THE EFFECT OF DEPTH COMPRESSION ON MULTIVIEW RENDERING QUALITY

P. Merkle, Y. Morvan, A. Smolic, D. Farin[1], K. Müller, P.H.N. de With*, and T. Wiegand*

Fraunhofer Institute for Telecommunications
Heinrich-Hertz-Institut
Image Processing Department
Einsteinufer 37, 10587 Berlin, Germany
{merkle/smolic/kmueller/wiegand}@hhi.de

* Eindhoven University of Technology
Signal Processing Systems
P.O. Box 513, 5600 MB Eindhoven
The Netherlands
{y.morvan/P.H.N.de.With}@tue.nl

ABSTRACT

This paper presents a comparative study on different techniques for depth-image compression and its implications on the quality of multiview video plus depth virtual view rendering. A novel coding algorithm for depth images that concentrates on their special characteristics, namely smooth regions delineated by sharp edges, is compared to H.264 intra-coding with depth-images. These two coding techniques are evaluated in the context of multiview video plus depth representations, where depth information is used to render virtual intermediate camera views of the scene. Therefore it is important to evaluate the influence of depth-image coding artifacts on the quality of rendered virtual views. The results of this evaluation show, that the coding algorithm specialized on the characteristics of depth images outperforms H.264 intra-coding, although its RD-performance is worse.

Index Terms— Depth compression, multiview video plus depth, free viewpoint video, image coding, view rendering.

1. INTRODUCTION

Multiview video (MVV) representations enable new applications such as free viewpoint video (FVV) and 3D television (3DTV) [1]. The main characteristic of 3DTV is to offer a 3D depth impression of the observed scenery. FVV on the other hand is characterized by providing the user the ability to interactively select an arbitrary viewpoint in the video scene as known from computer graphics. Both technologies do not exclude each other, but rather can be combined into one system. A common characteristic of such technologies is that they use MVV data, where a real world scene is recorded by multiple synchronized cameras.

A popular format for 3DTV uses a conventional color video and an associated per pixel depth map and MPEG specified a standard for efficient compression and transmission. In the context of MVV this format is combined with multiview video to the multiview video + depth (MVD) format, consisting of multiple color videos with associated multiple depth data of one scene. Since MVD representations cause a vast amount of data to be stored or transmitted to the user, efficient compression techniques are essential for realizing such applications. Previous work presented various solutions for multiview video coding (MVC), mostly based on H.264 with combined temporal and inter-view prediction, as well as different approaches for depth image coding, like transform- or wavelet-based depth compression.

As a first step towards standardization of technologies for 3DTV and FVV applications, a new standard addressing algorithms for compression, transmission and rendering of multiview video data is currently developed by the Joint Video Team (JVT) of VCEG and MPEG, which is scheduled to be finalized in early 2008. As a second step, MPEG started an Ad Hoc Group on Free Viewpoint Television recently, focusing on FVV and 3DTV systems from a normative point of view, including representation, generation, processing, coding and rendering of MVD format data.

This paper is organized as follows. Section 2 is about depth-image compression. The novel Platelet-based depth coding algorithm is introduced and the coding results are presented. Section 3 is about virtual view rendering, especially its application to evaluate the effects of depth-image compression, and presents the experimental results. Finally, the paper concludes with Section 4.

2. COMPRESSION OF DEPTH IMAGES

The MVD format consists of several camera sequences of color texture images and associated per sample depth images or depth maps as illustrated in Fig. 1. Depth images are a 2D representation of the 3D surface of the scene. Therefore the depth range is restricted to a range in between two extremes z_{near} and z_{far}, indicating the minimum and maximum distance of the corresponding 3D point from the camera respectively. By quantizing the values in this range, the depth image in Fig. 1 is specified, resulting in a grey scale image. A sequence of such depth images can be converted into a YUV 4:0:0 format video signal and compressed by any state-of-the-art video codec. Since depth images represent the scene depth their characteristics differ from texture images. Encoding depth images with video codecs that are highly optimized to the

[1] Dirk S. Farin, was with Eindhoven University of Technology, Eindhoven, The Netherlands. He is now with the Robert Bosch GmbH, Hildesheim, Germany.

Fig. 1. Example for the MVD format with texture image (left) and corresponding depth-image (right).

statistical properties and human perception of color or texture video sequences, might be efficient but results in disturbing artifacts. Therefore novel algorithms for depth image compression are developed, that are adapted to their special characteristics, namely smooth regions delineated by sharp edges. In order to evaluate the properties of the Platelet-based depth image coding algorithm presented in Section 2.1, the coding results in Section 2.2 are compared to H.264 Intra-coding as a reference.

