Hybrid Lossless-Lossy Compression for Real-Time Depth-Sensor Streams in 3D Telepresence Applications

Yunpeng Liu[1,2], Stephan Beck[1], Renfang Wang[2], Jin Li[2], Huixia Xu[2], Shijie Yao[2], Xiaopeng Tong[2], and Bernd Froehlich[1]

[1] Virtual Reality Systems Group, Bauhaus-Universität Weimar, Weimar, Germany
{yunpeng.liu,stephan.beck,bernd.froehlich}@uni-weimar.de
[2] College of Computer Science and Information Technology, Zhejiang Wanli University, Ningbo, China
{wangrenfang,lijin,xuhuixia,yaoshijie,tongxiaopeng}@zwu.edu.cn

Abstract. We developed and evaluated different schemes for the real-time compression of multiple depth image streams. Our analysis suggests that a hybrid lossless-lossy compression approach provides a good tradeoff between quality and compression ratio. Lossless compression based on run length encoding is used to preserve the information of the highest bits of the depth image pixels. The lowest 10-bits of a depth pixel value are directly encoded in the Y channel of a YUV image and encoded by a x264 codec. Our experiments show that the proposed method can encode and decode multiple depth image streams in less than 12 ms on average. Depending on the compression level, which can be adjusted during application runtime, we are able to achieve a compression ratio of about 4:1 to 20:1. Initial results indicate that the quality for 3D reconstructions is almost indistinguishable from the original for a compression ratio of up to 10:1.

Keywords: Depth map · Compression · x264 · 3D Tele-immersion

1 Introduction

Color and depth (RGBD) sensors, such as the Microsoft Kinect, are used in many research areas. In 2013, Beck et al. [3] presented an immersive group-to-group telepresence system which is based on a cluster of RGBD-sensors that continuously captures the participants. The color and depth image streams are then sent to a remote site over a network connection where they are used for the real-time reconstruction of the captured participants. Due to the large amount of image data that has to be transmitted, the network bandwidth can become a bottleneck. E.g., a setup of five Kinect V1 sensors running at 30 Hz requires a network connection of 2.46 GBit/s. While this might be feasible with high-speed local area networks, the bandwidth of internet connections is typically in the range of 10 to 100 Mbit/s. To reduce the amount of data that has to be transmitted, Beck et al. [3] already suggested the usage of a DXT compression scheme for the color images which compresses the color data by a fixed ratio of 6:1. However, the compression of depth streams was not addressed at all.

Our hybrid lossless-lossy compression method offers different levels of compression quality and compression ratios which can be adjusted during application runtime for adapting to the available bandwidth. The main contributions of our work are:

– Optimal bit assignment from depth values to YUV channels including a lossless encoding mode for the highest bits of the depth values.
– Optimal setting scheme of x264-based encoding parameters for depth maps.
– Efficient encoding and decoding implementation for multiple depth image streams in less than 12ms on average for compression ratios between 4:1 to 20:1.

The integration of our hybrid compression scheme into our immersive telepresence application shows that the quality for 3D reconstructions is almost indistinguishable from the original for a compression ratio of up to 10:1.

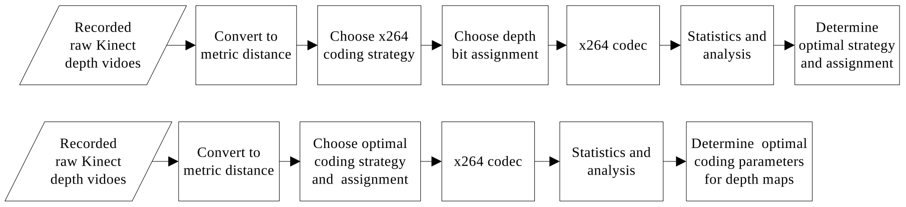
2 Related Work

The development of codecs for depth image compression has become an important research topic in recent years. In the context of 3D reconstruction, a class of methods operate on 3D meshes or point clouds [13, 2, 4, 14, 17]. However, the processing costs for 3D compression are typically much higher than for 2D image compression, and, the benefits of segmentation and prediction cannot be used effectively. In addition, geometry-based methods need to maintain special data structures which often rule out high compression ratios.

