# Adaptive Depth Boundary Sharpening for Effective View Synthesis

Yunseok Song, Cheon Lee, and Yo-Sung Ho
Gwangju Institute of Science and Technology (GIST)
South Korea
{ysong, leecheon, hoyo}@gist.ac.kr

Hocheon Wey and Jaejoon Lee
Samsung Advanced Institute of Technology
South Korea
{hc.wey, jaejoon1.lee}@samsung.com

***Abstract*—This paper focuses on sharpening boundaries in depth maps for multi-view video coding. Artifacts around boundaries degrade the quality of synthesized images. In order to encounter this problem, after applying the deblocking filter for each frame, we create a binary edge map and find the location of blocks that need to be altered. Subsequently, we apply a boundary sharpening filter which uses pixel frequency, similarity, and closeness as sub-costs. This filter is only applied to blocks which are near edges. Experimental results exhibit much more visual comfort level in synthesized images compared to when JMVC was used.**

***Keywords-depth coding, multi-view video coding, view synthesis, deblocking filter***

## I. INTRODUCTION

In recent years, 3D video has emerged as one of the rising multimedia technologies. 3D video provides a natural and realistic perception to the audience via 3D displays, e.g., auto-stereoscopic displays or free-view point TVs (FTV). In particular, both financial and critical successes of 3D films have sparked the 3D trend; now 3D contents are being produced more than ever. With the growing expectation levels from consumers, effective 3D video processing is vital.

3D video can be generated through the 3D video system where the input data includes multi-view video sequences and their corresponding depth data. In general, two or more cameras with slightly different viewpoints are required to capture images for creating 3D effect. In general, due to cost and space limitations, the number of cameras cannot be sufficiently high. Thus, the 3D video system should generate virtual views, i.e., views that are not originally captured from cameras. By exploiting the data from neighboring views, the virtual view can be implemented. This process is called view synthesis. Depth maps contain the information of distance between camera and objects, which can be utilized in creating 3D geometry data [1]. Then, the geometry data is exploited in the mapping process of corresponding pixels between different viewpoints. However, in depth map coding, blocking artifacts occur due to block-based processing, which ultimately deteriorate the quality of synthesized views.

Many engineers have proposed solutions to address this issue. Dorea et al. have adopted color-based region merging [2]. Oh et al. have proposed a filter using frequency, similarity and distance [3]. Liu et al. presents a filter based on similar characteristics [4]. In this paper, we propose a method to effectively reduce blocking artifacts around boundaries in depth maps, eventually making the synthesized view visually appealing.

The rest of this paper is organized as follows. In Section II, we explain the objective of deblocking filtering and boundary sharpening. Afterward, we propose our algorithm in Section III. Finally, the paper is concluded in Section V.

## II. BLOCKING ARTIFACTS

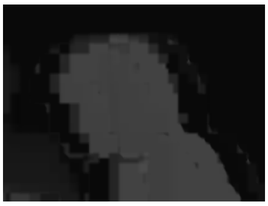
3D video systems use view synthesis methods to create virtual views for 3D displays. In general, multi-view color and their depth data are required for arbitrary viewpoint generation [5, 6]. Since virtual views are created by means of depth data, distortion in depth maps impact the synthesized view quality [7].

The deblocking filter was initially introduced in H.264. As the demand for 3D video has increased, the moving picture experts group (MPEG) has expanded their coding from single view to multiview. They have released a reference software called joint multiview video coding (JMVC) in order to cope with multiview video data. JMVC and H.264 are equipped with the same deblocking filter since JMVC has been designed with roots from H.264.

In JMVC, each frame is coded by 16x16-sized macroblock units. Due to the discontinuation of processing between blocks, artifacts occur. Figure 1 and Figure 2 show coded depth maps and virtual view images of Cafe, respectively with three different QPs used [8]. Noticeably, blocking artifacts around boundaries become much more significant in high QPs. This distortion leads to insufficient qualities of synthesized view images shown in Figure 2. Evidently, performance-wise, the deblocking filter has much room for improvement.

(a) QP 40

(b) QP 45



(c) QP 50

Figure 1. Depth maps of Newspaper

(a) QP 40

(b) QP 45

(c) QP 50

Figure 2. Synthesized views of Newspaper

Figure 3.

