still proportional to the number of capturing cameras (or called the input views). Since the user satisfaction of 3D visual experience improves as the number of input views increases, the amount of data to be processed may go beyond the affordable computational complexity.

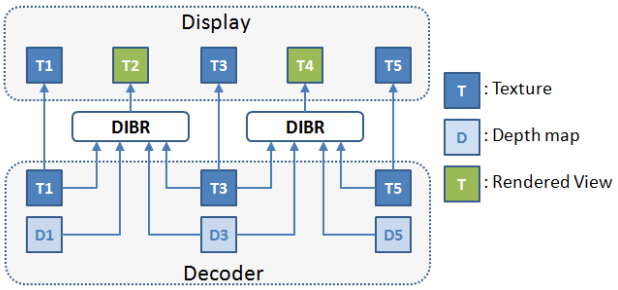


Figure 1. Illustration of the view synthesis process with the MVD video format.

To address this challenge, the Moving Pictures Experts Group (MPEG) has launched a new standardization effort, called the 3D Video Coding (3DVC), as the second phase of FTV [2]. 3DVC adopted the multi-view plus depth (MVD) format. For the MVD data format, we have multiple texture views and the corresponding depth-maps. Theoretically, the system based on the MVD format can generate infinite intermediate views from sparse input texture/depth pairs using the view synthesis technique, which is usually known as depth- image-based rendering (DIBR) [3]. Currently, the 3DVC standardization is progressing with both MVV and MVD simultaneously. The system using the MVD format is shown in Figure 1.

The main objective of depth coding in 3DVC is not only to compress the depth data efficiently but also to guarantee sufficiently high quality of a synthesized view. Although many studies on increasing coding efficiency in depth coding have been done, there is relatively little research on ensuring the quality of a rendered view. With the state-of-the-art HEVC coding standard [4], which outperforms H.264/AVC about twice in coding efficiency, we can have a high coding gain in depth-map compression due to its simple structure and texture. However, even with new in-loop filtering techniques adopted

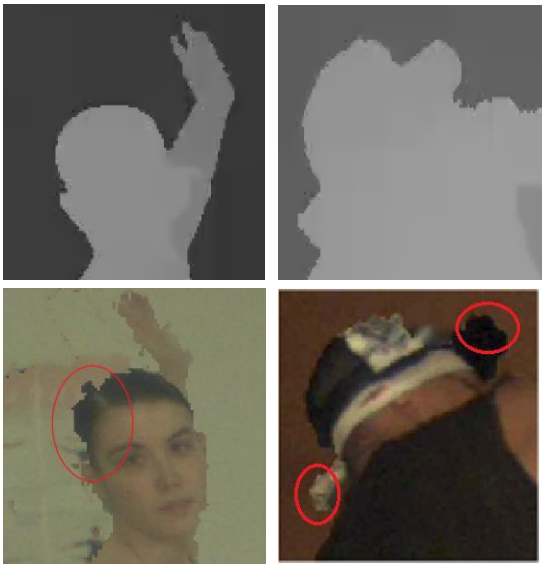


Figure 2. The visual artifacts in synthesized views.

by HEVC such as the Sample Adaptive Offset (SAO) and the Adaptive Loop Filter (ALF) [5], we cannot eliminate artifacts in the reconstructed depth-map and the rendered views.

The depth-map video has characteristics different from conventional video data. Concretely, the depth image usually consists of piecewise linear segments bounded by strong edges which means discontinuities. Therefore, the coding of the depth map with conventional tools such as the DCT transform followed by quantization results in severe distortion along the object boundary. This is a critical problem since the depth map is used for intermediate view rendering (rather than direct display). In the view synthesis process, the pixel value in the original depth map is related to a disparity change in the rendered views, and a small error around object boundary leads to severe subjective quality degradation in synthesized views.

There are several typical artifact types in a decoded depth map. First, the blocking artifact resulting from block partitioning, which is widely used in coding standards, introduces some unexpected "false" contours in a rendered view. Second, the blurring and ringing artifacts around sharp edges in a depth map introduce corrupted or growing edges. To illustrate the DIBR problem using a distorted depth map, we show examples of synthesized views from two pairs of compressed texture/depth data using the View Synthesis Reference Software (VSRS) [6] in Fig. 2.

