Massively Parallel CPU-based Virtual View Synthesis with Atomic Z-test

Jakub Stankowski
Institute of Multimedia Telecommunications
Poznań University of Technology
Polanka 3
61-131 Poznań, Poland
jakub.stankowski@put.poznan.pl

Adrian Dziembowski
Institute of Multimedia Telecommunications
Poznań University of Technology
Polanka 3
61-131 Poznań, Poland
adrian.dziembowski@put.poznan.pl

ABSTRACT

In this paper we deal with the problem of real-time virtual view synthesis, which is crucial in practical immersive video systems. The majority of existing real-time view synthesizers described in literature require using dedicated hardware. In the proposed approach, the view synthesis algorithm is implemented on a CPU increasing its usability for users equipped with consumer devices such as personal computers or laptops. The novelty of the proposed algorithm is based on the atomic z-test function, which allows for parallelization of the depth reprojection step, what was not possible in previous works. The proposal was evaluated on a test set containing miscellaneous perspective and omnidirectional sequences, both in terms of quality and computational time. The results were compared to the state-of-the-art view synthesis algorithm – RVS.

Keywords

Virtual view synthesis, immersive video systems, real-time video processing.

1. INTRODUCTION

The basic idea of an immersive video system is to allow a user for immersing into the scene by giving a possibility of free virtual navigation within a scene captured by a multicamera system [Goo12], [Sta18], equipped with perspective or omnidirectional cameras (Fig. 1). In a typical scenario, where the scene is represented using the multiview plus depth (MVD) representation [Mül11], such a possibility is provided by the synthesis (i.e., rendering) of virtual viewpoints.

There are multiple good-quality virtual view synthesis methods described in the literature, e.g., [Dzi19], [Fac18] or [Sen18]. However, these methods cannot be used in the real-time scenario, making them not suitable for practical immersive video systems, where the system's response to user's change of position should be immediate, and the virtual view should be synthesized with possibly smallest delay [Dzi18].

2. FAST VIRTUAL VIEW SYNTHESIS

In the literature, several real-time virtual view synthesis methods are described. However, the vast majority of them require dedicated hardware, such as powerful graphic cards (e.g., [Non18], [Zha17]), FPGA devices (e.g., [Aki15], [Li19]) or even VLSI devices [Hua19].

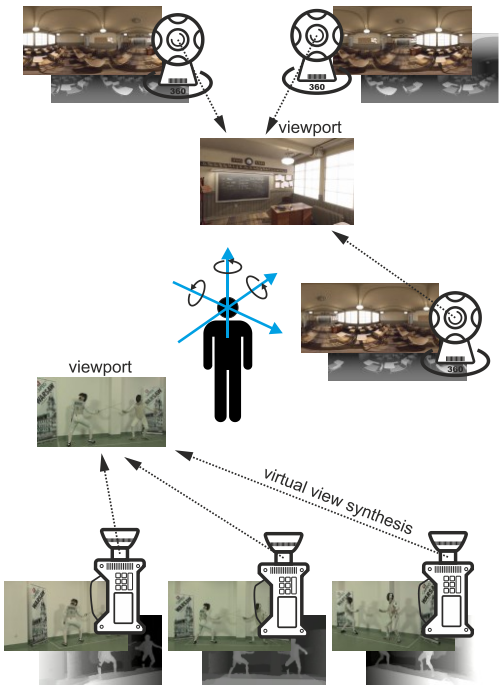

Figure 1. Idea of an immersive video system.

In this paper, we present a significant improvement of the methods we have described in [Sta20] and [Sta22], designed for real-time handling of perspective and omnidirectional content, respectively.

The general scheme of our fast CPU-based virtual view synthesis algorithm is presented in Fig. 2. It can be divided into three major steps: synthesis of the depth map corresponding to the virtual view (orange blocks in Fig. 2), synthesis of the virtual view itself (blue block), and postprocessing of synthesized view (grey blocks). The reprojection of depth and texture is performed differently for perspective views (using homography matrices [Sta20]) and omnidirectional ones (using equations described in [Sta22]). The postprocessing step is identical, independently on the video type and includes operations like filtering and inpainting.

