# Depth Coding using Coded Boundary Patterns

Kai-Hsiang Yang and Hsueh-Ming Hang
Department of Electronics Engineering, National Chiao Tung University
Email: khyang225@gmail.com and hmhang@mail.nctu.edu.tw; Tel: +886-35731861

***Abstract* -- The depth information plays an essential role in the virtual-view (or free-viewpoint) 3D video systems. In this paper, we propose a new algorithm to code a depth map for the purpose of virtual view synthesis. The idea is to use H.264/AVC to represent the rough shape (including depth values) of a depth map and then additional information is transmitted to improve the depth values around the object boundaries. The complete encoding and decoding simulation system was built on the H.264/AVC JM 18.0 platform. In our experiments, three tools can be turned on individually and thus four coding modes are defined and tested. Our data show that these proposed tools offer advantages in either coding efficiency or image quality improvement and some tools work best on simple images while the others work best on complex images. With proper parameter setting, the overall quality of virtual view rendering is noticeably improved.**

## I. INTRODUCTION

Three-dimension (3D) video is currently one of the most popular topics in multimedia. Depth perception is one of the most critical elements of 3D video. The recent free-viewpoint (or virtual view) video system wishes to provide views at any admissible viewing angles. The most popular approach is to synthesize new views using the given one or two views together with the depth information. Consequently, the depth information plays a very important role in 3D video system.

The international Moving Picture Experts Group (MPEG) started defining a new standard for 3D video coding. Unlike the ordinary color images, the depth maps are highly correlated spatially except at the object boundaries. Because of the sharp depth value transitions at object boundaries, these complex regions are more sensitive to coding errors and need much more coding bits compared to the other flat regions [1]. Thus, an adequate representation of the depth values near the object boundaries is a critical issue in depth coding. Several methods on depth coding have been published, for example, [1-5].

In this paper, we propose a new model-based algorithm to represent the depth information. Our aim is to provide better image quality in virtual view synthesis. Our proposed algorithm is described in section 2. The experimental results are shown and discussed in section 3. At the end, section 4 contains some final remarks.

This work was supported in part by the NSC, Taiwan under Grants 98-2221-E-009 -076 and by the Aim for the Top University Project of National Chiao Tung University, Taiwan.

## II. PROPOSED ALGORITHM

As discussed earlier, the complex regions are our focus in depth coding. In this section, we will give a step-by-step description of the proposed algorithm. We first give the flow of the proposed algorithm. An illustration of the proposed algorithm is shown in Fig. 1, which is composed of several our designed tools (the right column in blue color). The critical depth information are represented by the coded patterns.

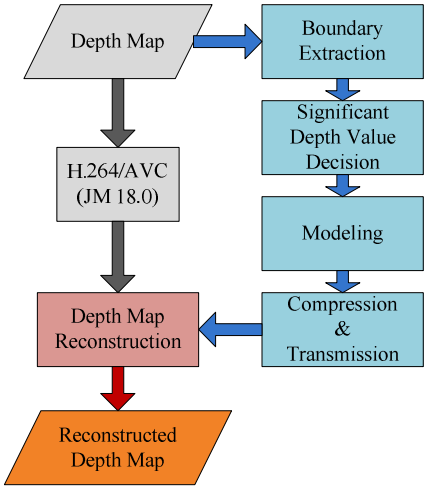


Fig. 1. The flowchart of the proposed algorithm

In the first step, we input the depth map into the codec of H.264/AVC (JM 18.0). At the same time, we extract the boundary information from the input data (the original depth map) as the side information. Second, we propose a criterion to choose the significant depth values we want to transmit. Third, we propose 128 block patterns (3*3) to model those significant depth values we choose in the previous step. After modeling, we encode the model information and compress them. As a result, the model information (also called *side information*) can be transmitted to the decoder more efficiently. After we obtain the decoded depth map from JM 18.0 and the decoded side (model) information, we replace the distorted depth values in the critical regions of the decoded depth map by the received side information. In consequence, we can reconstruct a better quality depth map.

