

Novel View Synthesis in Embedded Virtual Reality Devices

Laurie Van Bogaert, Daniele Bonatto, Sarah Fachada, Gauthier Lafruit

Laboratory of Image Synthesis and Analysis at the Université Libre de Bruxelles; Brussels, Belgium

Abstract

Virtual Reality and Free Viewpoint navigation require high-quality rendered images to be realistic. Current hardware assisted raytracing methods cannot reach the expected quality in real-time and are also limited by the 3D mesh quality. An alternative is depth image-based rendering (DIBR) where the input only consists of images and their associated depth maps for synthesizing virtual views to the Head Mounted Display (HMD). The MPEG Immersive Video (MIV) standard uses such a DIBR algorithm called the Reference View Synthesizer (RVS). We have first implemented a GPU version, called the Real-time Accelerated View Synthesizer (RaViS), that synthesizes two virtual views in real-time for the HMD. In the present paper, we explore the differences between desktop and embedded GPU platforms, porting RaViS to an embedded HMD without the need for a separate, discrete desktop GPU. The proposed solution gives a first insight into DIBR View Synthesis techniques in embedded HMDs using OpenGL and Vulkan, a cross-platform 3D rendering library with support for embedded devices.

Introduction

Virtual reality (VR) and Free Viewpoint navigation have become accessible to the researchers and to the public. However, rendering high-quality realistic sceneries remains a challenge. Even using modern methods such as hardware assisted raytracing [1], rendering such content in real-time is only achievable with raytracing capable GPU present in higher-end gamer computers. For embedded devices available to the public such as smartphones, raytracing is currently not possible as they have limited computational power. Moreover, traditional rendering and raytracing rely on 3D meshes, limiting the achievable image realism [2].

High-Quality realistic 3D models are constructed either by a talented 3D artist using a 3D software such as Blender [3] or the use of 3D reconstruction methods [4] which use special hardware (e.g. laser scanner [5], LIDAR camera [6], etc.) or image-based algorithms. In particular, structure-from-motion (SfM) [7] reconstructs a point-cloud (PC) by using hundreds to thousands of pictures of the scene (explicit model). The PC needs to be meshed afterwards to obtain a 3D model. This process is slow and can take up to hours [8] and the resulting 3D mesh contains noise to be removed by a trained technician.

Alternatively, instead of creating a 3D model, a novel view can be synthesized at the viewer position using only a few reference images, this process is referred to as image-based rendering (IBR) and uses mathematical properties of the camera model to recreate novel views implicitly [9, 10]. Advantageously, the computations are bound to the number of pixels in the views, while in traditional 3D meshes, the rendering time depends on the complexity of the model. Potentially, by using a low number of input

views, a high quality natural scene synthesis can be reached in real-time. This is case of the Reference View Synthesis Software (RVS) [2, 11, 12, 13] which is a Depth Image-Based Rendering (DIBR) method as for each reference view, a corresponding depth map is provided.

Nonetheless, synthesizing novel views on embedded devices is still a significant challenge due to the limited computational power and the restricted transfer bandwidth between the GPU and the main memory [14]. In addition, as such devices use batteries, efficient energy management is a concern [15].

This work demonstrates a port of RaViS [13], a state-of-the-art GPU accelerated DIBR method based on the Reference View Synthesis (RVS) [2, 11, 13, 12] to target embedded devices, and in particular an embedded Head Mounted Display (HMD).

Related work

The Reference View Synthesis Software (RVS) [2, 11, 13, 12] is an open-source DIBR view synthesizer of the MPEG Immersive Video group for their new standard: MIV [16]. Using only four 1080p input views and their corresponding depth maps, RVS synthesizes novel views up to 2×90 frames per second (fps) in a head mounted display (HMD) using common hardware (NVIDIA GTX 1080Ti) while outperforming the old reference software VSRS [17] by an average of 2.5 dB PSNR [18]. RVS takes as input static or dynamic content [2] and has been used in a variety of contexts, including Super-MultiView light field displays [19], foveated light field displays [20] and engraved holographic stereograms [21]. The last version (v4.0) adds basic support for non-Lambertian content [22, 23] (i.e. objects with transparency and reflections) by enhancing the depth maps with polynomial interpolation techniques.