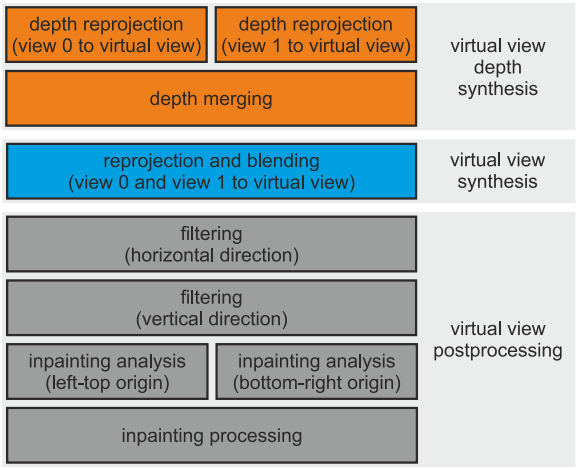

Figure 2. Overview of the fast virtual view synthesis algorithm. Figure from [Sta22].

As described in [Sta20] and [Sta22], the algorithm provides good-quality virtual views and allows to achieve real-time processing even for very high-resolution video, i.e., synthesizing of Full HD (or 2K×2K) virtual views based on two 4K input views.

The improvement presented in this paper allows to decrease this time even more, allowing for real-time synthesis of 4K sequences.

3. PARALLELIZATION LIMITATION

As described in [Sta20], the computational time of the view synthesis algorithm can be significantly decreased by using the multithread implementation and exploiting computing capabilities of modern multicore processors.

The stages related to texture reprojection and virtual view postprocessing (blue and grey blocks in Fig. 2) can be freely parallelized by dividing picture area into rows, slices, tiles, etc. and processing each one using separate thread.

However, this simple divide-and-process approach cannot be applied to the most complex stage – depth reprojection. The location of reprojected depth is unpredictable and may induce a situation where two threads try to write data into the same memory location. This can lead to race condition and corruption of reprojected depth.

A solution for above mentioned issue was proposed in [Sta20] and further developed in [Sta22]. A technique presented in [Sta20] introduced using of the algorithm called Independent Projection Targets (IPT). When using IPT, different slices of depth (processed in separate threads) are reprojected into separate target buffers and then merged (Fig. 3). Unfortunately, the IPT algorithm has scalability limitations. Each processing thread requires a dedicated set of intermediate target buffers which leads to increased memory complexity. To make things worse, the intermediate buffers have to be merged causing the merge operation to become more time consuming when higher number of processing threads is used. The overhead related to added complexity of intermediate buffers merging offsets the gain from using higher number of depth reprojection threads.

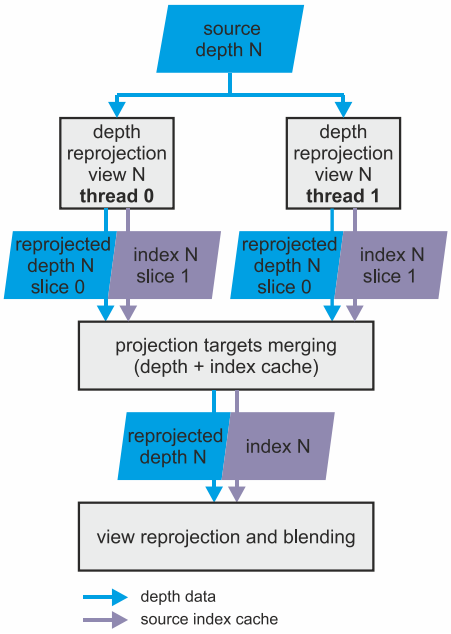

Figure 3. The idea of the independent projection targets (IPT) with data flow and intermediate data structures.