In this work, we propose a depth boundary filtering method with an objective to improve the quality of the synthesized view for the emerging HEVC video coding standard. This filtering process is integrated with the quad-tree structure of the emerging HEVC video coding standard as an in-loop filter. In particular, the L0-norm minimization technique is adopted to maintain sharp edges while eliminating coding artifacts around object boundaries.

The rest of this paper is organized as follows. In Section 2, we review related previous work and theoretical background required for our work. The depth boundary filtering method is proposed in Section 3, and experimental results are presented in Section 4. Finally, concluding remarks are given in Section 5.

II. Review of Previous Work

By following the framework in [7], Lai *et al.* [8] proposed a noise removal filtering technique based on the sparse representation as an in-loop filter. The de-noising operation is first achieved by thresholding transform coefficients. Then, they are re-constructed with a set of sparse transform basis. However, it has a shortcoming. That is, the computational complexity of the weighting pursuit process, which is used to assign a weight to a correlated basis, is high.

Oh et al. [9] proposed a trilateral filtering method to reconstruct the depth boundary and use it as an in-loop filter. They added one additional factor, called the occurrence frequency, to the conventional bilateral filter [10]. The bilateral filter outputs the weighted sum of two kennel functions. They are the domain kennel, which considers closeness among pixels, and the range kennel, which addresses the intensity value difference. In [9], Oh et al. examined the occurrence frequency, pixel value similarity, and pixel position closeness.

Liu at al. [11] proposed another trilateral in-loop filter that exploits the structure similarity between the depth map and its corresponding color video. It aims at coding artifact removal based on the proximity of pixel positions, the similarity of depth samples, and the similarity among collocated pixels in the video frame.

However, these solutions share one common problem. That is, most of them tried to eliminate artifacts using a weighted averaging technique. For example, the bi-/tri-lateral filters use the following Gaussian function as its kennel:

$$\hat{I}(x) = \frac{1}{K} \sum_{y \in N(x)} e^{\frac{-\|y-x\|^2}{2\sigma_d^2}} e^{\frac{-|I(y)-I(x)|^2}{2\sigma_r^2}} I(y) \quad (1)$$

This approach is suitable for removing noise around object boundaries and/or filling discontinuity along edges. On the other hand, it introduces a certain degree of blurring around edges, which might be acceptable in conventional texture coding but not in depth map coding because it results in a disparity error in the rendering process. As a result, the subjective quality of the rendered view is degraded with an annoying artifact as mentioned before.

The algorithm proposed in this work was inspired by the work of Li at al. [12], which used the L0-Norm minimization. It is originally designed for image smoothing, and it is particularly effective for sharpening sharp edges by increasing the slope of transition while eliminating the negligible structure such as noise of low magnitude. The optimization scheme proposed in [12] attempts to represent the whole image with a restricted number of intensity changes as given by

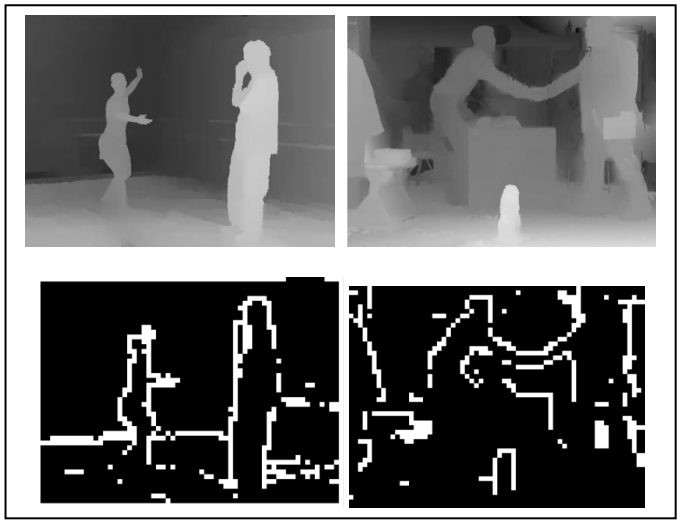


Figure 3. Detection results of boundary blocks.