From another point of view, depth images are ”special” gray images, which contain object edges (high frequency areas) and smooth surfaces (low frequency areas). Therefore depth image streams can be compressed by international standard [16, 18, 20, 12, 19, 11, 6, 8, 23, 22, 15] or non-standard [9, 5, 10, 1] video encoding algorithms. Non-standard encoding algorithms focus on retaining the integrity and quality of high frequency areas. On the other side, the compression ratio of above algorithms is not very high and most schemes are not compatible with color video compression.

Pece et al. [16] reported initial results to adapt standard video codec algorithms for depth images. However the error, which is introduced by the compression, is comparably high, even for lower compression ratios. This was mainly caused by the loss of the higher bits of the depth image’s pixels.

More recently, the High Efficiency Video Coding (HEVC) standard was introduced and investigated by many researchers [18, 20, 12, 19, 6, 11, 15]. However, the standardization of 3D Video Coding Extension Development of HEVC is still under development and many parts are in the experimental stage. Besides HEVC, other approaches like JMkta or JM based H.264/AVC depth map coding [8, 23, 22] have been proposed. However, neither the JMkta- nor the JM-reference software can meet real-time application requirements. Our approach is partially inspired by the work of Pece et al. [16]. We therefore further investigated and analyzed the depth bit assignment schemes in order to deduce optimal settings for standard real-time video coding algorithms. In addition, our proposed method uses a hybrid lossless-lossy compression approach, which provides a good tradeoff between quality, processing time and compression ratio.



Recorded raw Kinect depth vidoes
Convert to metric distance
Choose x264 coding strategy
Choose depth bit assignment
x264 codec
Statistics and analysis
Determine optimal strategy and assignment
Recorded raw Kinect depth vidoes
Convert to metric distance
Choose optimal coding strategy and assignment
x264 codec
Statistics and analysis
Determine optimal coding parameters for depth maps

Fig. 1: Pipeline for the analysis and the determination of optimal depth-bit assignment (upper row) and of optimal encoding configurations (lower row).

3 Compression Approach

3.1 System Overview

Our approach is based on the optimal determination of many available encoding parameters. In order to deduce the optimal settings, we recorded a sequence of multiple depth image streams which we used in our analysis. The precision of the depth images is clamped to 12 bit and converted to millimeter. The resulting depth-range of 4.000 mm is sufficient for most practical scenarios. In our analysis, we first investigate several schemes to assign 12 bit depth pixels to YUV-channels. Second, we evaluate different parameters which are used to configure the x246-encoder. Figure 1 gives an overview of our experimental setup. Our analysis is then used to obtain an optimal set of encoding parameters to provide several levels of quality and compression ratios. After the identification of the optimal parameter set we integrated the encoder as well as the decoder into a prototype pipeline for real-time 3D reconstruction (c.f. Figure 2). Based on this real-time 3D reconstruction pipeline, we were then able to perform a quantitative and qualitative evaluation of our proposed depth compression method.

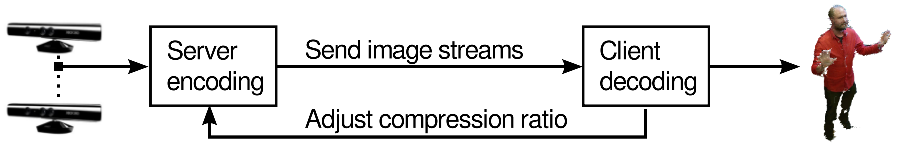


Fig. 2: Overview of the processing and streaming pipeline. Multiple RGB-sensors are connected to a server which encodes the image streams. The client receives the image streams and performs the decoding as well as the 3D-reconstruction.