### *A. Boundary Extraction*

The key concept of our proposed algorithm is transmitting the depth information near object boundaries to the receiver with very low distortion. Hence, we need to extract the depth information located on the object boundaries. First,

we use the Sobel operator to detect object boundaries. In order to detect both the horizontal and the vertical edges, we perform the Sobel operator horizontally and vertically and then combine these two detection results into one image. This image plays an important role in our algorithm. We assume the given depth map is perfect although the depth map in experiments are not always so. We adopted a set of 5*5 Sobel operator kernels to extract boundaries [6].

Some nonzero points on the output image of Sobel operation are not the object boundaries of our interests. Ideally, boundaries have one-pixel width. Further processing is performed to extract the so-called "essential" object boundaries from the output image of Sobel operation. We extract the essential boundaries by computing the difference between the depth values of all nonzero pixels and their four closest adjacent pixels. If all the differences are greater than a selected threshold, then we mark the current pixel as the essential boundary

Fig. 2 shows an example ("*Balloons*" Camera 3: Frame 001) of the result of the essential boundary extraction. We can indeed obtain the most essential information of the boundaries in a given depth map.

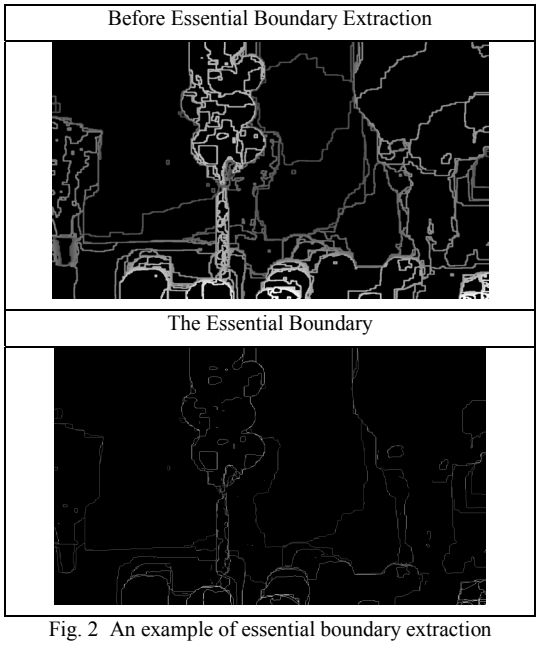

Fig. 2 An example of essential boundary extraction

### B. Boundary Modeling by Block Patterns

We next approximate the essential depth information by a "model". In the first step, we propose a procedure, Significant Depth Value Decision (SDVD), to select the *significant depth values* next to the essential boundaries. Then, we represent the neighborhood areas of essential boundaries and their associated significant values by "*block patterns*" we propose.

A depth map is portioned into 3*3 blocks. This size is chosen as a trade-off between bit rate and boundary approximation accuracy. If the width and/or the height of image is not a multiple of 3, we drop the left-most and/or the right-most columns or the top and the bottom rows so that the 3*3 block partition is plausible. We encode all the blocks but only the blocks containing essential boundaries are critical. We use the following procedure to reduce the number of pixel values to be transmitted. Generally, we pick up only one value for each 3*3 block by using SDVD composed of the four criterions below.

(1) If this block does not contain essential boundaries, it is labeled as "*skip_mode*"; its *significant depth value* is 0.

(2) If there is only one pixel with nonzero depth value in the current 3*3 block, we set the *significant depth value* of the current 3*3 block still to 0.

(3) If there exists just one depth value "$DV_1$" with the highest occurring frequency in the current 3*3 block, we set the *significant depth value* to "$DV_1$."

(4) If there are more than one depth values with highest frequency, the depth value with the highest occurring frequency and the largest magnitude is the *significant depth value*.

For each 3*3 block, the *pattern pixel* is set to "1" if the depth value is equal to the significant depth value decided by the procedure mentioned above. Otherwise, it is "0".