$$\min_S \left\{ \sum_p (S_p - I_p)^2 + \lambda \cdot C(S) \right\},$$

$$C(S) = \#\{ p \mid |\partial_x S_p| + |\partial_y S_p| \neq 0 \}, \qquad (2)$$

where I is the input image, S is the smoothed image, C(S) is the counting function which outputs the number of pixels that have neighboring pixels of different intensity values (indicating that the L0-gradient of the pixel is not zero), and λ is a weight parameter to control the degree of smoothness. To reduce the total energy in Eq. (2), the intensity change must occur at dominant edges. As a result, salient edges in the smoothed result would coincide with those in the original image while other weak fluctuations will be smoothed globally. It was shown in [12] that this solution offers better results than other solutions such as the bilateral filter, the weighted least squares, and the total variance.

In this work, we adopt the same L0-gradient minimization filtering approach as proposed in [12] and call it the L0-filtering in short. Our main contribution here is to tailor this framework to the new HEVC video coding standard. Specifically, the depth-map boundary filtering process is integrated with the quad-tree structure of the emerging HEVC video coding standard as an in-loop filter.

III. Depth-Map Boundary Filtering as HEVC In-Loop Filter

A. Boundary Block Detection

The L0-filtering was originally designed based on the global image statistics (i.e. all pixels in a frame) in [12]. To tailor it to the HEVC coding framework, we modify it so that it can work in the context of block-based processing. Since it might introduce another blocking artifact if the L0-filter is applied to all blocks in a frame, it is only applied to blocks in object boundaries. This is called the region-based filtering.

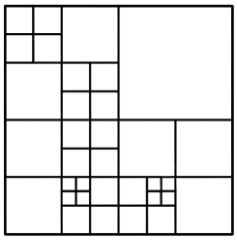


Figure 4. An example of LCU's Quad-tree structure

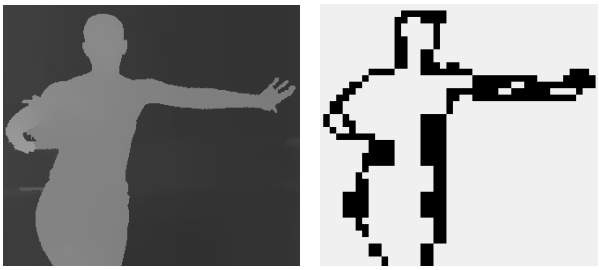


Figure 5. The quad-tree based detection result.

Here, we use the standard deviation of a block to detect where it is a boundary block since non-boundary blocks usually consist of homogeneous pixel values and have a smaller variance. Only when the standard deviation of a block exceeds a pre-defined threshold value, we perform the L0-filtering. The standard deviation is for a NxN block is

$$STD = Sqrt\left\{ \frac{1}{NxN} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left[I(i,j) - Mean_{blk} \right]^2 \right\} \qquad (3)$$

where N is the block size, $I(i,j)$ is the pixel intensity, and $Mean$ is the mean of the block. Figure 3 shows examples of detected boundary blocks. We observe that blocks that contain salient object edges are selected.

B. Quad-Tree Structure

As compared with previous video coding standards, one distinguishing feature of HEVC is that its coding is performed based on the coding unit (CU) of a variable block size [13]. Currently, CU can have a block size from 4x4 to 64x64. The largest CU is called as LCU (Largest Coding Unit), which can be partitioned hierarchically using a quad-tree structure to determine the best decomposition in terms of the rate-distortion (RD) performance. An example is shown in Figure 4.

For the L0 filtering, we use the optimally decomposed quad-tree structure, which is obtained after encoding each LCU as a basic unit, in detecting boundary blocks. This is because we can avoid over/under filtering by considering the regional characteristics of a block. Usually, blocks that contain object boundaries are usually encoded with variable block sizes while homogeneous blocks are more likely to be encoded with a larger block size. An example of a quad-tree decomposition of the depth map and the distribution of its boundary blocks are shown in Figure 5 to illustrate the above point. Therefore, we adopt the quad-tree structure of LCU in HEVC. For every CUs in LCU, we check whether its standard

(a) Original depth-map (b) Reconstructed depth-map with De-blocking filtering (c) Blocks to be filtered

(d) Reconstructed depth-map with L0-filtering (e) Reconstructed depth-map with De-blocking fitlering after L0-filtering

Figure 6. Illustration of (a) the coded depth map, (b) the reconstructed depth map with the de-blocking filter as the in-loop filter, (c) the blocks where the in-loop filter is applied, (d) the reconstructed depth map with the L-0 filtering as the in-loop filter, and (e) the constructed depth map with the proposed cascaded filters as the In-loop filter.

deviation is above a certain threshold. If the condition is met, we perform the L0-filtering in this block.