3.2 Analysis of Depth-bit Assignment

The x264-codec family provides three different YUV color spaces: 4:2:0, 4:2:2 and 4:4:4. In addition the x264-library can be compiled with an internal precision of 8 bit or 10 bit. In our analysis we tested both precisions. We assign the highest 8 bit/10 bit of a

12 bit depth image's pixel to Y space. The remaining lower bits can be assigned to the UV space using 4 different methods: *Method 1* assigns the remaining lower bits to UV space in byte order. This can be performed for all color spaces (4:2:0, 4:2:2 and 4:4:4). For color spaces 4:2:2 and 4:4:4, *Method 2* assigns the lowest bits to the corresponding lower bits of the U space, while the unassigned space of the byte is set to zero. For color space 4:4:4, *Method 3* interleaves the lowest bits to the corresponding lowest bit of U space and V space, the unassigned space is set to zero. *Method 4* assigns the remaining lower bits of the depth pixel to the lowest bit position in the UV space.

On the one hand, the lower bits of the depth pixels tend to contain noise. As a result, the inter frame prediction that is used in x264-encoding is limited. Therefore, a simple option is to discard the lower 2 bits of the depth pixels. Of course, this will introduce a depth error of 3 mm in the worst case.

On the other hand, the higher bits of the depth pixels are very important for an accurate 3D reconstruction. We therefore investigated a hybrid lossless-lossy compression scheme. The main idea of our hybrid encoding scheme is to encode the highest 2 bits of the depth pixels using a lossless compression algorithm and to assign the lower 10 bits in the Y channel of a YUV image for x264 encoding. For lossless encoding we evaluated the three most commonly used algorithms: Huffman, LZW and run length (RLE). We achieved the best compression ratio and encoding performance with RLE.

3.3 Optimal x264 Encoding Parameters for Depth Image Streams

After the analysis of the depth-bit assignment methods, we had to find the optimal parameter settings for x264-encoding. Because our evaluation revealed that an internal precision of 8 bit results in very poor quality (c.f. Table 1), we focus on the 10 bit version of x264 in the following. First, we decided to set the global encoding parameters to constants: *profile=high10*, *csp=I420*, *framerate=30*, *tune=zerolatency*. Next, we analyzed further encoding parameters, which have a special influence on the encoding quality of x264 for depth image streams. The most important parameters are *preset*, *IDR interval* and *crf*. We will provide details of our analysis in Section 4.2.

4 Results and Discussion

We performed our analysis and evaluation for the Kinect V1 and the Kinect V2. The Kinect V1 has a depth resolution of 640 x 480 pixels and the Kinect V2 has a depth resolution of 512 x 424 pixels. We recorded a sequence of approx. 4000 frames from five Kinect V1 sensors and three Kinect V2 sensors simultaneously. Three sensors of both Kinect-types were positioned at approximately the same location in order to capture the same depth image content, actually a person (standing and gesturing). For encoding we used version 142 of the x246 library [21], for decoding we used version 2.6.1 of the ffmpeg library [7]. The operating system was 64-bit Ubuntu 14.04.2 LTS running on a dual six-core Intel(R) Xeon(R) CPU E5-1650 V2@3.50GHz.

In our evaluation, we use the following notations: *Time* means average encoding (decoding) time in milliseconds per frame, *Ratio* means average compression ratio, *Error* lists the mean absolute error (MAE) between the decoded depth image and the original image along with the maximum absolute error in brackets, *Method* ranges from *1* to *4* and indicates the different assignment schemes which were used as described in Subsection 3.2.

4.1 Evaluation of Depth-bit Assignment-Methods

First, we evaluated our proposed depth-bit assignment schemes. The results are listed in Table 1 and Table 2. It can be seen from Table 1 that method 2 for color space 4:2:2 and method 4 for color space 4:4:4 using 10 bit precision for x264 result in a very good encoding performance. Both, the achieved compression ratios as well as the PSNR are relatively high. In Table 2, we compare the encoding performance for discarding the lowest 2 bits (denoted as *10bit@x264*) with the best results we achieved for the experiments from Table 1. We can see that the encoding is faster for *10bit@x264*, which is mainly achieved by avoiding the processing of the UV channels. In addition, both, PSNR and compression ratio are very close to *Method 2* and *Method 4*. As a first result, we can deduce that it is feasible to discard the lowest 2 bits instead of assigning them to UV channels. However, the error remains near 3 mm on average, as expected.

