On OpenGL-based View Synthesizer Optimization for Real-time 6DoF Immersive Video Streaming

Presenter: Eun-Seok Ryu (esryu@skku.edu)

Jong-Beom Jeong, Soonbin Lee, Eun-Seok Ryu

Multimedia Computing Systems Lab. (MCSL)

http://mcsl.skku.edu

Department of Computer Education

Sungkyunkwan University

Compliance with IEEE Standards Policies and Procedures

Subclause 5.2.1 of the *IEEE-SA Standards Board Bylaws* states, "While participating in IEEE standards development activities, all participants...shall act in accordance with all applicable laws (nation-based and international), the IEEE Code of Ethics, and with IEEE Standards policies and procedures."

The contributor acknowledges and accepts that this contribution is subject to

- The IEEE Standards copyright policy as stated in the *IEEE-SA Standards Board Bylaws*, section 7, http://standards.ieee.org/develop/policies/bylaws/sect6-7.html#7, and the *IEEE-SA Standards Board Operations Manual*, section 6.1, http://standards.ieee.org/develop/policies/opman/sect6.html
- The IEEE Standards patent policy as stated in the *IEEE-SA Standards Board Bylaws*, section 6, http://standards.ieee.org/guides/bylaws/sect6-7.html#6, and the *IEEE-SA Standards Board Operations Manual*, section 6.3, http://standards.ieee.org/develop/policies/opman/sect6.html

On OpenGL-based View Synthesizer Optimization for Real-time 6DoF Immersive Video Streaming

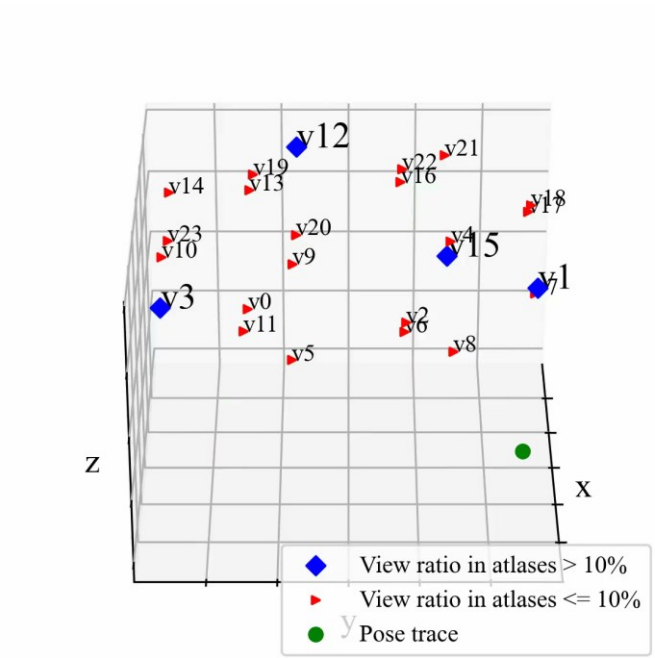
Date: 2021-07-19

Author(s): Jong-Beom Jeong, Soonbin Lee, Eun-Seok Ryu

Name	Affiliation	Phone [optional]	Email [optional]
Jong-Beom Jeong	Sungkyunkwan Univ.		uof4949@skku.edu
Soonbin Lee	Sungkyunkwan Univ.		soonbinlee@skku.edu
Eun-Seok Ryu	Sungkyunkwan Univ.		esryu@skku.edu

View Synthesis in Immersive Video Streaming

- 6DoF immersive video may contain multi-view videos
- Motion parallax needs to be supported in 6DoF environments
  - Locations of users and source view camera may not be the same -> virtual view synthesis is needed
- Real-time view synthesis for 6DoF immersive video streaming