C. In-loop Filtering

There are three in-loop filtering techniques in HEVC; namely, the de-blocking filtering, the Sample Adaptive Offset (SAO) and the Adaptive Loop Filter (ALF). The de-blocking filter is a must while SAO and ALF are optional. The in-loop filter reduces the prediction residual by including the filtering process in the encoding loop [6].

We implement the in-loop filter by cascading the L0 filter and the de-blocking filter. This implementation is adopted for the following reasons. First, we observe that the L0 filter could eliminate most coding artifacts while maintaining sharp edges as shown in Figure 6, where all figures were sharpened to show the phenomenon clearly. Figure 6(a) is the original depth-map, and Figure 6(b) is the reconstructed depth-map using HM5.0 [14], which is the reference software of HEVC, with its quantization parameter (QP) set to 32. There exist severe coding artifacts (mainly the ringing artifact in this case) around the boundaries even after the de-blocking filtering. Note that the de-blocking filter is only applied to boundary blocks as shown in Figure 6(c). Figure 6(d) is the result after the application of the L0 filter only. Although many annoying artifacts were eliminated, we observe new blocking artifacts introduced by the L0-norm filter around the arm region as shown in Figure 6 (d). This is due to the use of the block-based L0-filter, which represents the original block with a sparse number of representative values. Finally, we show the result of the proposed solution that has the L0-filter and the de-blocking filter in cascade as the new in-loop filter in Figure 6(e), where we hardly see any blocking artifact. The overall flow chart of the proposed in-loop filter for HEVC is shown in Figure 7.

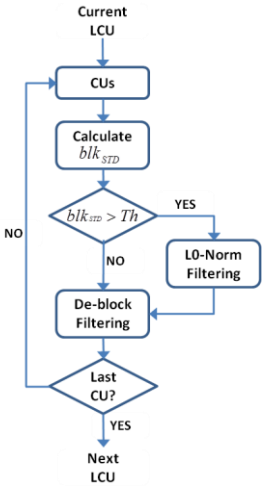


Figure 7. The flow chart of the proposed in-loop filter.