4. ATOMIC Z-TEST

In order to overcome the IPT scalability issues, we have developed a new algorithm called atomic z-test (AZT). The idea behind AZT is to use a special type

of memory access operation called “atomic” [Zhu84] to share single target buffer between processing threads in order to avoid excessive memory usage and intermediate buffers merging overhead. The atomic instruction allows to perform a single load-modify-write operation which cannot be interrupted by another core [Sch15].

In order to use the atomic z-test the data layout has to be changed. In previous implementations [Sta20], [Sta22] separate target depth and source index buffers were used. Since atomic instructions operate on single memory location, we had to combine target depth and source index buffers into single buffer. Therefore, the reprojected data buffer contains pairs of concatenated values – depth on more significant bits and index on least significant bits, both stored as 32-bit unsigned integer.

The idea behind z-test in depth reprojection is to select closest object (which corresponds to lowest depth / highest disparity value). Since one depth and index pair occupies a 64-bit memory location with depth placed on most significant bits, a 64-bit unsigned integer maximum operation can be used during z-test and depth merging. This allows us to perform z-test by updating a single 64-bit value as one atomic operation.

Different CPU (and GPU) architectures allow for a variety of atomic operations. The most common is compare-and-exchange also called compare-and-swap (CAS), however more sophisticated operations like addition, subtraction, etc., are sometimes available. In our case the desired instruction would be 64-bit atomic maximum. Unfortunately, no general-purpose CPU offers such an instruction. This leads us to necessity of simulating it by using compare-and-swap (CAS) operation. Implementation details are provided in section 5.

Fig. 4 illustrates simplified data flow in AZT algorithm (only two processing threads are drawn for clarity reasons). The presented diagram shows that AZT allows for using single reprojected depth and index buffer which allows for eliminating the time-consuming stage of depth merging.

According to [Zhu84], atomic operations are slightly slower than regular memory accesses, however the biggest performance penalty is related to a situation where two cores try to perform an atomic operation on the same location (to be precise – within the same cache line) and memory subsystem has to serialize request coming from different CPU cores. This leads us to conclusion that the performance of proposed algorithm can depend on two factors: the number of inter-thread collisions and the quality of the CPU memory subsystem implementation.

Nevertheless, the usage of AZT allows us to use all available CPU cores/threads and overcome IPT scalability issues.

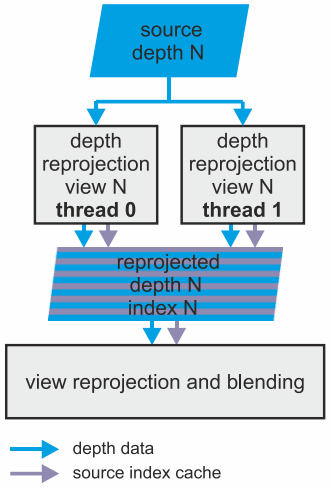

Figure 4. The data flow of the atomic z-test (AZT) with intermediate data structures.

5. IMPLEMENTATION

The proposed atomic z-test algorithm was implemented using the C++11 standard library [ISO11] and operations defined in `<atomic>` header. The `std::compare_exchange_weak<uint64_t>` function was chosen as a portable way to use compare-and-swap operation. The exemplary implementation of atomic z-test is provided below:

```
/**
@brief Performs atomic z test on DepthIndex buffer
@param PtrDI is pointer to location within buffer
@param NewD is reprojected depth value
@param NewI is index representing source depth location
*/
void AZT(uint64_t* PtrDI, uint32_t NewD, uint32_t NewI)
{
  uint64_t BuffDI = *PtrDI; //DI - DepthIndex
  uint32_t BuffD  = (uint32_t)(BuffDI >> 32);
  if(BuffD <= NewD)
  {
    uint64_t NewDI = ((uint64_t)NewD<<32)|(uint64)NewI;
    while(!std::atomic_compare_exchange_weak(
       (std::atomic_uint64_t*)(PtrDI), &BuffDI, NewDI))
    {
      if((uint32_t)(BuffDI >> 32) >= NewD) { break; }
    }
  }
}
```

In addition to the usage of atomic CAS, we used already described vectorization techniques [Sta20], [Sta22] by using AVX2 and AVX512 extensions.