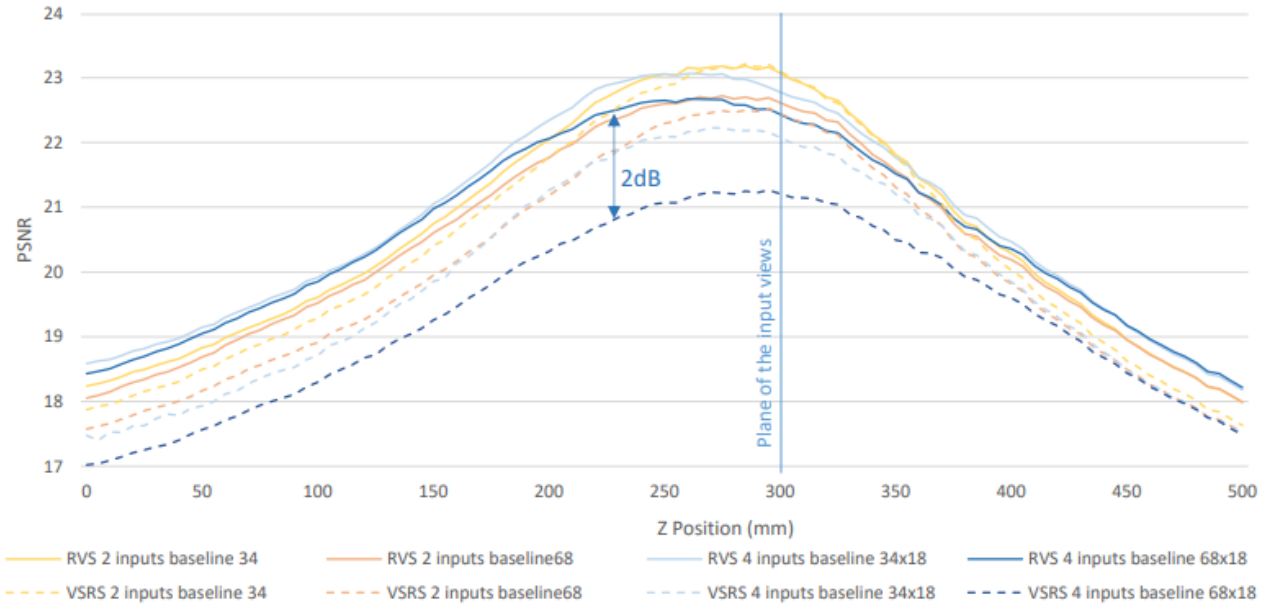


Pose trace movements example

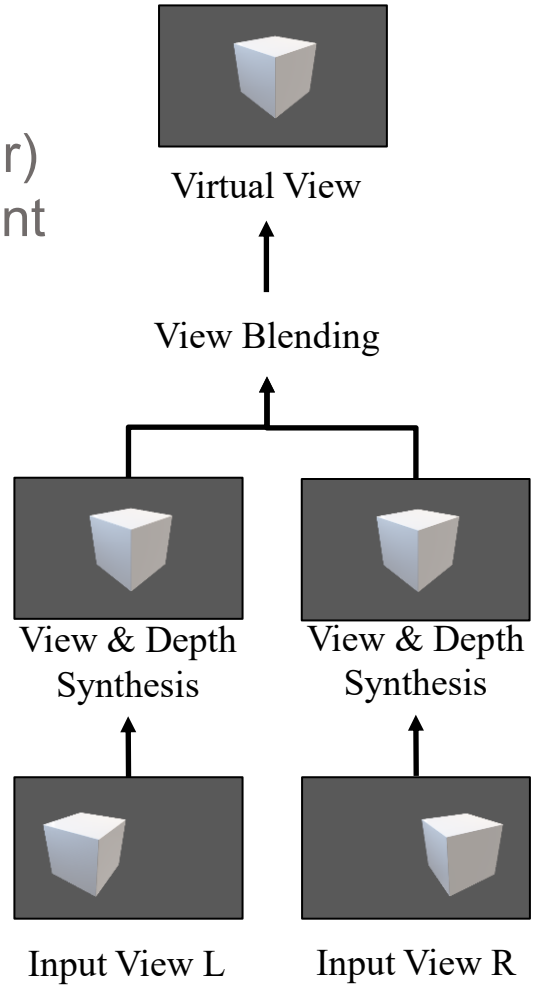
Rendered pose trace example

Reference View Synthesizer

- Reference software of MPEG-I
- Developed by ULB and Philips
- Gets views more than four (max. views in VSRS: four)
- Outperforms VSRS with over 2dB PSNR improvement



VSRS vs RVS PSNR comparison for ULB Unicorn cuboid



Example of view synthesis

Source: S. Fachada, D. Bonatto, A. Schenkel, G. Lafruit,
Free Navigation in Natural Scenery With DIBR: RVS and VSRS in MPEG-I Standardization,
2018 International Conference on 3D Immersion (IC3D), pp. 1-6, 2018.