RVS performs the view synthesis in two main steps: warping and blending. RaViS [13], the real-time extension of RVS, exploits the graphics pipeline, using the Open Graphics Library (OpenGL) [24], and encodes each step as an OpenGL pass. During the warping step, the pixels of the reference view are implicitly projected in 3D using the depth map and the camera parameters given as input before being reprojected at the novel view camera's pose. The core idea is to link together adjacent pixels into triangles and exploit the rasterization step to colorize them after the reprojection, avoiding the costly mesh generation in SfM techniques. Using a ping-pong buffer, the synthesized views at the final pose are blended together. To avoid classical view synthesis issues [11], a weight is given to discard unfit triangles.

Other approaches explore the use of neural networks to implicitly represent a scene in the form of a neural radiance field [25, 26] or in multiplane-images (MPI) [27, 28] which are a stack of color images plus transparency, recreating the parallax effect. Embedded devices can display MPI images very efficiently as a new view can be created by using inverse homography and al-



(a) On an Android smartphone

(b) On the Oculus Quest 2

Figure 1. The proposed implementation of RVS using Vulkan running on a smartphone (Google Pixel 4a)(a) and an embedded head mounted display (Oculus Quest 2)(b)

pha composition on the different plane of the MPI. By construction, one MPI handles only a particular pose with a small parallax which is insufficient for six degrees of freedom (6 DoF) VR application. Multiple MPIs can be blend together to enhance the navigation in the scene but this tends to produces jumping artefacts. Current methods to generate MPI (Local Light Field Fusion (LLFF) [27], Nex [28]) for realistic scenes use neural networks and require extensive training time with heavy pre-processing. Furthermore, they are not yet ready for dynamic content as preliminary MPI on Android [29] experiments have been conducted by the MPEG Immersive video (MIV) standard but could not handle 1080p streams.

In addition of displaying high quality natural sceneries, novel view synthesis finds applications in embedded devices to correct the camera feed to the position of viewer's eyes [30, 31].

Another approach to render in real-time on embedded devices compensates the reduced computational power available by performing remote rendering: the rendering is performed on a high-end server and streamed via the network to the embedded device. However, the network latency introduces major issues in embedded VR as new views must be computed with an ideal of 60 to 120 frames per seconds (FPS) to follow the user movements. As a workaround, [32] proposes to synthesize discrete views on the server and upsamples the framerate of the device by using a proxy-guided IBR algorithm with a dual view representation. The proxy representation of the scene is generated by pre-computing different levels-of-details meshes. In addition, Facebook has recently developed (April 2021) a remote rendering solution (Oculus Air Link) for their Oculus headset [33].

Proposed approach

To synthesize novel views on embedded devices, the DIBR method is selected for its advantages [11], however, RaViS uses the OpenGL API, not available on such devices. This leaves WebGL, OpenGL ES or Vulkan as a replacement for it. WebGL does not support geometry shader required by RaViS and thus is dismissed. A solution would be to switch to OpenGL ES which is a simplification of OpenGL only for embedded systems. Vulkan [34], a recent cross-platform graphics library allows a fine control of the graphics pipeline, removing many OpenGL's synchronization limitations while still reaching equal or better performances. The fine tuning capabilities of Vulkan allows to tailor the graphics pipeline for the embedded devices while in

OpenGL/OpenGL ES the driver have to infer more and thus will produce a higher GPU usage. It also takes better into account modern GPU architectures, like tiled rendering, common in mobile. Even if new extensions are added regularly to OpenGL ES, there is no new core version since 2015 (OpenGL ES 3.2). Furthermore, Vulkan helps to reduce energy consumption [35], crucial in embedded devices [15]. For all this reasons, Vulkan was chosen to replace OpenGL for the embedded 3D rendering.

It is not sufficient to translate OpenGL to Vulkan code as their rendering commands have different costs between desktop and embedded devices due to differences in hardware design. Therefore, custom steps are added, following the mobile development guidelines [36, 37]. Mainly, all operations causing access to main memory must be limited. The guidelines also recommend the usage of subpasses which does not exist in OpenGL. Correctly using them allows the driver to regroup multiple rendering command on the same tile.

Even with those optimisations, mobile devices can only handle a very limited number of triangles. This is a problem since the warping phase in RaViS generates around 4M triangles for a 1080p input while the Oculus documentation suggests the developers to keep the triangles count to 750k-1000k by frame [38]. In our experiments, a smaller size dataset is also added.