6. EXPERIMENTS

Test sequences

The test set contained 9 miscellaneous test sequences (Fig. 5), including:

- 3 omnidirectional sequences:
  - A01: ClassroomVideo [Kro18] (4K×2K),
  - C01: Hijack [Dor18] (4K×2K),
  - C02: Cyberpunk [Jeo21] (2K×2K),

- 3 perspective computer-generated sequences:
  - J01: Kitchen [Boi18] (FullHD),
  - J04: Fan [Dor20] (FullHD),
  - W02: Dancing [Boi18] (FullHD),
- 3 perspective natural sequences:
  - D01: Painter [Doy18] (2K×1K),
  - L01: Fencing [Dom16] (FullHD),
  - L03: MartialArts [Mie23] (FullHD).

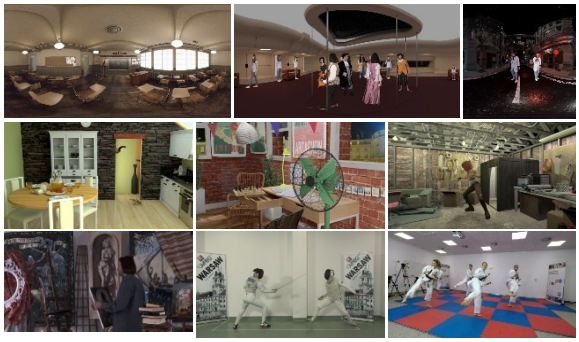

Figure 5. Test set used in the experiments. 1st row (from left): ClassroomVideo, Hijack, Cyberpunk; 2nd row: Kitchen, Fan, Dancing; 3rd row: Painter, Fencing, MartialArts.

The sequences are commonly used in immersive video applications, e.g., within ISO/IEC JTC1/SC29/WG04 MPEG Video Coding group [MPEG23].

Experiment setup

Test was performed on two computers with modern x86-64 CPUs: AMD Ryzen 9 5950X (containing 16 uniform cores) and Intel Core i7-12700k (containing 8 regular and 4 weak cores). Both processors are able to execute instructions from AVX2 extension set so during experimental evaluation the AVX2 vectorized implementation was used. Unfortunately, during experiments we have no access to any AVX512 capable CPU, therefore no results for AVX512 implementation is provided.

Quality and time evaluation

In order to assess the quality of virtual views synthesized using proposed real-time view synthesizer, we compared it to the state-of-the-art ISO/IEC MPEG's reference software – RVS [Fac18], [MPEG18]. The comparison is reported in terms of two objective quality metrics commonly used in works on immersive video: IV-PSNR [Dzi22] and WS-PSNR [Sun17].

The computational complexity was evaluated by measuring the average processing time required for synthesizing of a single frame of the sequence including timing for individual processing stages (depth projection, depth combining, view projection, virtual view filtering and inpainting).

Evaluation results

The results of performed experiments are presented in Tables 1 – 3.

Tables 1 and 2 present a detailed comparison of computational time of the proposed algorithm and the previous implementation described in [Sta20]. As presented, the proposed atomic z-test (AZT) operation allows for a significant reduction of the computational time.