## III. Proposed Algorithm

The proposed algorithm comprises three parts: creating binary edge map, applying boundary sharpening filter, and adaptively selecting blocks which to apply the filter depending on macroblock characteristics. Each function is described in the following sub-sections. Figure 3 shows the flowchart of the proposed algorithm. After the conventional JMVC deblocking filter is applied, the proposed algorithm is added for post-processing.

### *A. Binary edge map*

The purpose of binary edge map is to determine which portion contains edge information. As covered in the previous section, our focus lies on blocking artifacts around boundaries. The binary edge map provides the locations of boundaries that need to be filtered. We adopt the Canny method, which is one of most effective edge detection algorithm.

Particularly, the Canny edge detector employs three steps: noise reduction, non-maximum suppression, and hysteresis thresholding. Noise is reduced via a Gaussian filter, blurring the image to effectively find magnitude and orientation of image gradients for the following step. Non-maximum suppression, also known as thinning, checks if the pixel is local maximum along gradient direction to determine whether it is edge or not. Further two thresholds are used; high threshold initiates edge curves while low thresholds continue the curves.

For each frame, we generate a binary edge map via Canny edge detector using reconstructed images as input data. Figure 4 demonstrates a binary edge map of Ballet.

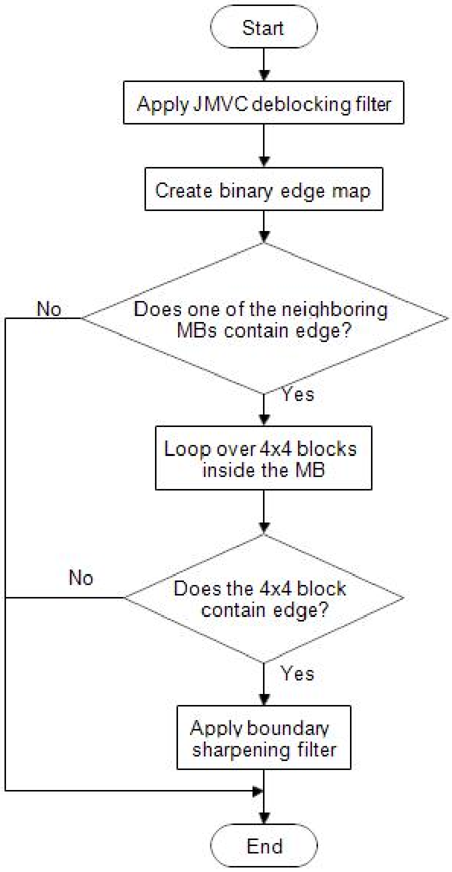



Figure 4. Algorithm flowchart

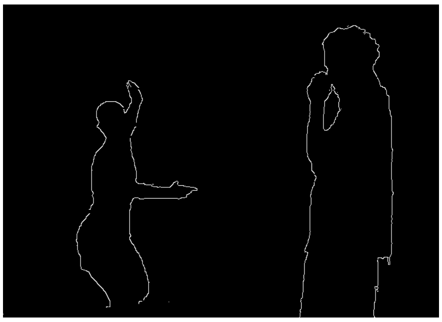


Figure 5. Binary edge map of Ballet via Canny edge detector

Since the binary edge map quality heavily depends on parameters, we start by observing the sequence. We choose a number of objects. Consecutively, two thresholds are properly selected in order to capture the edges only from the objects of interest.

*B. Boundary sharpening filter*

We use a boundary sharpening filter, a modified filter in [3]. The filter defines a window around each pixel and evaluates which pixel within the window should replace the current pixel. Initially, we estimate the number of pixels with distinct intensities. Sequentially, we compute three sub-costs: frequency, similarity, and closeness between the current pixel and pixel within the window. Pixel intensity is the parameter for all sub-costs.