Since Vulkan is a cross-platform library, the proposed work is able to run on variety of devices from desktops to embedded devices as smartphones in addition of the embedded head mounted displays. However, the interactions with the user have been implemented differently across the platforms. On desktop, a basic implementation displays the novel view on the screen and reacts to keyboard inputs. Simple touchscreen interactions are added to support smartphones. However, porting the applications to an embedded head mounted display necessitates additional changes. Vulkan is a general purpose graphics library, it does not handle HMD specific operations such as lens distortion, input handling, headset pose, movement prediction, *etc.* To interface Vulkan with the hardware, the recent OpenXR standard [39], a non-vendor specific API for VR devices is used.

Finally, RVS performs a basic post-processing inpainting method which affects the borders of the synthesized views, it is not implemented in our approach to avoid additional data transfers and to focus on the view synthesis aspect.

Evaluations

To ensure we do not alter the software results, our implementation quality is compared with RVS v3.1 on several standardized datasets using peak-signal-to-noise-ratio (PSNR). Small differences in quality are to be expected due to the difference between the graphics frameworks. A second experiment reports the speed (FPS) on corresponding native screen resolutions on different device types.

Hardware




The comparisons are performed between:

- A high end desktop computer, equipped with an 11th gen intel core i7-11700K 3.6 GHz and an NVIDIA RTX 3080 GPU.
- A Google Pixel 4a (Android 11), as embedded device, equipped with a Qualcomm Snapdragon 730G SoC, integrated GPU Adreno 618 and a resolution of 2340x1080 pixels.
- An Oculus Quest 2 HMD device, equipped with a Qualcomm Snapdragon XR2 SoC, integrated GPU Adreno 650 and a resolution of 1832x1920 per eye.

Datasets

The speed and quality are evaluated on the standardized datasets featuring synthetic and real-world content: ToysTable, Classroom, Museum (see table below). The depth maps of real-world dataset can be estimated using the MPEG-I Depth Estimation Reference Software (DERS) [40] or the Immersive Video Depth Estimation (IVDE) software [41, 42]. ToysTable [43, 44, 45] dataset at lower resolution was obtained by resizing the inputs (area interpolation) and dividing the "Resolution", "Focal" and "Principle_point" (given in pixels) fields in the configuration file by two.

Datasets used during the experiments with their dimensions, projection type (equirectangular or perspective) and if the content is natural (N) or synthetic (S)

Name	Dimension	Proj.	Type	Image
ToysTable [43, 44, 45]	1920x1080 960x540	pers.	N	
Classroom [44]	4096x2048	equi.	S	
Museum [44]	2048x2048	equi.	N	

Quality

The classical image quality metrics, peak-signal-to-noise-ratio (PSNR), is used to ensure the quality is not altered by the code modifications. The PSNR is taken as the average of the luminance and chrominance channels of the YUV file format. The images are cropped to remove the area where inpainting takes place. The first comparison (Table 2a, b) with one input view ensures that the warping phases match in the two implementations, while

the second (c, d) ensures that the blending operations are correctly supported. As it is shown in the table below, the results obtained by the two implementations are similar. The last comparison tests the equivalence for datasets requiring equirectangular projection and unprojection. Here, more visible differences can be noted (*i.e.* Figure 2).

PSNR for the original OpenGL implementation (a, c) and the proposed Vulkan implementation (b, d) on the ToysTable dataset for z=300mm using 1 and 4 input views. The baseline for the x direction is 34 mm while for the y direction is 23 mm. The results obtained by a synthesized view at the position of one of the input views is indicated in gray.

Using 1 input view

34.61	36.06	37.33	36.13	34.73	34.60	36.06	37.33	36.13	34.53
35.53	38.11	79.40	38.45	35.81	35.51	38.11	79.40	38.45	35.17
34.16	35.64	36.74	35.74	34.35	34.15	35.64	36.74	35.74	33.95

(a) RVS 3.1 (OpenGL)

(b) proposed (Vulkan)

Using 4 input views

36.57	39.11	38.75	39.33	36.60	36.57	39.11	38.75	39.33	36.60
37.01	39.04	39.39	39.15	37.01	37.01	39.04	39.39	39.15	37.01
36.61	39.41	38.90	39.51	36.59	36.61	39.41	38.90	39.51	36.59

(c) RVS 3.1 (OpenGL)

(d) proposed (Vulkan)

PSNR for Classroom Dataset for 4 input views (v0,v4,v7,v8) with a baseline of at most 20.7 cm between two views.