2.1. Platelet-based Depth Coding

We present a novel approach for depth-image coding that is based on piecewise-linear functions [2]. The idea followed is to approximate the image content using modeling functions. In our framework, we use two classes of modeling functions: a class of piecewise-constant functions and a class of piecewise-linear functions. First, regions of constant depth show smooth regions in the depth image. These smooth regions can therefore be approximated by a piecewise-constant function. Second, planar surfaces of the scene like the ground plane and walls of e.g. a room, appear as regions of gradually changing gray levels in the depth image. Hence, such a planar region can be approximated by a single linear function. To specify the 2D-support of the modeling functions in the image, we employ a quadtree decomposition that hierarchically divides the image into blocks, i.e. nodes of different size. In some cases, the depth image within one block can be approximated with one modeling function. If no suitable approximation can be determined for the block, it is subdivided into four smaller blocks. To prevent that many small blocks are required along a discontinuity, we divide the block into two regions separated by a straight line. Each of these two regions is coded with an independent function. Consequently, the coding algorithm chooses between four modeling functions for each leaf in the quadtree:

- Modeling function f_1: Approximate the block content with a constant function.
- Modeling function f_2: Approximate the block content with a linear function;
- Modeling function f_3: Subdivide the block into two regions separated by a straight line and approximate each region with a piecewise-constant function (a *wedgelet* function);
- Modeling function f_4: Subdivide the block into two regions separated by a straight line and approximate each region with a piecewise-linear function (a *platelet* function);

Fig. 2. Example of a quadtree decomposition. Each block, i.e. node, of the quadtree is approximated by one modeling function.

Figure 2 shows a quadtree decomposition of a depth image where each node is approximated by one of the modeling function f_1, f_2, f_3, or f_4.

The decision for each modeling function is based on a rate-distortion decision criterion that we now detail. Considering our lossy encoder/decoder framework, our aim is to optimize the compression of a given image to satisfy a Rate-Distortion (R-D) constraint. In our practical case, there are three parameters that influence this trade-off: (1) the selection of modeling functions, (2) the quadtree decomposition and (3) the quantization step-size of the modeling-function coefficients. Thus, the problem statement is to adjust each of the previous parameters such that the objective R-D constraint is satisfied.

To optimize these three parameters in an R-D sense, the adopted approach is to define a cost function that combines both rate R and distortion D of the image i. Typically, the Lagrangian cost function

$$J(R_i) = D_i(R_i) + \lambda R_i$$

is used, where R_i and D_i represent rate and distortion of the image, respectively, and λ is a weighting factor that controls the rate-distortion trade-off. Using the above Lagrangian cost function principle, the algorithm successively performs three independent parameters optimizations: (1) an independent selection of modeling functions, (2) a quadtree decomposition optimization and (3) the quantizer step-size selection. Let us now detail these three parameters optimization procedures.

(1) Modeling function selection. First, we assume that an optimal quadtree segmentation and quantizer step-size is provided. Since the rate and distortion are additive functions over all blocks, an independent optimization can be performed within the blocks. Therefore, for each block, the algorithm selects the modeling function that leads to the minimum coding cost of the Lagrangian cost function.

(2) Quadtree decomposition. To obtain an optimal quadtree decomposition of the image, a well-known approach is to perform a so-called *bottom-up* tree-pruning technique. The guiding principle is to parse the initial full tree from bottom to top and recursively prune nodes (i.e. merge blocks) of the tree according to a decision criterion. Similarly to the modeling function selection procedure, the decision criterion is based on a Lagrangian cost function that merges (prunes) four children nodes whenever the sum of the four coding costs is higher than the coding cost of the parent node.

(3) Quantizer selection. Quantizer selection is the problem of selecting a (scalar) quantizer that corresponds to the quadtree decomposition of the depth image. For example, it is not appropriate to combine a coarse quantization with a fine quadtree decomposition. To properly quantize modeling

functions coefficients, we re-use the principle of the Lagrangian cost function and select the quantizer that minimizes the Lagrangian coding cost of the image.

As a final step, to reduce the redundancy between nodes in the quadtree, a predictive coding technique is introduced. In more detail, this means that we decorrelate the remaining dependencies between each block to further enhance coding efficiency. For more details related to the depth coding algorithm, we refer to our paper [2].