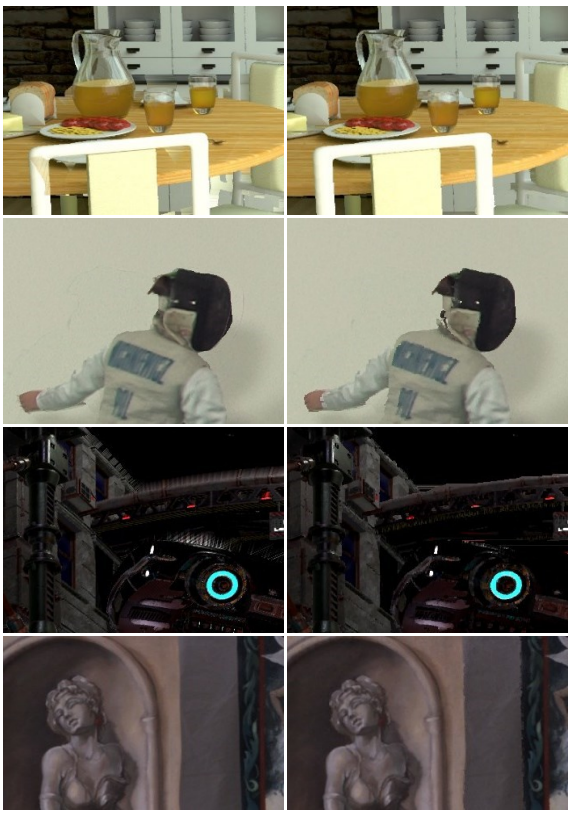

Figure 6. Fragments of virtual views synthesized using RVS (left) and the proposed method (right); sequences (from top): Kitchen, Fencing, Cyberpunk, and Painter.

For the 16-core AMD Ryzen CPU, AZT decreases the time required for synthesis of a single frame by more than 40%. For Intel i7-12700K CPU the decrease is smaller, but still significant. There are two reasons for lower gain caused by the use of AZT. The first one – a smaller number of cores, thus lower parallelization. The second reason is the heterogeneous structure of that CPU, which operates on 8 performant cores and 4 slower ones leading to unequal processing time for each core type.

As presented in Tables 1 and 2, the decrease of the computational time is caused by introducing the possibility of efficient parallelization of the depth projection step (denoted as "DP"). Such an implementation may lead to slightly longer step of combining depth candidates into the final virtual depth

Sequence	IPT algorithm processing time [ms]						Proposed algorithm processing time [ms]						Time reduction [%]	
	DP	DC	TP	F	I	Total	DP	DC	TP	F	I	Total	Depth	Total
A01	38.70	2.57	10.43	6.17	3.56	61.44	14.48	4.65	11.07	6.27	3.21	39.68	54%	35%
C01	38.52	2.52	10.05	5.27	2.73	59.10	13.72	4.66	10.94	4.90	2.71	36.93	55%	38%
C02	23.37	6.19	5.17	2.07	1.40	38.19	7.30	1.85	5.58	2.18	2.04	18.95	69%	50%
D01	5.60	3.31	2.87	2.03	1.01	14.81	2.98	0.84	2.79	1.55	0.67	8.84	57%	40%
J01	5.74	2.94	2.39	1.07	0.63	12.77	2.79	0.78	2.48	1.12	0.58	7.73	59%	39%
J04	7.54	2.93	2.47	1.37	0.76	15.07	2.82	0.77	2.50	1.15	0.60	7.84	66%	48%
L01	7.57	2.87	2.77	1.30	0.73	15.24	3.11	0.81	2.63	1.22	0.60	8.37	62%	45%
L03	7.71	2.94	2.42	1.24	0.80	15.11	2.90	0.84	2.55	1.14	0.65	8.07	65%	47%
W02	8.06	2.91	2.51	1.39	0.81	15.68	3.07	0.80	2.52	1.23	0.61	8.24	65%	47%
Average													**61%**	**43%**

Table 1. Performance evaluation – computation time comparison between IPT [Sta20] and proposed AZT algorithm for 16-core AMD Ryzen 9 5950X CPU. Processing stages: DP – depth projection, DC – depth combining, TP – texture projection, F – filtering, I – inpainting. "Depth time reduction" includes depth projection and depth combining.