Name	v0	v1	v2	v3	v4	v5	v6	v7
RVS 3.1	35.28	34.95	34.95	35.03	35.24	34.86	34.81	35.17
proposed	35.39	35.10	35.05	35.17	35.31	35.01	34.94	35.29
	v8	v9	v10	v11	v12	v13	v14	
RVS 3.1	35.26	34.61	34.72	34.65	34.69	34.52	34.48	
proposed	35.35	34.79	34.95	34.79	34.92	34.70	34.65	



(a) RVS 3.1 (OpenGL)

(b) proposed (Vulkan)

Figure 2. Zoom into the synthesized view at position v5 in the Classroom dataset.

Speed

The execution time (FPS) is evaluated on the different datasets for desktop (Figure 3), as well that on mobile (Figure 4) and on the headset (Figure 5). As expected the framerate is lower

on the mobile and the HMD. Missing values correspond to rendering times too high for the driver. Also note that in the case of the head-mounted display, OpenXR will introduce a small overhead. While the Oculus Quest runs by default at a framerate of 72 FPS, it is not mandatory to reach it. Indeed, Facebook introduced the SpaceWarp [46] feature for the Oculus Quest, which allows to display 72 FPS on the headset while the application only renders half the frames (36 FPS). However, this technique requires to calculate motion vectors between two frames, not available to us.

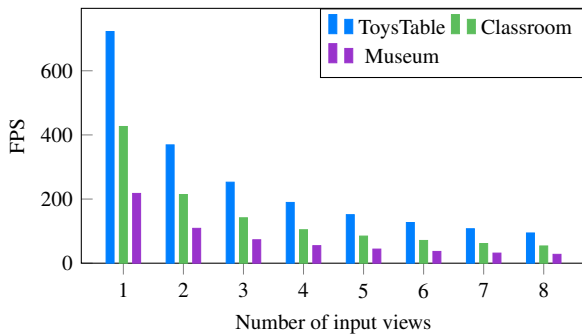


Figure 3. Number of FPS (frames per second) achieved by the desktop application for a 1080p resolution.

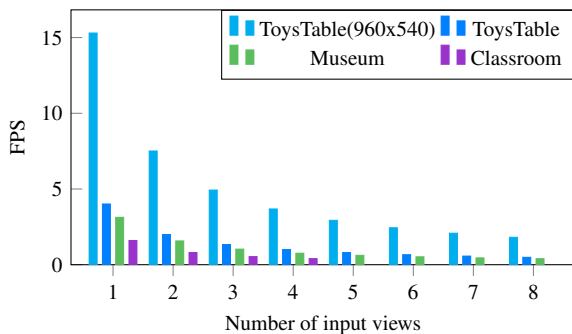


Figure 4. Number of FPS (frames per second) achieved by the application on the mobile device for the 2340x1080 resolution

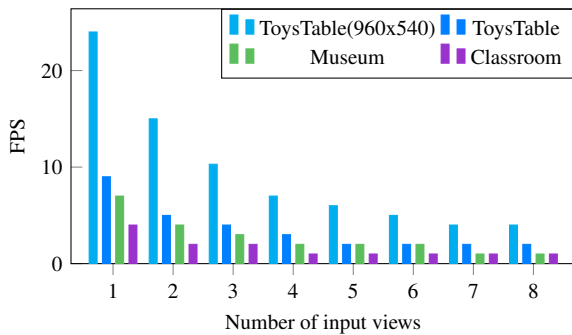


Figure 5. Number of FPS (frames per second) achieved by the application on the Oculus Quest 2 with a target resolution of 1440x1584 as requested by the OpenXR API.

Conclusion and future work

While real-time view synthesis of natural content in embedded devices is still out of reach, this work presents a successful port of the state-of-the-art DIBR software RVS using cross-platform Vulkan and OpenXR libraries, allowing for further explorations in the embedded world. The proposed work runs on a mid-range smartphone even if the framerate is lower for higher dimension datasets. While the results currently do not reach the high framerate necessary for VR headsets, they are still encouraging. Future work will explore the reduction of the number of triangles used in the warping phase to reach higher framerates. This, with the addition of the SpaceWarp feature, should allow to reach the framerate required for a comfortable VR experience with embedded devices.

Moreover, since loading the enhanced depth map is more expensive, the Non-Lambertian support (added in RVS 4.0) will be left as future work for the mobile version.