Then, we propose 128 binary *patterns* to represent the 3*3 binary blocks. The 128 model patterns are proposed based on collected depth map statistics. The distribution of pixels with binary value "1" is mostly along the shape of foreground objects. Thus, we construct the model patterns only with good pixel connectivity. We discard some models that are too complex or too rare. In our test data, 90% or higher area in our essential boundary images can be modeled by the proposed 3*3 block patterns. The first 8 groups of patterns are shown in Fig. 3. There are in total 25 groups of patterns. Each group contains several patterns. For example, Group 1 contains 6 patterns. The dark pixel represents pattern pixel value "1" and white pixel represents "0". The 3-pixel "1" stripe may have 3 possible locations (left, middle and right) located vertically, which gives 3 patters. It may lie horizontally at 3 possible locations, which gives another 3 patters. The complete 128 patterns can be found in [6].

### C. Data Transmission

*Syntax*

The syntax of representing a model pattern is composed of an *indicator*, a *block pattern index*, and a *significant depth value*. Note that each slice corresponds to one 3*3 block. The beginning of a slice is an indicator, which is used to indicate whether the current 3*3 block is the "*skip_mode*" or not. If it is a "*skip_mode*" block, it means that all the pattern pixels have value "0" and an indicator of "0" is assigned to this block. Obviously, there is no need to additional information after the indicator "0." Indicator "1" means that the current 3*3 block is not the "*skip_mode*." As a consequence, we append the block pattern index and

the significant depth value for the current 3*3 block. The slice syntax is shown below.

*Syntax* : *Indicator* + *Block Pattern Index* + *Significant Depth Value*

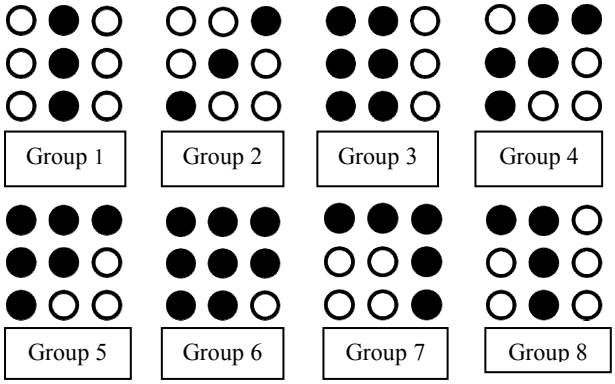


Fig. 3. The first 8 groups of model patterns

*Compression*

In fact, most 3*3 blocks after the modeling step turn out to be the "*skip_mode*." According to [4], there are only about 20% regions with sharp depth value transitions. In other words, most blocks are "*skip_mode*." Based on our design concept, only the 3*3 blocks containing sharp depth value transitions are coded by the non-skip modes. Thus, the indicator "0" often appears consecutively. In order to reduce the bitrate needed to transmit the side information of the indicators, we use the run length coding technique to compress the indicator data. We compute the runs row by row. Then, we use the Huffman coding to compress the run length codes. Limited by the space, please refer to [6] for more details of this Huffman code.

### *D. Down/Up Sampling*

One noticeable characteristic of a depth map is that most regions are generally flat or homogeneous. Therefore, we add the down/up sampling step into our algorithm. We only adopt down/up sampling technique in the path of main information processing (H.264/AVC coding). The sampling factor is set to 2. Thus, the bitrate used for H.264 coding is reduced significantly. We upsample the decompressed depth map and resize it to the original size. Note that the down/up sampling technique is an additional option.

### *E. Compressed Depth Map Enhancement*

At the decoder, we first receive the depth map compressed by JM 18.0. Then, we receive the information of 3*3 model patterns and the *significant depth value* associated with every 3*3 block by decoding the side information bit stream. This portion of information will be combined with AVC-coded depth map to reconstruct the complete depth map. The flowchart of depth map reconstruction is shown in Fig. 4.