Reference View Synthesizer (-Cont'd)

- Triangle mesh-based view synthesis
  - Pixel movement between viewpoints of the same world coordinate
  - Input view is divided into triangles with the pixels centers as vertices
  - Use affine transformation
  - Being filled with interpolated colors after affine transformation

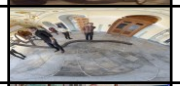
(a) Input view

(b) Obtained view after synthesizing view and depth

Source: ISO/IEC SC29 WG11, Reference View Synthesizer (RVS) manual, 124th meeting of ISO/IEC SC29 WG11, n18068, 2018.

RVS Profiling - Experimental Setup

- Chose nine mandatory immersive video test sequences
- Synthesized all views, with texture, geometry 10bit
- Used Windows 10 desktop with i7-10700(16 threads), GTX1080

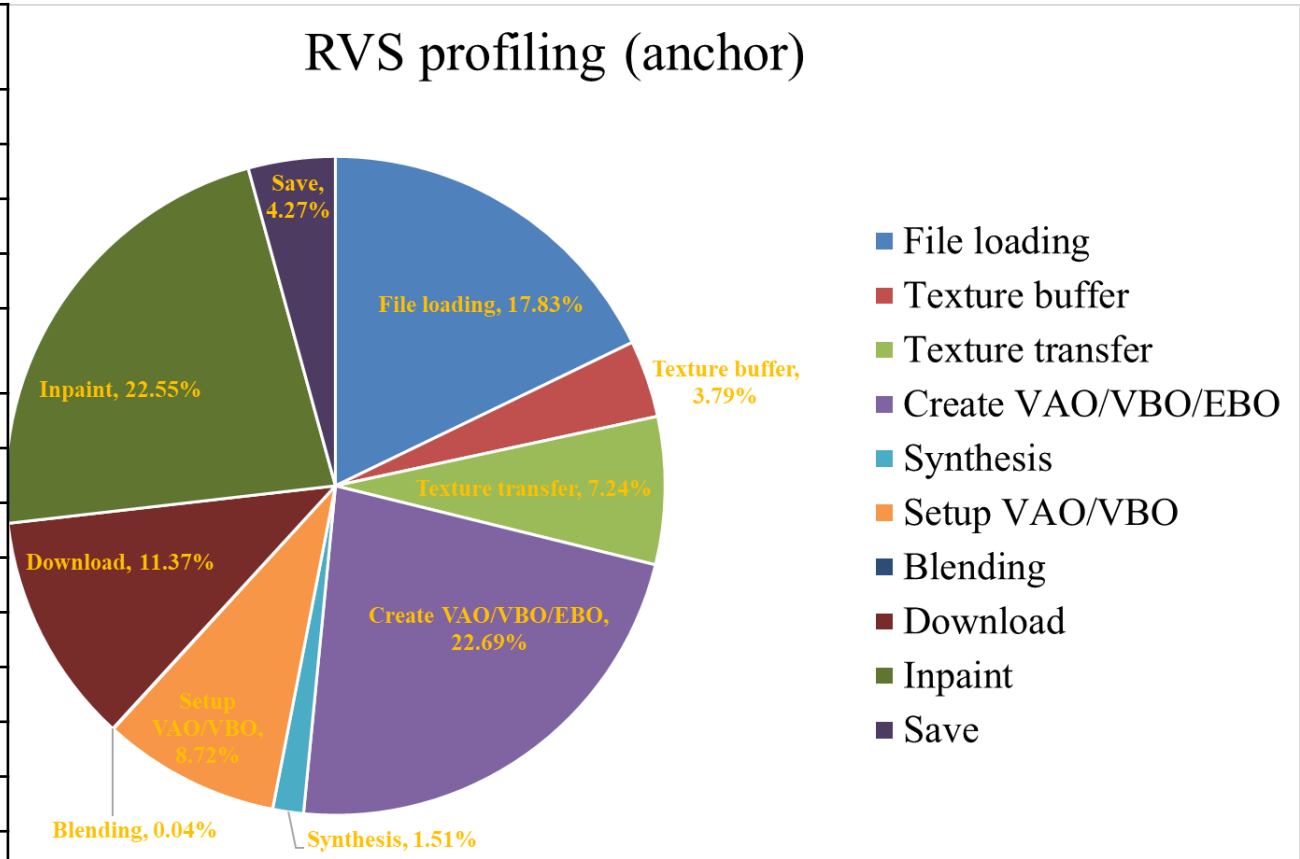
Class (M/O)	Sequence name Resolution	Thumbnail	Type	No. of views	Depth range	Frame count	Frame rate	Bit depth
CG-A (M)	ClassroomVideo 4096 × 2048		ERP	15	[0.8m, inf]	120	30fps	Texture: 10, Geometry: 16
CG-B (M)	Museum 2048 × 2048		ERP	24	[0.5m, 25m]	300	30fps	Texture: 10, Geometry: 16
CG-O (M)	Fan 1920 × 1080		Perspective	15	[0.35m, 12.5m]	97	30fps	Texture: 10, Geometry: 16
CG-J (M)	Kitchen 1920 × 1080		Perspective	25 (5 × 5)	[2.2, 7.2]	97	30fps	Texture: 10, Geometry: 10
NC-D (M)	Painter 2048x1088		Perspective	16 (4 × 4)	-	300	30fps	Texture: 10, Geometry: 16
NC-E (M)	Frog 1920 × 1080		Perspective	13 (13 × 1)	[0.3, 1.62]	300	30fps	Texture: 10, Geometry: 16
NC-P (M)	CarPark 1920 × 1088		Perspective	9 (9 × 1)	[34.5064, 2760.511]	250	25fps	Texture: 10, Geometry: 16
CG-N (M)	Chess 2048 × 2048		ERP	10	[0.1m, 500m]	300	30fps	Texture: 10, Geometry: 16
CG-R (M)	Group 1920 × 1080		Perspective	21	[1.5m, 25.0m]	99	30fps	Texture: 10, Geometry: 16
CG-C (O)	Hijack 4096 × 2048		ERP	10	[0.5m, 25m]	300	30fps	Texture: 10, Geometry: 16
CG-Q (O)	ChessPieces 2048 × 2048		ERP	10	[0.1m, 500m]	300	30fps	Texture: 10, Geometry: 16