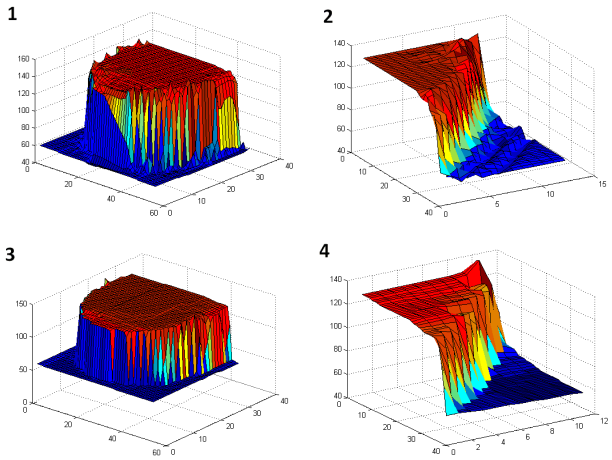


Figure 8. Graphical illustration for the improvement

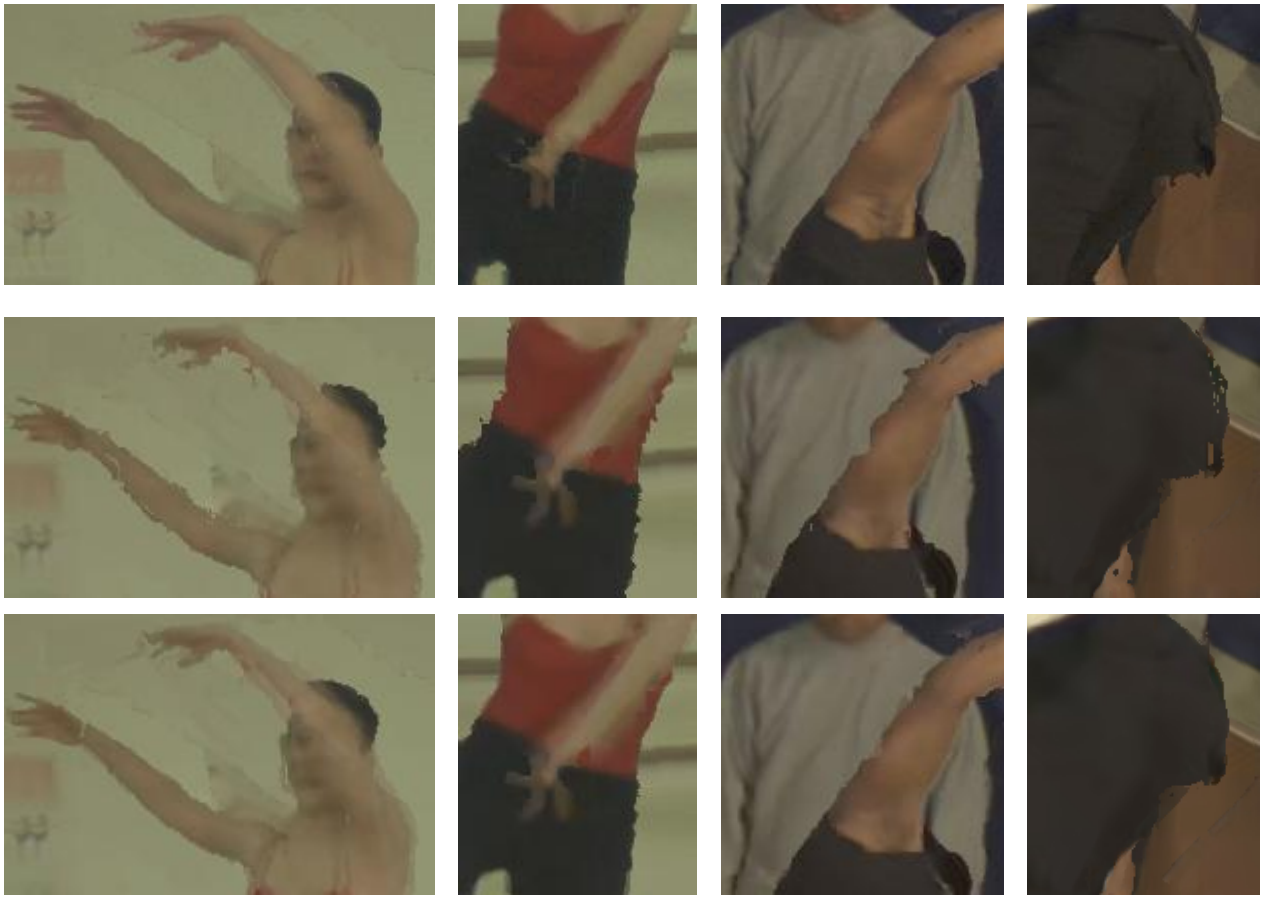


Figure 9. Subjective quality improvement in the synthesized View (a) the first row: the reference view with the original color image and the depth map, (b) the second row: the synthesized view with the de-blocking filter only, (c) the third row: the synthesized View with the cascade of the de-blocking filter and the L0 filter.

The 3D graphic plots of the neighborhood of a sharp edge for two representative cases are shown in Figure 8, where cases 3 and 4 are the improved edges for cases 1 and 2, respectively. We see that most coding artifacts along the object boundary are removed. Accordingly, the boundary region is aligned well with the salient boundary of the original image.

IV. Experimental Results

We used HM5.0 with the low-delay configuration [5] in the experiments. Two sequences, 'ballet' and 'break dancers', were encoded for both the texture video and the depth-map video with 50 frames. For QP values, 26, 32, 36, and 41 were used as specified in the simulating condition for depth coding in the 3DVC standard by MPEG [15], and four reference frames were used for inter coding. Among the 8 views, view 3 and view 5 were used as reference views while view 4 was set to the virtual view. The virtual view was synthesized with VSRS 3.5 software [16].