Acknowledgments

This work was supported by the Fonds de la Recherche Scientifique - FNRS, Belgium, under Grant n°33679514, ColibriH, by the Walloon region under a FRIA funding, and by the EU project HoviTron, Grant Agreement n°951989 on Interactive Technologies, Horizon 2020. Sarah Fachada is a Research Fellow of the Fonds de la Recherche Scientifique - FNRS, Belgium.

References

- [1] NVIDIA Corporation, "NVIDIA TURING GPU ARCHITECTURE: Graphics Reinvented." <https://images.nvidia.com/aem-dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf>, 2018.
- [2] D. Bonatto, S. Fachada, S. Rogge, A. Munteanu, and G. Lafruit, "Real-Time Depth Video Based Rendering for 6-DoF HMD Navigation and Light Field Displays," IEEE Access, pp. 1–20, Oct. 2021.
- [3] Blender Online Community, Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2021.
- [4] Z. Kang, J. Yang, Z. Yang, and S. Cheng, "A Review of Techniques for 3D Reconstruction of Indoor Environments," ISPRS International Journal of Geo-Information, vol. 9, no. 5, p. 330, 2020.
- [5] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 311–318, 1994.
- [6] S. You, J. Hu, U. Neumann, and P. Fox, "Urban Site Modeling From LiDAR," in International Conference on Computational Science and its Applications, pp. 579–588, Springer, 2003.
- [7] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building Rome in a Day," Commun. ACM, vol. 54, p. 105–112, Oct. 2011.
- [8] C. Griwodz, S. Gasparini, L. Calvet, P. Gurdjos, F. Castan, B. Maujean, G. D. Lillo, and Y. Lanthony, "Alicevision Meshroom: An open-source 3D reconstruction pipeline," in Proceedings of the 12th ACM Multimedia Systems Conference - MMSys '21, ACM Press, 2021.
- [9] M. Levoy and P. Hanrahan, "Light field rendering," in Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96, pp. 31–42, ACM Press, 1996.
- [10] C. Fehn, "Depth-image-based rendering (DIBR), compression, and