MPEG-I immersive video test sequences

RVS Profiling - Experimental Setup

- Following processes of RVS consumed most of resources:
  1) creating VVE, 2) inpainting, 3) file loading, 4) download, 5) transfer
  -> needs to be optimized

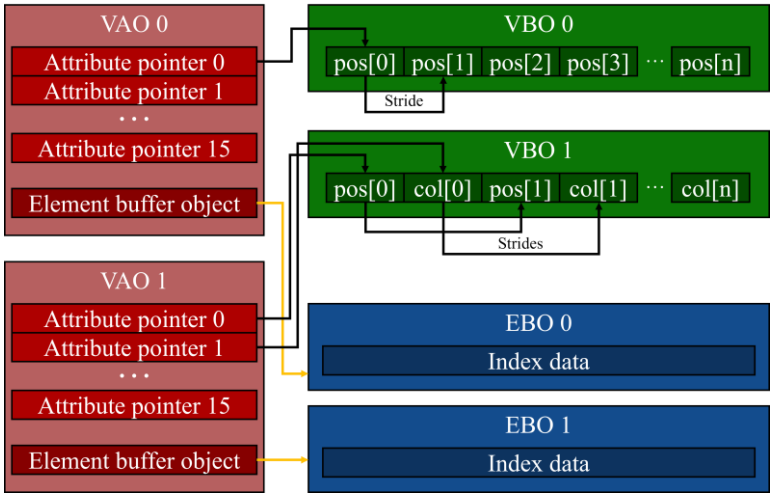
Process	Exe. Time (ms)	Portion (%)
File loading	**83.87**	**17.83%**
Synthesis shader	**180.50**	**35.23%**
Texture buffer	27.93	3.79%
Texture transfer	35.30	7.24%
Create VAO/VBO/EBO	110.15	22.69%
Synthesis	7.11	1.51%
Blender shader	**45.36**	**8.75%**
Setup VAO/VBO	45.20	8.72%
Blending	0.16	0.04%
Result	**210.99**	**38.19%**
Download	55.20	11.37%
Inpaint	136.90	22.55%
Save	18.89	4.27%
Total	**520.71**	**100.00%**