Frequency represents the number of occurrences while similarity means the absolute difference between the parameter intensity and current pixel intensity. The closeness calculates all the Euclidean distance between the current pixel and pixels which possess the parameter intensity value. We divide the sum by the frequency. These factors are represented in the following equations.

$$F(k)=\sum_{i=0}^{n\times n-1}\delta[k-W_{n\times n}(i)] \quad (1)$$

$$S(k)=|I_{cur}-k| \quad (2)$$

$$C(k)=\frac{1}{F(k)}\sum_{i}^{F(k)-1}\sqrt{(x_{cur}-x_i)^2-(y_{cur}-y_i)^2} \quad (3)$$

For each factor, we find the minimum and maximum values to normalize the cost. Considering the cost range, the sub-costs are adjusted as in the following equations.

$$J_F(k)=\frac{F(k)-F(\min)}{F(\max)-F(\min)} \quad (4)$$

$$J_S(k)=\frac{S(\max)-S(k)}{S(\max)-S(\min)} \quad (5)$$

$$J_C(k)=\frac{C(\max)-C(k)}{C(\max)-C(\min)} \quad (6)$$

$J_F(k)$, $J_S(k)$ and $J_C(k)$ denote cost of frequency, similarity, and closeness respectively, for pixel $k$. By independently evaluating the sub-costs' reliability, from our visual judgment, we concluded that frequency was the most crucial factor, while closeness was the least contributing one.

$$J(k)=3\times J_F(k)+2\times J_S(k)+J_C(k) \quad (7)$$

Hence, we assigned weights accordingly as in (7). The intensity $k$ which leads to the least $J(k)$ is selected as the new intensity for the current pixel. Since $k$ exists in the window, the replacement intensity is not a fabricated value, which means noise can be reduced.

*C. Macroblock characteristics*

In order to apply the boundary sharpening filter only at boundaries, we created criteria for macroblocks. We filter the region which satisfied two conditions; at least one of the eight neighboring blocks contain edge, and inside the current block, partitioned 4x4 block should hold edge as well.

To start with, we examine eight neighboring blocks to see whether edge is included in the neighbors or not. If edge cannot be found, we assume there would be very little chance that the current macroblock will contain edge, ending the process.

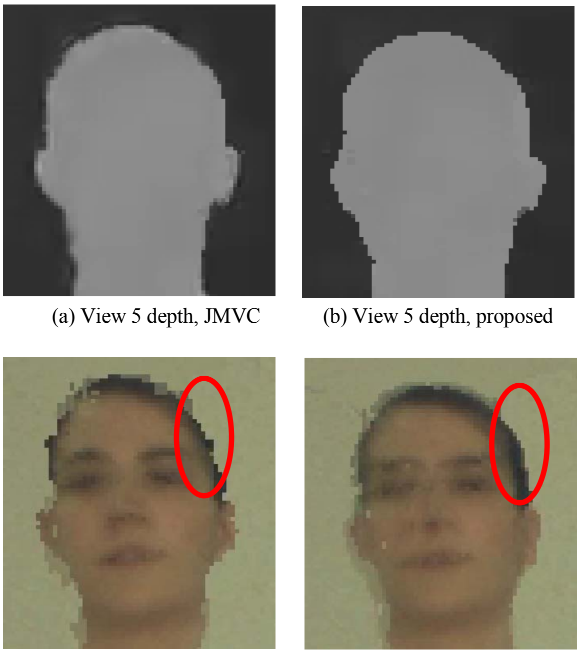
On the contrary, if edge is found in at least one neighbor, then we determine that the current block also holds edge. The goal is to apply boundary sharpening filter only at the boundaries, or in other words, edges. The current block is 16x16-sized, meaning there are 16 blocks within the block which are 4x4-sized. To scan the specific place of the edge, we loop over 4x4-sized blocks. If edge is not contained, we exit the process. Instead, if edge is found, we apply the boundary sharpening filter.

## IV. Experimental Results

We implemented our algorithm on JMVC 8.3 as a post-processing. The Canny thresholds were manually selected for best performance, which is not practical. We plan to solve this problem by selecting an automatic edge detection method in our future work. Table 1 lists the experiment configuration.