2.2. Coding Results

We conducted the coding experiments for two MVD test data sets named "Breakdancers" and "Ballet", both consisting of eight linearly arranged camera views. From both test data sets the first depth image of the sequence was compressed for each camera view at different qualities. First the depth images were encoded and decoded with the Platelet-based coder presented in the last section, revealing that the proposed algorithm can approximate large smooth areas as well as sharp edges with a single node. Second, the same depth images were encoded and decoded with a standard conforming H.264 coder as I frames. The rate-distortion results of these coding experiments are shown in Fig. 3. Except for very low bitrates H.264 intra-coding outperforms the Platelet-based coding approach in terms of PSNR performance. In addition to these objective results Fig. 4 shows examples for the subjective quality at a very low bit rate, highlighting the typical coding artifacts for the two evaluated coding algorithms.

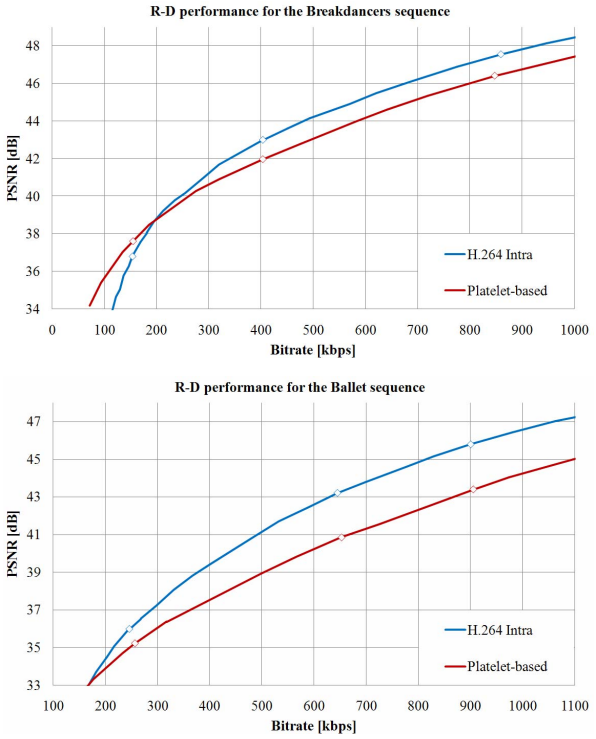

Fig. 3. Rate-distortion curves for "Breakdancers" (top) and "Ballet" (bottom) depth images, comparing Platelet-based and H.264 Intra coding.

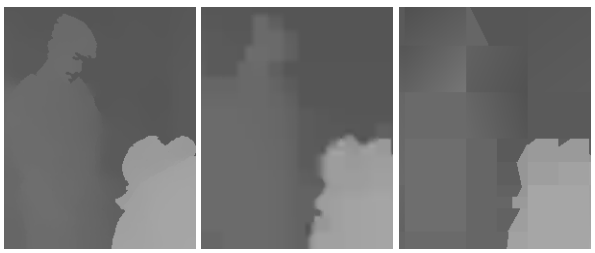

Fig. 4. Examples for coding artifacts: Original uncompressed (left), H.264 Intra (middle) and Platelet-based (right).

3. VIRTUAL VIEW RENDERING

The main advantage of MVD representations in contrast to MVV is that due to the availability of depth information 3D rendering based applications like FVV can be realized. Multiview video + depth data together with camera geometry provides the possibility to synthesize or render arbitrary intermediate views from a 3D representation of the scene.

Virtual view rendering uses pairs of neighboring original camera views to render arbitrary virtual views on a specified camera path between them. The relation between points in 3D scene space and the values in a depth image is defined by the projection matrix and the quantization function, allowing for projecting and unprojecting depth data. First the depth images are unprojected, resulting in a colored 3D particle cloud for each original camera. Next the projection matrix of a virtual camera is calculated from the two original cameras projection matrices by spherical linear interpolation (SLERP) and linear interpolation (LERP). These methods originate from computer graphics in the context of quaternion interpolation. Now the two original camera's colored point clouds can be projected into the virtual camera view, as depicted in Fig. 5 left and right top. In a next step these two rendered color images are blended into each other, using the information from the rendered depth maps as well as texture weighting according to the virtual camera's position relative to the original cameras, as depicted in the middle of Fig. 5.

3.1. Evaluation of Compression Effects

We now describe a method for analyzing the impact of depth compression on the quality of rendered virtual views [3]. For this purpose the rendering technique described in the previous section is first applied to uncompressed input data, resulting in a reference output image. In a second step the same virtual view is rendered by using compressed input data. The impact of coding artifacts on the quality of rendered views can now be

Fig. 5. Rendering of a virtual intermediate view from the projected MVD data of two neighboring original cameras.

analyzed by comparing the reference picture with the one rendered with compressed input data in terms of objective and subjective quality.