- transmission for a new approach on 3D-TV,” in *Stereoscopic Displays and Virtual Reality Systems XI*, vol. 5291, pp. 93–105, International Society for Optics and Photonics, May 2004.
- [11] S. Fachada, D. Bonatto, M. Teratani, and G. Lafruit, “View Synthesis tool for VR Immersive Video,” in *3D Computer Graphics*, IntechOpen, Dec. 2021.
- [12] S. Fachada, D. Bonatto, A. Schenkel, and G. Lafruit, “Depth Image Based View Synthesis With Multiple Reference Views For Virtual Reality,” in *IEEE 2018-3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, (Helsinki), pp. 1–4, IEEE, June 2018.
- [13] D. Bonatto, S. Fachada, and G. Lafruit, “RaViS: Real-time accelerated View Synthesizer for immersive video 6DoF VR,” *Electronic Imaging*, vol. 2020, no. 13, pp. 382–1–382–9, 2020.
- [14] J. Barker, S. Martin, R. Guy, J.-E. Munoz-Lopez, A. Kapoulkine, and K. Chang, “Moving mobile graphics,” in *ACM SIGGRAPH 2020 Courses, SIGGRAPH ’20*, (New York, NY, USA), Association for Computing Machinery, 2020.
- [15] P. K. D. Pramanik, N. Sinhababu, B. Mukherjee, S. Padmanaban, A. Maity, B. K. Upadhyaya, J. B. Holm-Nielsen, and P. Choudhury, “Power consumption analysis, measurement, management, and issues: A state-of-the-art review of smartphone battery and energy usage,” *IEEE Access*, vol. 7, pp. 182113–182172, 2019.
- [16] G. Lafruit, D. Bonatto, C. Tulvan, M. Preda, and L. Yu, “Understanding MPEG-I Coding Standardization in Immersive VR/AR Applications,” *SMPTE Motion Imaging Journal*, vol. 128, no. 10, pp. 33–39, 2019.
- [17] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, “View generation with 3D warping using depth information for FTV,” *Signal Processing: Image Communication*, vol. 24, no. 1, pp. 65 – 72, 2009. Special issue on advances in three-dimensional television and video.
- [18] S. Fachada, D. Bonatto, A. Schenkel, and G. Lafruit, “Free Navigation in Natural Scenery With DIBR: RVS and VSRS in MPEG-I Standardization,” in *2018 International Conference on 3D Immersion (IC3D)*, (Brussels, Belgium), pp. 1–6, IEEE, Dec. 2018.
- [19] T. Balogh, T. Forgács, T. Agács, O. Balet, E. Bouvier, F. Bettio, E. Gobetti, and G. Zanetti, “A Scalable Hardware and Software System for the Holographic Display of Interactive Graphics Applications,” *Eurographics (Short Presentations)*, pp. 109–112, 2005.
- [20] D. Bonatto, H. Grégoire, K. Alexander, S. Fachada, and G. Lafruit, “MPEG Immersive Video tools for Light Field Head Mounted Displays,” in *2021 International Conference on Visual Communications and Image Processing (VCIP)*, IEEE, 2021.
- [21] S. Fachada, D. Bonatto, and G. Lafruit, “High Quality Holographic Stereograms Generation using four RGBD Images,” in *Applied Optics*, OSA Publishing, Oct. 2020. ISSN: 1559-128X, 2155-3165.
- [22] S. Fachada, D. Bonatto, M. Teratani, and G. Lafruit, “Light Field Rendering for non-Lambertian Objects,” *Electronic Imaging*, vol. 2021, pp. 54–1, 01 2021.
- [23] S. Fachada, D. Bonatto, Y. Xie, P. Rondao Alface, M. Teratani, and G. Lafruit, “Depth Image-Based Rendering of Non-Lambertian Content in MPEG Immersive Video,” in *2021 International Conference on 3D Immersion (IC3D)*, Oct. 2021.
- [24] M. Segal and K. Akeley, “The OpenGL® Graphics System: A Specification (Version 4.6 (Core Profile) - October 22, 2019)” <https://www.khronos.org/registry/OpenGL/specs/gl/glspec46.core.withchanges.pdf>, 2019.
- [25] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” in *Computer Vision – ECCV 2020* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 405–421, Springer International Publishing, 2020.
- [26] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz, “HyperNeRF: a higher-dimensional representation for topologically varying neural radiance fields,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 6, pp. 1–12, 2021.
- [27] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, “Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines,” *ACM Trans. Graph.*, vol. 38, July 2019.
- [28] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwanakorn, “Nex: Real-time view synthesis with neural basis expansion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8534–8543, 2021.
- [29] J. Fleureau, F. Thudor, G. Briand, R. Tapie, T. Gendrot, and R. Doré, “MIV / MPI on Android.” <https://www.interdigital.com/contributions/mpeg-i-visual-mivmpi-on-android-informative>, 2021.
- [30] G. Chaurasia, A. Nieuwoudt, A.-E. Ichim, R. Szeliski, and A. Sorkine-Hornung, “Passthrough+ Real-time Stereoscopic View Synthesis for Mobile Mixed Reality,” *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 3, no. 1, pp. 1–17, 2020.
- [31] T. Schöps, M. R. Oswald, P. Speciale, S. Yang, and M. Pollefeys, “Real-time View Correction for Mobile Devices,” *IEEE transactions on visualization and computer graphics*, vol. 23, no. 11, pp. 2455–2462, 2017.
- [32] B. Reinert, J. Kopf, T. Ritschel, E. Cuervo, D. Chu, and H.-P. Seidel, “Proxy-guided Image-based Rendering for Mobile Devices,” *Computer Graphics Forum*, 2016.
- [33] Oculus, “Introducing Oculus Air Link, a Wireless Way to Play PC VR Games on Oculus Quest 2, Plus Infinite Office Updates, Support for 120 Hz on Quest 2, and More.” <https://www.oculus.com/blog/introducing-oculus-air-link-a-wireless-way-to-play-pc-vr-games-on-oculus-quest-2-plus-infinite-office-updates-support-for-120-hz-on-quest-2-and-more>, Apr. 2021. Accessed: 28-07-2021.
- [34] The Khronos® Vulkan Working Group, “Vulkan® 1.1.188 - A Specification (with all registered Vulkan extensions).” <https://www.khronos.org/registry/vulkan/specs/1.1-extensions/pdf/vkspec.pdf>, 2021.
- [35] M. Lujan, M. Baum, D. Chen, and Z. Zong, “Evaluating the Performance and Energy Efficiency of OpenGL and Vulkan on a Graphics Rendering Server,” in *2019 International Conference on Computing, Networking and Communications (ICNC)*, pp. 777–781, 2019.
- [36] The Khronos Group, “ Vulkan-Samples/samples/performance/ .” <https://github.com/KhronosGroup/Vulkan-Samples/tree/master/samples/performance>, 2021. Accessed: 2-08-2021.
- [37] J.-E. Munoz-Lopez, “Vulkan on mobile done right.” https://community.arm.com/cfs-file/__key/communityserver-blogs-components-weblogfiles/00-00-00-20-66/2_2D00_mmg2020_2D00_vulkan_2D00_emilio.pdf, 2020.
- [38] “Art Direction for All-in-One VR Performance.” <https://developer.oculus.com/documentation/native/android/po-art-direction/>. Accessed: 10-01-2022.