Next we compare our proposed hybrid lossless-lossy compression approach (denoted as *10bit@x264+2bit@RLE*) to the method *10bit@x264*. The results are listed in Table 2. As one can see, the encoding time increases for *10bit@x264+2bit@RLE*, which is mainly caused by the RLE encoding time overhead, and, the compression ratio is lower. However, the quality of compression in terms of PSNR, MAE (1.31 mm), as well as maximum error (26 mm) is much better compared to *10bit@x264*. As a second result, we can summarize, that our hybrid approach can achieve a very high quality at the cost of a lower compression ratio and a slightly longer encoding time.

Table 1: Experimental results for different depth-bit assignment methods (1 to 4).

Bit depth	Color space	Method	Time (ms)	Ratio	PSNR (dB)	Error (mm)
8	420	1	13.0	0.43	54.55	3.60 [97]
	422	1	14.5	0.41	53.01	5.41 [112]
		2	14.4	0.39	52.06	7.83 [101]
	444	1	14.6	0.34	53.13	5.11 [89]
		2	14.5	0.34	53.67	4.90 [104]
		3	14.5	0.31	54.23	3.40 [99]
		4	14.7	0.29	56.20	3.49 [103]
10	420	1	14.0	0.21	57.58	3.47 [70]
	422	1	14.5	0.18	58.11	3.51 [70]
		2	**13.9**	**0.17**	**58.90**	**2.70 [77]**
	444	1	14.8	0.18	57.30	4.01 [79]
		2	14.7	0.17	57.42	3.88 [88]
		3	14.7	0.17	58.18	3.78 [90]
		4	**14.6**	**0.15**	**58.40**	**3.26 [89]**

Table 2: Evaluation of discarding the lowest 2 bits per depth pixel and comparison of hybrid lossless-lossy encoding (*10bit@x264+2bit@RLE*) vs. pure x264-encoding (*10bit@x264*).

Color space	Method	Time (ms)	Ratio	PSNR (dB)	Error (mm)
444	4	14.6	0.15	58.40	3.26 [89]
422	2	13.9	0.17	58.90	2.70 [77]
420	10bit@x264	11.8	0.15	58.66	3.07 [80]
420	**10bit@x264 + 2bit@RLE**	**15.1**	**0.23**	**65.17**	**1.31 [26]**

4.2 Evaluation of x264 Encoding Parameter Settings

Next, we evaluated the parameter settings for x264-encoding. The results are listed in Table 3 and Table 4. In Table 3, *crf* is set to 1, and the depth bit assignment scheme is *10bit@x264*. It is obvious that the encoding performance is best for *IDR interval=1*. The reason for this is that depth images typically have areas with silhouette edges and also smooth surfaces, which is more suitable for intra-prediction than inter-prediction. Therefore we set both, *crf* and *IDR interval* to 1 for the evaluation of different *preset* settings (Table 3). In comparison to *veryfast* and *superfast*, *ultrafast* results in smaller depth errors and shorter encoding times with only little loss in the compression ratio.

In our next evaluation, we therefore set *preset* to *ultrafast* and *IDR interval* to 1. We were now interested in a comparison between *10bit@x264+2bit@RLE* and *10bit@x264* for different *crf* settings. The results are listed in Table 4. While encoding times are always higher for our hybrid method the quality in terms compression ratio, PSNR and error is much better. However, *10bit@x264* can still be an efficient encoding method if a higher compression ratio (at the cost of quality) is preferred.

Table 3: Evaluation of different *IDR interval* settings and different *preset* settings.