TABLE I. EXPERIMENT CONFIGURATION

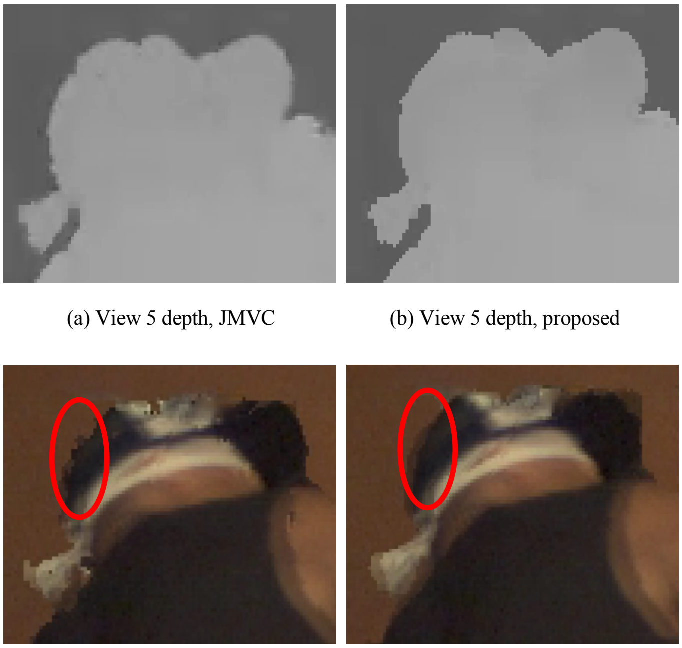
| | |
|---|---|
| **Sequence** | Ballet, Breakdancers |
| **QP** | 32, 37, 42, 47 |
| **Number of frames** | 100 |
| **View synthesis** | Synthesize view 4 using view 3 and view 5 |
| **Canny thresholds** | Ballet: 170, 180<br>Breakdancers: 100, 130 |



(a) View 5 depth, JMVC (b) View 5 depth, proposed

(c) Synthesized, JMVC (d) Synthesized, proposed

Figure 6. Ballet 41st frame, QP 37



(a) View 5 depth, JMVC (b) View 5 depth, proposed

(c) Synthesized, JMVC (d) Synthesized, proposed

Figure 7. Breakdancers 1st frame, QP 32

Figure 5 and Figure 6 show the coded depth maps and synthesized views, respectively. In low QPs, the impact of blocking artifacts is minor. Thus, our algorithm was more effective in higher QPs. For both Figure 5 and Figure 6, in (b), the boundaries in depth maps are significantly sharpened compared to (a). In other words, smoothness was reduced. As a result, much improved synthesized views were generated. The head part is our major concern in both Figure 5 and Figure 6. As covered in the previous section, our focus lies on blocking artifacts around boundaries.

Compared to JMVC, the proposed algorithm produced more clear boundaries in our interested region. In particular, the red circles indicate mostly enhanced areas. Noise around the object boundaries are significantly reduced in our visual assessment.

## V. Conclusions

In this paper, we introduced an efficient method to sharpen object boundaries in depth maps as post-processing to the deblocking filter. The proposed algorithm consists of three parts: binary edge map creation, applying boundary sharpening filter, and adaptive block selection. Initially, we acquire the edge map using the Canny edge detector to and find the location of blocks that need to be altered. Subsequently, we apply a boundary sharpening filter which uses pixel frequency, similarity, and closeness as sub-costs. The filter is only applied to edge-containing 4x4 blocks when their neighboring macroblocks contain edges as well. Experimental results exhibit much improved visual comfort in synthesized images compared to the results made by JMVC. In the future work, we plan to select an automatic edge detection method for flexibility and present objective quality measurement.

## Acknowledgment

This work was supported by Samsung Electronics Co. Ltd.

## 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] C. Dorea and R. L. de Queiroz, “Depth map reconstruction using color-based region merging,” *IEEE International Conference on Image Processing*, Sep. 2011.

[3] K.-J. Oh, A. Vetro, and Y.-S. Ho, “Depth coding using a bourndary reconstruction filter for 3-D video systems,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 21, no. 3, pp. 350-359, March 2011.

[4] 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.

[5] 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.

[6] S. Yea and A. Vetro, "View Synthesis Prediction for Multiview Video Coding," *Signal Processing: Image Communication,* vol. 24, pp. 89-100, Jan. 2009.

[7] C. Lee, B. H. Choi, and Y. S. Ho, "Efficient multiview depth video coding using depth synthesis prediction," *Optical Engineering,* vol. 20, July 2011.

[8] ISO/IEC JTC1/SC29/WG11, "MPEG 3DV EE1 Results on Cafe," *MPEG* M19277, Jan. 2011.