A. Subjective Quality Improvement

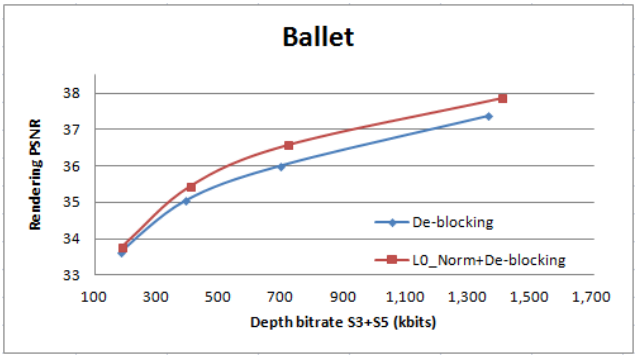
Figure 9 shows the synthesized view (i.e., view 4) of the proposed in-loop filter. The results in the first row were rendered with the original texture/depth-map as a reference. The de-blocking filter was only applied to the results in the second row while the results in the last row were processed using the L0-filter followed by the de-blocking filter. The visual artifacts appear in the second row due to the distorted depth-map especially around object boundaries. After applying the L0-filter, severe false contours were improved and holes in the object were filled. Thus, it provides enhanced subjective quality.

B. Objective Quality Improvement

In terms of objective quality, we plot the rate-distortion (RD) curve, where the x-axis is the bit rate used to encode the left/right depth-maps and the y-axis is the PSNR value against the reference synthesized view (the first row in Figure 8), which is computed as follows.

$$PSNR_{Syn} = 10 \times \log_{10}\left(\frac{255^2}{MSE_{Syn}}\right)$$

$$MSE_{Syn} = \frac{1}{N \times N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left[I_{Ref}(i,j) - I_{Syn}(i,j)\right]^2 \quad (4)$$



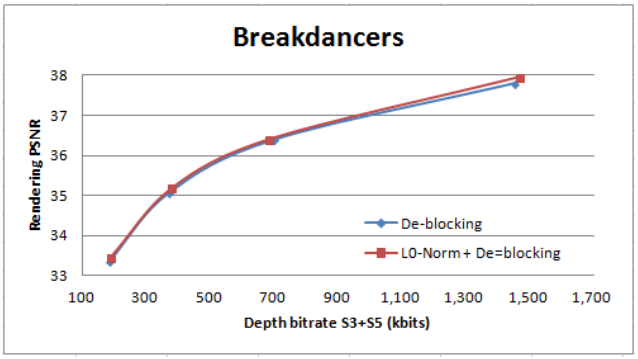


Figure 10. RD-Curves of Test Sequences

where N is block size, $I_{Ref}(i,j)$ and $I_{Syn}(i,j)$ are pixel intensity values of the reference view and the synthesized view, respectively.

Figure 10 shows the RD-curves of the two test sequences. As shown in the figure, the proposed L0-filter cascaded by the de-blocking filter has a PSNR gain in both cases. For the ballet sequence, the coding gain is obvious over the entire bit rate range while the coding gain for the break-dancers sequence is less obvious. Note that the rendering quality without applying the L0-filter for the break-dancers sequence is better than that of the Ballet sequence. Thus, the room for improvement is less by introducing the L0-filter. In both results, we can observe that coding gain in the range of low bit-rate is very small comparing to the range of higher bit-rate. This is because the fidelity of coded-depthmap itself is very poor when we used large QP value. Therefore, even though we apply L0-filtering, the chances of improvement are so small.

C. *Complexity Analysis*

For L0-filtering, we should solve the optimization problem in Eq. (2). In general, it is known that L0 optimization problem is computationally costly. However, a solution in [12] can reduce the complexity significantly by using approximation. The FFT is a key function of the solution, and we used FFTW [17] which is open library ensuring fast speed. We analyzed the computational complexity of L0-filtering by comparing the encoding time between original encoder and the proposed algorithm, which is summarized in Table 1. The increase of encoding time is under 2% in every case, thus it is negligible.