The side information provides more accurate depth map is called the "Enhancement Layer (EL)". These pixels locate typically around the object boundaries. Then, we use EL to improve the AVC-compressed depth map and generate the reconstructed depth map by using "Boundary Enhancement (BE)" or "Enhanced Region Expansion (ERE)" processes.

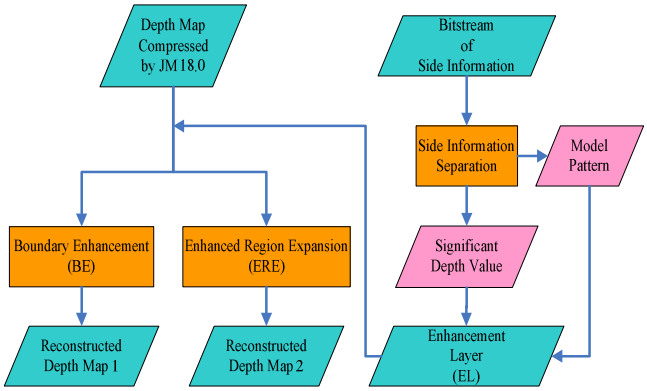


Fig. 4 The flow chart of depth map reconstruction

*Boundary Enhancement* (BE)

For each pixel in EL, we check whether its pixel value is equal to zero or not. If this EL pixel value is non-zero, it means that the pixel value with the same location in the depth map compressed by H.264/AVC should be updated to the *significant depth value*. The *significant depth value* is extracted from the original depth map. Consequently, the significant depth value is equal to the pixel value of the original depth map.

*Enhanced Region Expansion* (ERE)

In order to enlarge the enhanced region, two additional operations, "Dilation" and "Substitution Decision", are proposed in ERE. First, the dilation operation is applied to EL. Then, the nonzero EL pixels substitute for the pixels near the object boundaries on the AVC-compressed depth map if the following criterion is met. The substitution is executed if the difference between the depth value on EL and that on the compressed depth map is smaller than or equal to a pre-chosen threshold value. The reason of adopting this criterion is to avoid substituting the foreground pixels by the nearby background pixels.

## III. EXPERIMENTS

In the experiments, we turn on and off the proposed tools to test their performance. In total, there are 4 processing modes. Mode 1 enables "BE" to enhance the compressed depth map and Mode 2 enables "ERE" (BE together with expansion). In Fig.4, Mode 1 generates Depth Map 1 and Mode 2 generates Depth Map 2. Both Mode 1 and Mode 2 do not run the down/up sampling tools. Then, Mode 3 and Mode 4 are the counterparts of Mode 1 and Mode 2 but accompany with the down/up sampling tool.

As shown in the basic flow of our proposed algorithm (Fig. 1), the JM 18.0 with five QP values: 16, 23, 30, 37, and 44 is in use. At the end of our proposed algorithm, the reconstructed depth map is produced. However, our focus is the quality of the rendered virtual view. We use the View Synthesis Reference Software (VSRS) provided by MPEG

to synthesize the virtual view in the middle of two reference views (2-view scenario). The test sequences are "Balloons", "Newspaper" and "Lovebird1" (resolution: 1024*768). The target views for synthesis are: View_4, View_5 and View_7, respectively. To increase the image database, we pick up three frames (frames 1, 41, and 81) from each sequence as test images.

### A. *Objective Quality*

In measuring the objective quality of virtual view rendering, we adopt the conventional Peak Signal-to-Noise Ratio (PSNR) as the metric. Note that the image synthesized using the *uncompressed* depth maps is taken as the ground truth for our experiments. The following figures show the R-D curves of two test sequences. The target we compare with is the JM 18.0 only scheme in coding the depth maps.