Preset	IDR interval	Time (ms)	Ratio	PSNR (dB)	Error (mm)
very fast	20	12.0	0.17	57.66	3.40 [98]
very fast	10	12.0	0.17	57.86	3.33 [98]
very fast	1	11.8	0.15	58.66	3.07 [80]
superfast	1	9.7	0.16	59.80	3.00 [73]
ultrafast	1	**8.0**	**0.17**	**60.34**	**3.04 [68]**

4.3 Depth Compression for Real-Time 3D Reconstruction

We evaluated our compression approach in the context of real-time 3D reconstruction. The 3D reconstruction itself is performed using the approach from [3]. We mapped the most suitable configurations to 8 compression levels as listed in Table 4. The compression ratio for depth encoding can then be adjusted during runtime from 1 (lowest) to 8 (highest). Figure 3 and Figure 4 illustrate results for 3D reconstructions for different

Table 4: Evaluation of different *crf* settings along with the mapping to 8 compression levels.

Method	crf	Time (ms)	Ratio	PSNR (dB)	Error (mm)	Compr. level
	1	11.7	0.25	68.02	1.19 [15]	1
	6	11.0	0.18	62.73	2.17 [28]	2
10bit@x264+2bit@RLE	12	8.5	0.12	57.92	3.66 [50]	4
	18	7.5	0.06	54.14	5.65 [87]	not used
	24	7.3	0.05	50.30	8.39 [183]	not used
	1	8.0	0.17	60.34	3.04 [68]	3
	6	7.7	0.11	56.50	4.59 [112]	5
10bit@x264	12	7.2	0.08	53.47	7.00 [168]	6
	18	7.2	0.04	49.72	10.16 [281]	7
	24	7.1	0.04	44.90	15.21 [542]	8

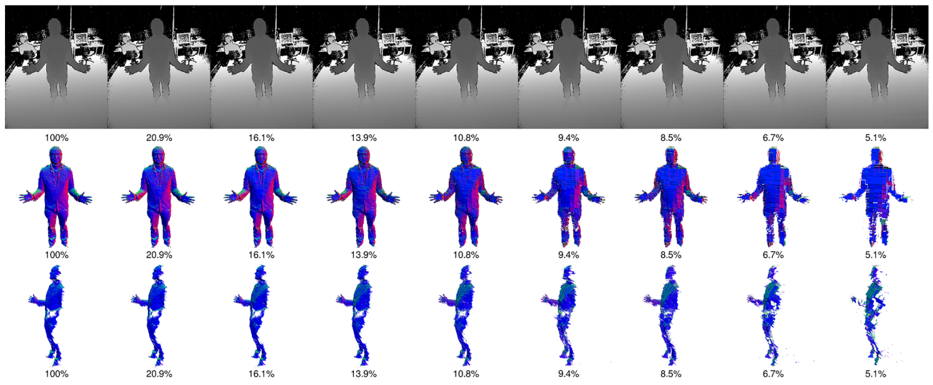


Fig. 3: Surface normal visualizations of 3D reconstructions for different compression ratios, generated from the corresponding depth image (upper row). Note, that the images in the lower row are reconstructed from the corresponding depth image, too.

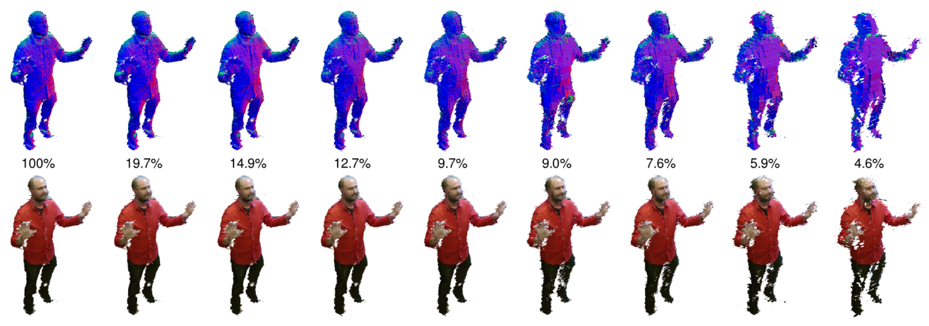


Fig. 4: Comparison of the resulting 3D reconstruction from 2 overlapping Kinect V2 sensors for different compression ratios.

compression levels. We can see that the subjective 3D reconstruction perception using our compression approach with a compression level from approx. 20% to 10% is almost indistinguishable from the original depth image. The perceived quality is still acceptable for a compression ratio of approx. 7% to 8%. For the highest compression settings, shown in the two rightmost columns, the quality of the reconstruction is obviously reduced, e.g. edges tend to deteriorate and the reconstructed surface appears blocky.