- [39] The Khronos Group Inc., “The OpenXR Specification.” <https://www.khronos.org/registry/OpenXR/specs/1.0/pdf/xrspec.pdf>, 2021.
- [40] S. Rogge, D. Bonatto, J. Sancho, R. Salvador, E. Juarez, A. Munteanu, and G. Lafruit, “MPEG-I Depth Estimation Reference Software,” in 2019 International Conference on 3D Immersion (IC3D), (Brussels, Belgium), pp. 1–6, IEEE, Dec. 2019.
- [41] D. Mieloch, O. Stankiewicz, and M. Domański, “Depth Map Estimation for Free-Viewpoint Television and Virtual Navigation,” IEEE Access, vol. 8, pp. 5760–5776, 2020. Conference Name: IEEE Access.
- [42] D. Mieloch and A. Dziembowski, “Proposal of IVDE 3.0 [m55751],” ISO/IEC JTC1/SC29/WG11, Dec. 2020.
- [43] D. Bonatto, S. Fachada, and G. Lafruit, “ULB ToysTable.” <https://zenodo.org/record/5055543>, Feb. 2021.
- [44] A. Schenkel, D. Bonatto, S. Fachada, H.-L. Guillaume, and G. Lafruit, “Natural Scenes Datasets for Exploration in 6DOF Navigation,” in 2018 International Conference on 3D Immersion (IC3D), (Brussels, Belgium), pp. 1–8, IEEE, Dec. 2018.
- [45] D. Bonatto, A. Schenkel, T. Lenertz, Y. Li, and G. Lafruit, “[MPEG-I Visual] ULB High Density 2D/3D Camera Array data set, version 2 [m41083],” (Torino, Italy), July 2017.
- [46] “Application SpaceWarp Developer Guide.” <https://developer.oculus.com/documentation/native/android/mobile-asw/>. Accessed: 10-01-2022.

Author Biography

Laurie Van Bogaert obtained her master’s degree of science in Computer Science and Engineering in 2021 from the Université Libre de Bruxelles, Belgium. She is currently pursuing her Ph.D degree on Multi-Plane Image with Transparency and Specularity for 6 degrees of Freedom Virtual Reality. She received a FRIA funding for her Ph.D from the Walloon Region, Belgium. Her research interests are novel view synthesis, real-time rendering, Vulkan, Cuda and OpenXR.

Daniele Bonatto received a computational intelligence software engineering degree in applied sciences from the Université Libre de Bruxelles (ULB, 2016). He is pursuing a PhD program jointly between the ULB and the Vrije Universiteit Brussel (VUB). He works on real-time free-viewpoint rendering of natural scenery with sparse multicamera acquisition setups. Jointly with the Moving Picture Experts Group (MPEG), Bonatto developed the Reference View Synthesis software (2018) and two high-density static and dynamic natural scene datasets.

Sarah Fachada graduated from Ecole polytechnique (France) and Trinity College of Dublin (Ireland) in 2017, majoring in computer science. She is now a PhD student at Université Libre de Bruxelles (Belgium) under a FNRS fellowship, working on acquisition and rendering in light fields and DIBR. Her work explores fields such as rendering with non-pinhole cameras, geometric algebra applications and rendering non-Lambertian objects. Jointly with MPEG, Fachada developed the Reference View Synthesis software in MPEG.

Gauthier Lafruit received the M.Sc. and Ph.D. degrees from Vrije Universiteit Brussel (VUB), in 1989 and 1995, respectively. He is currently an associate professor immersive light field technologies at Université Libre de Bruxelles (ULB), Brussels, Belgium. He works in visual data compression and rendering, participating to compression standardization committees like CCSDS (space applications), JPEG (still picture coding) and MPEG (moving picture coding). His research interests focus on depth image-based rendering, immersive video, and digital holography.