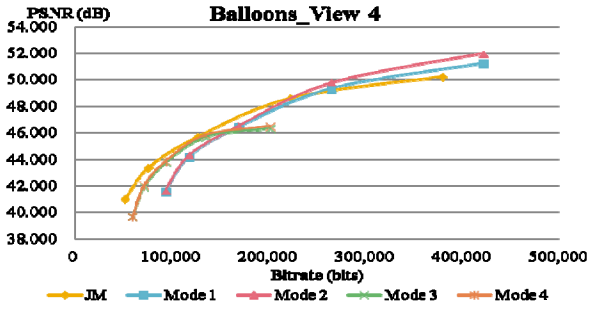


Figure 5 R-D curves using different modes (Balloons)

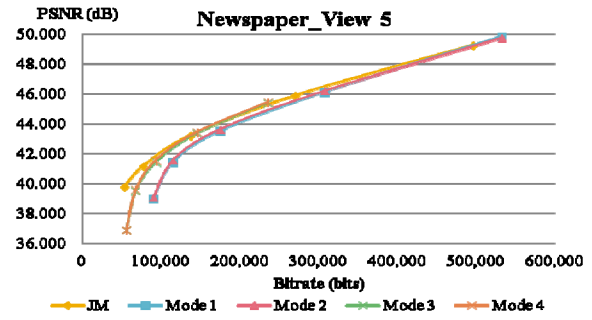


Figure 6 R-D curves with different modes (Newspaper)

We can see that the performance of modes in our proposed algorithm is highly dependent on the complexity of the original depth map. For simple depth maps (e.g., Lovebird1), Mode 3 and Mode 4 perform better than H.264/AVC because the down-sampling distortion of depth map can be recovered according to the information within EL we extract and transmit. On the contrary, Mode 1 and Mode 2 are more suitable for complex depth maps (e.g., Balloons) because the boundary area (in percentage) is higher. The distortion due to down-sampling is more serious and less ratio can be recovered by the transmitted information in comparison with simpler depth map case. However, the pattern coding bit rate is high and thus the R-D advantage shows up only at the high bit rates. There is certainly room for improvement for, say, lowing the side information bits.

### B. *Subjective Quality*

In addition to PSNR, we like to compare the subjective quality of the synthesized images using different tools. We will show the synthesized images at comparable bitrates using JM 18.0, Mode 2 and Mode 4. We found some visible improvements around the object boundaries such as the gray clothes and the leaf in Newspaper, and the man's hair and clothes in "Lovebird1" as shown in [6]. Note that the image synthesized by using the depth maps enhanced in Mode 4 may induce new artifacts due to down/up sampling technique [6].

## IV. CONCLUSIONS

In this study, we propose a new method to enhance the compressed depth map by sending additional side information. The proposed algorithm corrects the depth values near the foreground object boundaries. Therefore, the depth map compressed by JM 18.0 can be enhanced by the additional side information. The main idea is to represent the depth information around object boundaries using pre-selected 3*3 patterns. Simulations show that our proposed scheme indeed provides some visual improvements.

## V. REFERENCES


[1] K. J. Oh, A. Vetro, and Y. S. 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, pp.350-359, Mar. 2011.

[2] D. V. S. X. De Silva, W. A. C. Fernando, and S. T. Worrall, "Intra Mode Selection Method for Depth Maps of 3D Video Based on Rendering Distortion Modeling, " *IEEE Tran. On Consumer Electronics, vol. 56, issue 4,* Nov. 2010.

[3] Q. W. Zhang, P. An, Y. Zhang, Q. Zhang, and Z. Y. Zhang, "Reduced Resolution Depth Compression for Multiview Video plus Depth Coding," *IEEE International Conference on Signal Processing,* 2010.

[4] Y. Morvan, D. Farin, and P. H. N. de With, "Depth-image compression based on an R-D optimized quadtree decomposition for the transmission of multiview images," in *Proc. ICIP*, vol. 5, pp. 105-108, Oct. 2007.

[5] P. Merkle, et al., "The effect of depth compression on multiview rendering quality," in *Proc. 3DTV Conf.*, pp. 245-246, May 2008.

[6] K.-H. Yang, *Depth Coding for 3D Video*, MS Thesis, National Chiao-Tung University, Taiwan, March, 2012.