Table 5 lists the processing times, as well as the compression ratios and the necessary bit rates assuming the devices are running at 30Hz for different configurations. As one can see, we are able to encode and decode multiple depth image streams in real-time. The encoding as well as the decoding time scales with the compression level. The number of streams has only little influence due to highly parallelized processing. Note, that we do not provide an evaluation for multiple color image streams. In our current system, we use a DXT1-based color compression which is able to encode multiple color image streams in real-time, e.g. less than 30 ms for resolutions up to 1280 x 1080 per color image.

Table 5: Evaluation of different stream configurations and compression levels.

Streams config.	Compr. level	Encoding (ms)	Decoding (ms)	Ratio	Bitrate (Mbit/s)
1x640x480	1	11.9	6.6	25.7	4.5
	4	7.8	5.2	11.4	2.0
	8	7.2	4.3	4.6	0.8
3x640x480	1	13	6.7	23.6	12.4
	4	9.5	5.5	10.4	5.5
	8	8.6	4.6	4.3	2.2
5x640x480	1	13.8	8.6	24.7	21.8
	4	11	6.6	11.3	10.1
	8	10.2	5.8	4.6	4.1
1x512x424	1	10.2	6.2	21.3	2.6
	4	6.1	4.0	10.7	1.33
	8	5.3	3.4	5.0	0.62
3x512x424	1	11.8	6.3	20.6	7.6
	4	7.8	4.2	10.3	3.8
	8	7.2	3.6	4.9	1.82

5 Conclusion

The real-time compression of depth image streams becomes a basic necessity if multiple streams have to be sent over wide area networks. We presented an efficient solution to adapt the x264-video codec, which is designed for 10-bit YUV color space images, to 12-bit depth maps without modifying the codec algorithm itself. We deduced and evaluated optimal x264 encoding parameters for depth images through experimental statistics. Our research suggests that a hybrid lossless-lossy depth compression scheme provides a good tradeoff between quality and compression ratio by using x264 for lossy encoding and run length encoding to preserve the highest bits of 12 bit depth images. Our evaluation shows that we are able to achieve compression ratios in the range of 4:1 to 20:1 for multiple depth image streams in real-time. The visual quality of 3D reconstructions is close to the original for a compression ratio of up to 10:1. Although we did not provide results for real-time color compression our method can simplify the pipeline of color stream processing. A promising direction for future research is a silhouette-aware joint color and depth compression which can further increase both, compression ratio and reconstruction quality.

Acknowledgements

The author gratefully acknowledges the support of K. C. Wong Education Foundation and DAAD, NSFC (Grant No. 61073074, 61303144), Ningbo Natural Science Foundation (Grant No. 2013A610069), Zhejiang Science and Technology Program (Grant No. 2012C21004), Projects in Science and Technique of Ningbo Municipal (Grant No. 2012B82003), Zhejiang Higher Education Reform Project (Grant No. jg2013135), National Students' Innovation and Entrepreneurship Project (Grant No. 201410876012).

References

[1] C. Bal and T. Nguyen. Multiview video plus depth coding with depth-based prediction mode. *Circuits and Systems for Video Technology, IEEE Transactions on*, 24(6):995–1005, June 2014.

[2] F. Bannò, P. S. Gasparello, F. Tecchia, and M. Bergamasco. Real-time compression of depth streams through meshification and valence-based encoding. In *International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI'12, pages 263–270, New York, NY, USA, 2012. ACM.

[3] S. Beck, A. Kunert, A. Kulik, and B. Froehlich. Immersive group-to-group telepresence. *Visualization and Computer Graphics, IEEE Transactions on*, 19(4):616–625, April 2013.

[4] Y. Champawat and S. Kumar. Online point-cloud transmission for tele-immersion. In *International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI'12, pages 79–82, New York, NY, USA, 2012. ACM.