RVS anchor profiling results

RVS Optimization - Reusing VVE

- OpenGL uses three objects:
  1) vertex array object (VAO)
  2) vertex buffer object (VBO)
  3) element buffer object (EBO)
- Current RVS creates them for all frames, all views to:
  create VVE for different resolution of each views
- Reusing VVE in synthesis
  -> 109.73 ms saving
- Reusing VAO/VBO in blending
  -> 45.15 ms saving
- Storing locations of each arrays and transfer them as struct
  -> 13.38 ms saving



VAO, VBO, EBO in OpenGL

RVS Optimization - Reusing VVE (-Cont'd)

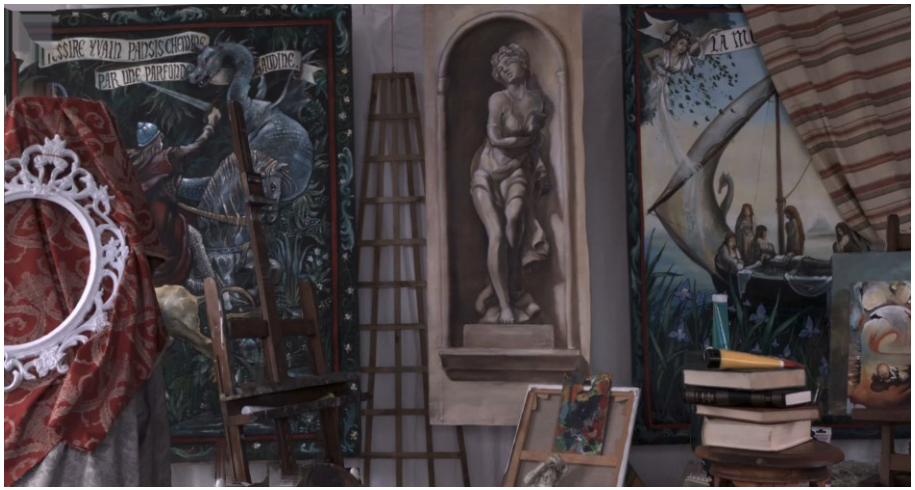
- 36.29% runtime saving compared to RVS anchor (1.92 FPS -> 2.98 FPS)

	Anchor		Reusing VVE	
Process	Exe. Time (ms)	Portion (%)	Exe. Time (ms)	Portion (%)
File loading	**83.87**	**17.83%**	**72.74**	**24.28%**
Synthesis shader	**180.50**	**35.23%**	**58.42**	**17.51%**
Texture buffer	27.93	3.79%	16.61	3.99%
Texture transfer	35.30	7.24%	40.17	12.88%
Create VAO/VBO/EBO	110.15	22.69%	0.42	0.14%
Synthesis	7.11	1.51%	1.22	0.50%
Blender shader	**45.36**	**8.75%**	**0.05**	**0.02%**
Setup VAO/VBO	45.20	8.72%	0.04	0.02%
Blending	0.16	0.04%	0.00	0.00%
Result	**210.99**	**38.19%**	**204.12**	**58.20%**
Download	55.20	11.37%	43.06	14.33%
Inpaint	136.90	22.55%	140.94	36.97%
Save	18.89	4.27%	20.12	6.89%
Total	**520.71**	**100.00%**	**335.32**	**100.00%**

RVS optimization results

RVS Optimization - Reusing VVE (-Cont'd)

- Virtual view synthesis example
- Painter, 16 views, 2048 × 1088 resolution, natural content
- RVS OpenGL anchor: 310.75ms per frame (3.218 FPS)
- Proposed: 203.30ms per frame (4.918 FPS)



Rendered view by RVS anchor
(3.218 FPS)

Rendered view by the proposed method
(4.918 FPS)

Conclusion

- Optimization points (Goal: over 30 FPS)
  - Parallel file loading using multi-thread
  - Multiple texture buffer allocation for each input view
  - Reusing buffer of conversion from OpenGL to OpenCV (download)
  - OpenGL implementation of inpainting
  - Saving function optimization
- Future Work
  - OpenGL implementation of inpainting
  - System integration with Oculus renderer, streaming module
  - Moving to the latest version of TMIV