Sequence	IPT algorithm processing time [ms]						Proposed algorithm processing time [ms]						Time reduction [%]	
	DP	DC	TP	F	I	Total	DP	DC	TP	F	I	Total	Depth	Total
A01	38.54	4.18	11.41	9.25	3.93	67.31	29.89	4.25	11.66	9.24	3.45	58.48	20%	13%
C01	37.34	4.13	11.10	6.46	3.45	62.48	30.47	4.24	11.29	6.76	3.16	55.92	16%	11%
C02	20.66	6.95	5.34	3.41	1.87	38.23	15.82	1.96	5.54	3.78	1.97	29.06	36%	24%
D01	5.81	3.36	3.00	2.91	1.19	16.29	6.67	0.81	2.73	2.95	1.10	14.27	18%	12%
J01	4.66	3.19	2.70	1.81	0.76	13.12	5.25	0.69	2.29	1.70	1.17	11.10	24%	15%
J04	5.17	3.19	2.70	1.88	0.75	13.69	5.43	0.71	2.35	1.95	0.78	11.23	27%	18%
L01	5.54	3.26	2.87	2.02	0.78	14.48	5.70	0.75	2.64	1.95	0.75	11.79	27%	19%
L03	5.36	3.18	2.98	1.54	1.68	14.74	5.62	0.71	2.91	1.61	0.85	11.69	26%	21%
W02	5.65	3.22	2.86	2.02	1.61	15.36	6.02	0.74	2.68	2.07	0.82	12.33	24%	20%
Average													**24%**	**17%**

Table 2. Performance evaluation – computation time comparison between IPT [Sta20] and proposed AZT algorithm for [8+4] core Intel i7-12700K CPU. Processing stages: DP – depth projection, DC – depth combining, TP – texture projection, F – filtering, I – inpainting. "Depth time reduction" includes depth projection and depth combining.

Sequence	Processing time [ms]			IV-PSNR [dB]			WS-PSNR [dB]		
	RVS	Proposed	Speedup	RVS	Proposed	Delta	RVS	Proposed	Delta
A01	15885	39.68	400	43.56	42.68	-0.89	32.21	31.74	-0.47
C01	15547	36.93	421	45.04	45.85	0.82	38.14	38.57	0.43
C02	7878	18.95	416	47.73	48.23	0.50	41.01	41.43	0.42
D01	3838	8.84	434	48.59	46.85	-1.74	38.46	36.94	-1.52
J01	3370	7.73	436	37.03	38.12	1.09	28.82	29.30	0.49
J04	3723	7.84	475	36.80	37.55	0.74	27.21	27.98	0.76
L01	3355	8.37	401	40.54	40.14	-0.40	29.55	29.21	-0.34
L03	3285	8.07	407	31.80	31.24	-0.55	26.81	26.14	-0.67
W02	3437	8.24	417	41.63	41.06	-0.57	29.43	28.91	-0.52
Average			**423**	41.41	41.30	**-0.11**	32.41	32.25	**-0.16**

Table 3. Performance and quality evaluation – comparison with the state-of-the-art view synthesis method RVS [Fac18].

map (depth map corresponding to the virtual view), but in total the entire depth processing step is significantly faster. All remaining steps (texture projection, filtering, and inpainting) are not impacted by the proposed AZT algorithm.

Table 3 shows the comparison of the quality of virtual views synthesized using the proposed method and the state-of-the-art synthesizer RVS. As presented, in terms of objective quality, both algorithms provide similar results, both for IV-PSNR and WS-PSNR. A slight quality decrease (0.11 for IV-PSNR and 0.16 for WS-PSNR, on average) is the cost for a huge speedup – the proposed algorithm is more than 400 times faster than RVS.