3.2. Rendering Results

In the case of depth image coding we conducted these experiments on the quality of rendered virtual views in order to identify which coding algorithm performs better. For this purpose we rendered a series of virtual views along the camera path, using compressed depth images and uncompressed color textures, to determine the PSNR versus the reference. By doing so for the depth images from R-D points, where both coding algorithms produce the same bit rate (see markers in Fig. 3), we obtain two PSNR values, indicating which coding algorithm performs better for low, middle, and high bitrates. Fig. 6 and Tab. 1 show, that in most cases Platelet-based coded depth images achieve a equal or better rendering quality, although their coding quality is worse than H.264 Intra. In return this means that Platelet-based coding would clearly outperform H.264 intra-coding in virtual view rendering quality, if comparing compressed depth images with equal PSNR.

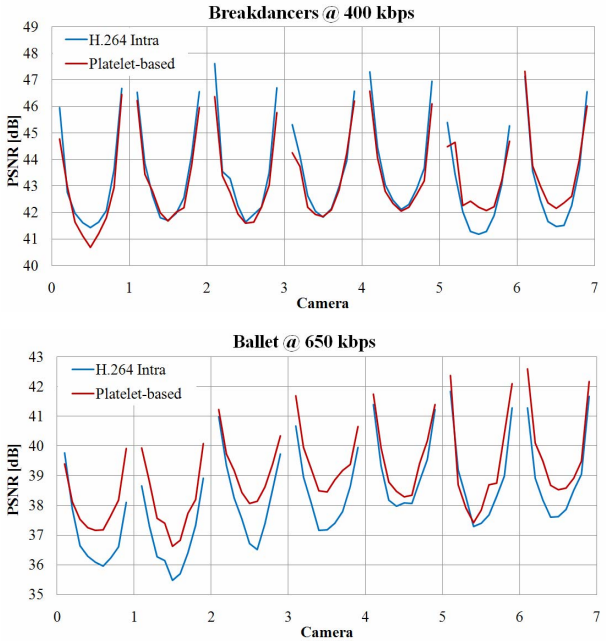

Fig. 6. Quality of rendered virtual views along the camera path at a middle depth bit rate for "Breakdancers" (top) and "Ballet" (bottom) test data.

Breakdancers			
Bitrate [kbps]	150	400	850
ΔPSNR [dB] Coding	0.80	-1.04	-1.14
ΔPSNR [dB] Rendering	0.90	-0.13	-1.04
Ballet			
Bitrate [kbps]	250	650	900
ΔPSNR [dB] Coding	-0.76	-2.35	-2.40
ΔPSNR [dB] Rendering	1.38	0.84	-0.35

Tab. 1. Comparison of ΔPSNR from coding and virtual view rendering for low, middle and high bit rates, where ΔPSNR is the average difference in PSNR results between H.264 Intra and Platelet-based depth coding.

4. CONCLUSIONS

We have presented a comparative study on depth-image compression, intended to answer the question, which effect different types of coding artifacts have on the quality of virtual view rendering for MVD data. The Platelet-based depth-image coding algorithm was introduced, separating continuous regions by straight lines along their boundaries by modeling them with piecewise-constant or -linear functions. A comparison of the R-D performance turned out that it is being outperformed by H.264 Intra-coding, which is optimized for bit rate efficiency to a great extent. Since depth image compression only has an impact on the quality of rendered virtual camera views in a MVD application scenario, we presented a technique for rendering and analyzing virtual views from compressed and uncompressed MVD data. The results indicate that a worse coding PSNR does not imply a worse rendering PSNR for Platelet-based depth coding, leading to the conclusion that this depth coding approach enables higher rendering quality than H.264 coding, because depth discontinuities are better preserved. Consequently the development of advanced algorithms for MVD coding needs to optimize the R-D performance with respect to not only the distortion of the original camera view, but also the distortion of rendered intermediate views. This requires future research to address joint compression and rendering algorithms as well as appropriate quality metrics.

6. ACKNOWLEDGMENTS

This work is supported by European Union 6th Framework Program under Grant No. FP6-511568 (3DTV NoE Project). We would like to thank the Interactive Visual Media Group of Microsoft Research for providing the "Ballet" and "Breakdancers" data sets.

6. REFERENCES

[1] H. Ozaktas, and L. Onural, *Three-Dimensional Television: Capture, Transmission, and Display*, Springer, Heidelberg, December 2007.

[2] 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", *ICIP 2007, IEEE International Conference on Image Processing*, San Antonio, TX, USA, September 2007.

[3] P. Merkle, A. Smolic, K. Müller, and T. Wiegand, "Multi-view Video Plus Depth Representation and Coding", *ICIP 2007, IEEE International Conference on Image Processing*, San Antonio, TX, USA, September 2007.