Hybrid Ray/Voxel-Tracing Fixed Capacity Grids

Baktash Abdollah-shamshir-saz
baktash@toomuchvoltage.com
www.toomuchvoltage.com

TOOMUCHVOLTAGE SOFTWARE INC.

1. Introduction

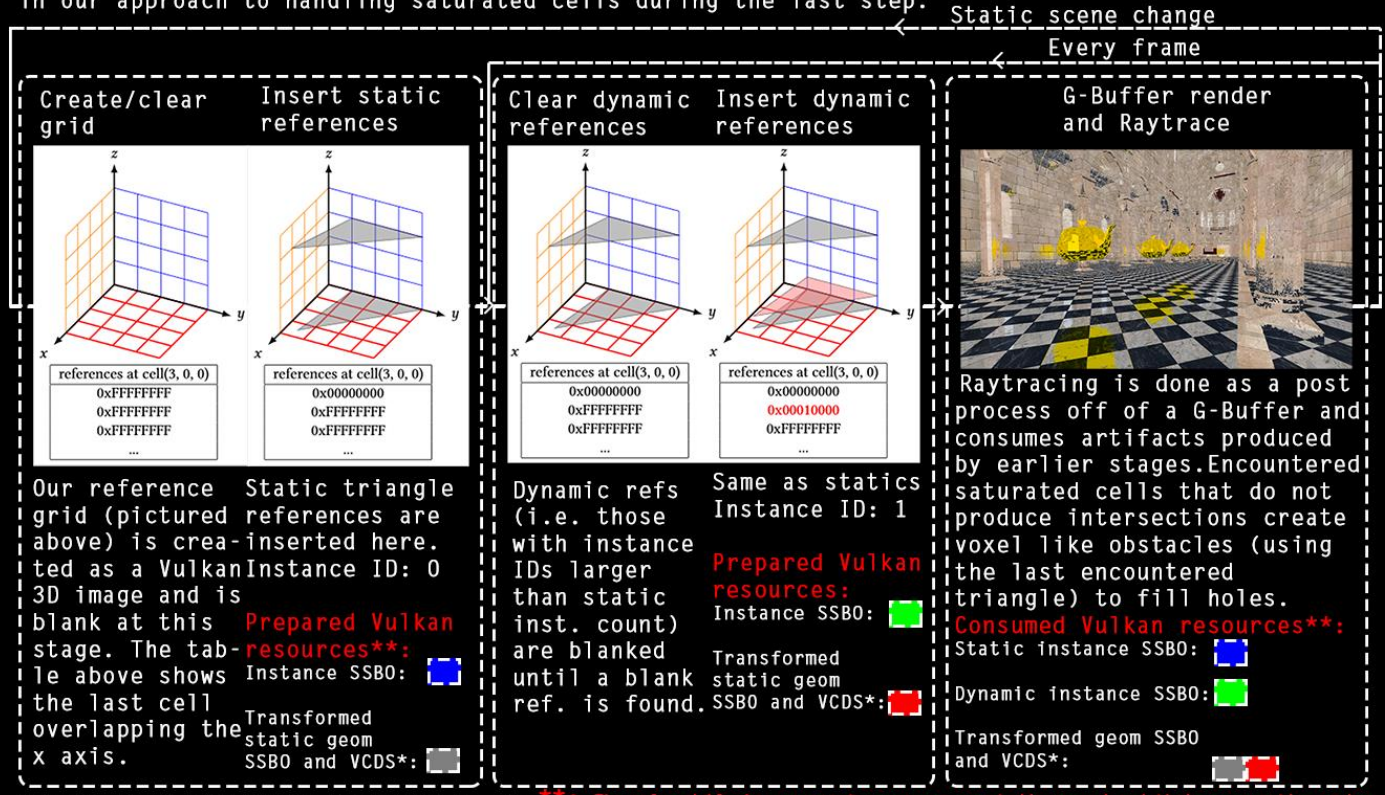
Widely available hardware accelerated ray-tracing such as RTX, has seen adoption by many real-time and interactive applications such as games. However, showcases such as CryTek's Neon Noir have demonstrated shader-based alternatives in some limited form [Kajalin 2019]. This poster explores this approach, extends it to alleviate its dependence on geometry proxies and is accompanied by a demo showcasing this extension.

2. Overview

Our technique has three components: **scene build**, **scene update** and **scene trace**. Scene build generally happens once (unless static geometry is modified due to events such as procedural destruction). Scene updates clear and update dynamic geometry present within the scene and the trace step ray traces the scene. Complete implementation details are provided in the abstract paper hosted on our website.

3. Algorithm details:

Below is an example with two static triangles (black) and one dynamic triangle (red) representing two geometry instances in total. The novelty of our algorithm resides in our approach to handling saturated cells during the last step.



VCDS*: Variable Count Descriptor Set

**: The colored blocks represent resources created/consumed and their consumption order represents the order in which their contents are arranged in memory during trace.

3.1 Scene parameters

Our scene parameters are currently chosen manually for each test scene. They are provided in Table 1. voxelDim is in cubic inches in virtual world. maxTrisPerCell denotes the number of triangles needed to saturate a cell.

Table 1: Scene Parameters

Scene	maxTrisPerCell	voxelDim	grid mem usage (MBs)
Fire Place	30	0.5	567
Sponza	10	1.0	522
Sibenik	10	1.0	821

4. Results

Listed below are results for walkthroughs of scenes with 1 (Fireplace) and 3 (Sponza and Sibenik) animated Utah teapots. Min and max are attributable to slowest and fastest frustums (largely related to average ray travel distance). On average frames costed from 15 to 23 milliseconds across all scenes.

Scene	Ours on RadeonVII								
	build(ms)			update(ms)			trace(ms)		
	min	max	avg	min	max	avg	min	max	avg
Fireplace	18.53	20.47	18.84	0.58	34.49	0.77	0.07	46.32	18.96
Sponza	241.60	245.45	243.04	0.92	2.40	1.04	2.59	34.38	14.79
Sibenik	42.06	52.21	43.20	1.05	8.02	1.24	0.06	47.47	17.00

Scene	Ours on RTX 2080Ti								
	build(ms)			update(ms)			trace(ms)		
	min	max	avg	min	max	avg	min	max	avg
Fireplace	18.44	61.30	27.66	1.06	2.81	1.38	1.02	22.59	6.58
Sponza	148.20	169.18	153.74	2.31	8.39	3.03	0.11	19.81	5.72
Sibenik	31.94	121.80	47.47	2.38	15.49	3.27	0.09	24.43	6.98

Scene	RTX on RTX 2080Ti								
	build(ms)			update(ms)			trace(ms)		
	min	max	avg	min	max	avg	min	max	avg
Fireplace	7.03	9.07	7.55	0.09	0.68	0.14	0.55	3.37	1.39
Sponza	12.66	14.45	13.36	0.09	0.65	0.15	1.25	5.71	2.17
Sibenik	4.62	5.75	5.01	0.09	0.58	0.12	0.92	4.26	1.49

5. Acknowledgements

The author would like to thank Matthäus G. Chajdas of AMD for assisting in debugging some of the issues encountered on AMDRadeon VII. Special thanks are extended to Azam Khan for reviewing this writing.

6. References

Ron Frölich Vladimir Kajalin. 2019. How we made Neon Noir - Ray Traced Reflectionsin CRYENGINE and more! https://www.cryengine.com/news/view/how-we-made-neon-noir-ray-traced-reflections-in-cryengine-and-more. Accessed: 2019-12-10.