Also the subjective quality of virtual views synthesized using RVS and proposed method is similar (Fig. 6). The characteristics of the synthesis artifacts is slightly different (e.g., caused by a different inpainting technique), but it can be certainly stated, that the proposed real-time algorithm allows to achieve at least similar quality to the state-of-the-art view synthesis technique.

7. CONCLUSIONS

In the paper we have presented a versatile CPU-based virtual view synthesis method which can be used in practical, real-time immersive video systems.

The novelty of the proposed method is based on introducing the atomic z-test approach, allowing for massive parallelization of the depth reprojection step, which is a crucial and most time-consuming part of the entire view synthesis pipeline.

The proposed virtual view synthesis method allows for achieving real-time processing for both perspective and omnidirectional sequences, even for very high resolutions. As presented in the paper, for 4K sequence it is possible to achieve real-time view synthesis at 25 frames per second. For lower resolutions (i.e., FullHD) it requires less than 10 ms per frame, making it possible to be used also for high frame rate immersive video systems.

8. ACKNOWLEDGMENTS

This work was supported by the Ministry of Education and Science of Republic of Poland.

9. REFERENCES

[Aki15] Akin, A., Capoccia, R., Narinx, J., Masur, J., Schmid, A., and Leblebici, Y. Real-time free viewpoint synthesis using three-camera disparity estimation hardware. 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon, pp. 2525-2528, 2015.

[Boi18] Boissonade P., and Jung J. Proposition of new sequences for Windowed-6DoF experiments on compression, synthesis, and depth estimation. Document ISO/IEC JTC1/SC29/WG11 MPEG/M43318, Ljubljana, Slovenia, Jul. 2018.

[Dom16] Domański M. et al. Multiview test video sequences for free navigation exploration obtained using pairs of cameras. Doc. ISO/IEC JTC1/SC29/WG11, MPEG M38247, 2016.

[Dor18] Doré, R. Technicolor 3DoF+ test materials. ISO/IEC JTC1/SC29/WG11 MPEG, M42349, San Diego, CA, USA, 04.2018.

[Dor20] Doré R. et al. InterdigitalFan0 content proposal for MIV. Doc. ISO/IEC JTC1/SC29/ WG04 MPEG VC/ M54732, Online, Jul. 2020.

[Doy18] Doyen D. et al. [MPEG-I Visual] New Version of the Pseudo-Rectified Technicolor painter Content. Doc. ISO/IEC JTC1/SC29/ WG11 MPEG/M43366, Ljublana, 2018.

[Dzi18] Dziembowski, A., and Stankowski, J. Real-time CPU-based virtual view synthesis. 2018 International Conference on Signals and Electronic Systems, Kraków, Poland, 2018.

[Dzi19] Dziembowski, A., Mieloch, D., Stankiewicz, O., Domański, M., Lee, G., and Seo, J. Virtual view synthesis for 3DoF+ video. 2019 Picture Coding Symposium (PCS), Ningbo, China, 2019.

[Dzi22] Dziembowski A., Mieloch D., Stankowski J. and Grzelka A., IV-PSNR—The Objective Quality Metric for Immersive Video Applications, IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 11, pp. 7575-7591, Nov. 2022, doi: 10.1109/TCSVT.2022.3179575.

[Fac18] Fachada, S., Bonatto, D., Schenkel, A., and Lafruit, G. Depth image based view synthesis with multiple reference views for virtual reality. 3DTV-Conference: The True Vision – Capture, Transmission and Display of 3D Video (3DTV-CON), Helsinki, Finland, 2018.

[Goo12] Goorts, P., Dumont, M., Rogmans, S., and Bekaert, P. An end-to-end system for free viewpoint video for smooth camera transitions. 2012 International Conference on 3D Imaging (IC3D). Liege, Belgium, 2012.

[Hua19] Huang, H., Wang, Y., Chen, W., Lin, P. and Huang, C. System and VLSI implementation of phase-based view synthesis. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, pp. 1428-1432, 2019.