	Encoding time of HM 5.0 (sec)	Encoding time of Proposed (sec)	Time Increase (%) $\left(\frac{column\_3}{column\_2}\times 100\right)$
Ballet (V3)	647	658	101.7
Ballet (V5)	638	651	102.0
Breakdancers (V3)	696	701	100.7
Breakdancers (V5)	693	704	101.6

Table 1. Encoding Time Increase for Each Sequence: One reference frame was used for Inter-coing

V. Conclusion and Future Work

We proposed an in-loop filtering technique and applied it to the boundary blocks of the depth map video in this work. The in-loop filter contains an L0-norm minimization filter to remove coding artifacts while maintaining sharp edges in the coded depth-map as well as a de-blocking filter. This is important because artifacts around object boundaries introduce a severe artifact in the synthesized view in the rendering process so that the subjective quality is degraded significantly. It was demonstrated by preliminary experimental results that the proposed in-loop filter improves the subjective quality of the rendered view as well as the objective quality metrics for a simple test sequence "Ballet". More experiments are needed to demonstrate the advantage of the proposed in-loop filter in the near future.

References

[1] Text of ISO/IEC 14496-10:2008/FDAM 1 Multiview Video Coding, document w9978, ISO/IEC JTC1/SC29/WG11, Oct. 2008.

[2] Text of ISO/IEC JTC1/SC29/WG11 MPEG2011/N12035 Geneva, Switzerland Mar. 2011

[3] A. Smolic, K. Müller,K. Dix, P. Merkle, P. Kauff, and T.Wiegand, "Intermediate view interpolation based on multiview video plus depth for advanced 3D video systems," in Proc. IEEE Int. Conf. Image Process. (ICIP), San Diego, CA, Oct. 2008

[4] G. J. Sullivan and J.-R. Ohm, "Recent developments in standardization of high efficiency video coding (HEVC)," Proc. SPIE, vol. 7798, Aug. 2010.

[5] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 2nd Meeting: Geneva, CH, 21-28 July, 2010, JCTVC-B310, 'Tool Experiment 10: In-loop filtering'

[6] M. Tanimoto, T. Fujii, and K. Suzuki, "View synthesis algorithm in view synthesis reference software" Tech. Rep. Document M16090, SO/IEC JTC1/SC29/WG11, Feb. 2009.

[7] C.C. Dorea, O. Divorra Escoda, P. Yin, and C. Gomila, "A direction adaptive in-loop deartifacting filter for video coding," in Proc. IEEE ICIP, San Diego, CA, Oct. 2008, pp. 1624–1627.

[8] P. Lai, A. Ortega, C.C. Dorea, P. Yin, and C. Gomila, "Improving view rendering quality and coding efficiency by suppressing compression artifacts in depth-image coding," in Proc. SPIE VCIP, San Jose, CA, Jan. 2009.

[9] Kwan-Jung Oh, Anthony Vetro, and Yo-Sung Ho, "Depth Coding Using a Boundary Reconstruction Filter for 3-D Video Systems", IEEE Trans. on Circuits and Systems for Video Technology, Vol., 21 NO. 3, Mar 2011.

[10] C.Tomasi and R.Manduchi, “Bilateral filtering for gray and color images,” in Proc.ICCV, pp. 839–846.

[11] S. Liu, P. Lai, D. Tian, and C. W. Chen, “New Depth Coding Techniques With Utilization of Corresponding Video,” IEEE Trans. on Broadcasting, vol. 57, no. 2, pp. 551-561, 2011.

[12] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia “Image Smoothing via L0 Gradient Minimization”, ACM Transactions on Graphics Vol 30, No.6, December 2011.

[13] JCT-VC, “HM3: High Efficiency Video Coding (HEVC) Test Model 3 Encoder Description, JCTVC-E602, 5th JCT-VC Meeting: Geneva, CH, 16-23 March, 2011.

[14] http://hevc.hhi.fraunhofer.de/

[15] ISO/IEC JTC1/SC29/WG11 MPEG2011/NXXXX, Common Test Conditions for AVC and HEVC-based 3DV

[16] VSRS:http://wg11.sc29.org/svn/repos/MPEG/test/trunk/3D/view_synthesis/VSRS

[17] http://www.fftw.org/