[5] T. Cho, Y. Lee, and J. Shin. A homogenizing filter for depth map compressive sensing using edge-awarded method. In *International Conference on ICT Convergence*, pages 591–595, Oct 2013.

[6] J.-A. Choi and Y.-S. Ho. Improved near-lossless hevc codec for depth map based on statistical analysis of residual data. In *International Symposium on Circuits and Systems*, pages 894–897, May 2012.

[7] FFMpeg. Ffmpeg (2014, nov. version 2.6.1): audio and video codec library.

[8] J. Fu, D. Miao, W. Yu, S. Wang, Y. Lu, and S. Li. Kinect-like depth data compression. *Multimedia, IEEE Transactions on*, 15(6):1340–1352, Oct 2013.

[9] J. Gautier, O. Le Meur, and C. Guillemot. Efficient depth map compression based on lossless edge coding and diffusion. In *Picture Coding Symposium, 2012*, pages 81–84, May 2012.

[10] S. Hoffmann, M. Mainberger, J. Weickert, and M. Puhl. Compression of depth maps with segment-based homogeneous diffusion. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 319–330. Springer Berlin Heidelberg, 2013.

[11] H. Ko and C.-C. Kuo. A new in-loop filter for depth map coding in hevc. In *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–7, Dec 2012.

[12] C. Lan, J. Xu, and F. Wu. Object-based coding for kinect depth and color videos. In *Visual Communications and Image Processing*, pages 1–6, Nov 2012.

[13] K. Mamou, T. Zaharia, and F. Prêteux. Tfan: A low complexity 3d mesh compression algorithm. *Comput. Animat. Virtual Worlds*, 20(23):343–354, June 2009.

[14] R. Mekuria, M. Sanna, S. Asioli, E. Izquierdo, D. C. A. Bulterman, and P. Cesar. A 3d tele-immersion system based on live captured mesh geometry. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys'13, pages 24–35, New York, NY, USA, 2013. ACM.

[15] B. T. Oh. An adaptive quantization algorithm without side information for depth map coding. *Image Commun.*, 29(9):962–970, Jan. 2014.

[16] F. Pece, J. Kautz, and T. Weyrich. Adapting standard video codecs for depth streaming. In *Eurographics Conference on Virtual Environments Joint Virtual Reality*, EGVE-JVRC'11, pages 59–66, Aire-la-Ville, Switzerland, Switzerland, 2011. Eurographics Association.

[17] S. Raghuraman, K. Venkatraman, Z. Wang, B. Prabhakaran, and X. Guo. A 3d tele-immersion streaming approach using skeleton-based prediction. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM'13, pages 721–724, New York, NY, USA, 2013. ACM.

[18] H. Schwarz, C. Bartnik, S. Bosse, H. Brust, T. Hinz, H. Lakshman, P. Merkle, K. Muller, H. Rhee, G. Tech, M. Winken, D. Marpe, and T. Wiegand. Extension of high efficiency video coding (hevc) for multiview video and depth data. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 205–208, Sept 2012.

[19] S. Schwarz, R. Olsson, M. Sjostrom, and S. Tourancheau. Adaptive depth filtering for hevc 3d video coding. In *Picture Coding Symposium*, pages 49–52, May 2012.

[20] G. Van Wallendael, S. Van Leuven, J. De Cock, F. Bruls, and R. Van de Walle. Multiview and depth map compression based on hevc. In *International Conference on Consumer Electronics*, pages 168–169, Jan 2012.

[21] Videolan. x264 (2014, sep. version 142) : a free h264/avc encoder.

[22] H. Yuan, S. Kwong, J. Liu, and J. Sun. A novel distortion model and lagrangian multiplier for depth maps coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 24(3):443–451, March 2014.

[23] M. Zamarin, M. Salmistraro, S. Forchhammer, and A. Ortega. Edge-preserving intra depth coding based on context-coding and h.264/avc. In *International Conference on Multimedia and Expo*, pages 1–6, July 2013.