[ISO11] Information technology — Programming languages — C++, ISO/IEC 14882:2011, ISO/IEC JTC 1/SC 22 International Organization for Standardization.

[Jeo21] Jeong, J.Y., Yun, K.J., Lee, G., Cheong, W.S., and Yoo, S. [MIV] ERP Content Proposal for MIV ver.1 Verification Test. ISO/IEC JTC1/SC29/WG04 MPEG VC, M58433, Online, 10.2021.

[Kro18] Kroon, B. 3DoF+ test sequence ClassroomVideo. ISO/IEC JTC1/SC29/WG11 MPEG, M42415, San Diego, CA, USA, 04.2018.

[Li19] Li, Y., Claesen, L., Huang, K., and Zhao, M. A real-time high-quality complete system for depth image-based rendering on FPGA. IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 4, pp. 1179-1193, 2019.

[Mie23] Mieloch, D., Dziembowski, A., Szydełko, B., Klóska, D., Grzelka, A., Stankowski, J., Domański, M., Lee, G., and Jeong, J.Y. [MIV] New natural content – MartialArts. ISO/IEC JTC1/SC29/WG04 MPEG VC, M61949, Online, 01.2023.

[MPEG18] "Reference View Synthesizer (RVS) manual," Doc. ISO/IEC JTC1/SC29/WG11 MPEG, N18068, Macao, Oct. 2018.

[MPEG23] Common test conditions for MPEG immersive video. ISO/IEC JTC1/SC29/WG04 MPEG VC, N0307, Online, Jan. 2023.

[Mül11] Müller, K., Merkle, P., and Wiegand, T. 3-D Video Representation Using Depth Maps. Proceedings of the IEEE, vol. 99, no. 4, pp. 643-656, Apr. 2011.

[Non18] Nonaka, K., Watanabe, R., Chen, J., Sabirin, H., and Naito, S. Fast plane-based free-viewpoint synthesis for real-time live streaming. 2018 IEEE Visual Communications and Image Processing (VCIP), Taichung, Taiwan, pp. 1-4, 2018.

[Sen18] Senoh, T., Tetsutani, N., and Yasuda, H. Depth estimation and view synthesis for immersive media. 2018 International Conference on 3D Immersion (IC3D), Brussels, Belgium, 2018.

[Sch15] Schweizer H., Besta M. and Hoefler T., "Evaluating the Cost of Atomic Operations on Modern Architectures," 2015 International Conference on Parallel Architecture and Compilation (PACT), San Francisco, CA, USA, 2015, pp. 445-456, doi: 10.1109/PACT.2015.24.

[Sta18] Stankiewicz, O., Domański, M., Dziembowski, A., Grzelka, A., Mieloch, D., Samelak, and J. A Free-viewpoint Television system for horizontal virtual navigation. IEEE Transactions on Multimedia, vol. 20, no. 8, pp. 2182-2195, 2018.

[Sta20] Stankowski, J., and Dziembowski, A. Fast view synthesis for immersive video systems. Proceedings of the 28. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG'2020, Plzen, Czech Republic, 05.2020.

[Sta22] Stankowski J., and Dziembowski A., Real-time CPU-based view synthesis for omnidirectional video, 30th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG 2022, Pilsen, Czech Republic, 05.2022.

[Sun17] Sun, Y., Lu, A., and Yu, L. Weighted-to-Spherically-Uniform Quality Evaluation for Omnidirectional Video. IEEE Signal Processing Letters 24.9(2017):1408-1412.

[Zha17] Zhang, L., Li, Y., Zhu, Q., and Li, M. Generating virtual images for multi-view video. Chinese Journal of Electronics, vol. 26, no. 4, pp. 810-813, 2017.

[Zhu84] Zhu C.-Q. and Yew P.-C., "A synchronization scheme and its applications for large multiprocessor systems", Proc. 4th Int. Conf. Distrib. Computing Syst., pp. 486-493, 1984.