EUROGRAPHICS CONFERENCE 2024

Comparing NVIDIA RTX and a Novel Voxel-Space Ray Marching Approach as Global Illumination Solutions

O. Erlich[1], S. Aristizabal[1], L. Li[1], B. Woodard[2] , I. Humer and C. Eckhardt[1]
[1] Cal Poly San Luis Obispo, USA, [2] Brown University, USA

PROBLEM

In this presentation, we investigate the performance—as well as the quality difference—between the state of the art NVIDIA DXR ray tracing pipeline and a voxelspace ray marching (VSRM). In order to maintain an acceptable quality image outcome, as well as frame-rate, for tested low numbers of rays from one to 32, we use a simple denoiser. We show a similar quality outcome and less progressive dependency on the number of rays for VSRM compared with DXR.

OVERVIEW

Due to increasing GPU processing power, real-time ray tracing becomes feasible for games and graphic intensive real-time applications with an underlying acceleration structure for fast ray-triangle collision [1,2,3]. NVIDIA DXR is using a two-level acceleration structure to find the corresponding triangles of a given geometry for every ray. The recent Vulkan API has a similar approach to the acceleration structure [4,5]. Ray tracing still has a high computational demand, and for each pixel in the output-image, only a small number of rays can be computed while still maintaining real-time frame-rates. To compensate for the noisy outcome, denoisers are used. We investigate a VSRM approach where geometry density only affects voxelization, not ray-collision detection. The question is whether VSRM can match the global illumination quality of DXR and the performance differences due to its different approach of ray-geometry collision detection. Our VSRM is based on voxel-cone tracing, but we use ray marching to find the ray-geometry collision due to 1) it's higher collision resolution, 2) the linear scaling of computational time due to a static raymarch distance progression (over all rays), and 3) the necessity of a denoiser in DXR as well as VSRM for a better comparison.

METHODOLOGY

The goal is to investigate the performance and quality of the ray tracing acceleration structure of the NVIDIA RTX platform as opposed to a VSRM solution for generating indirect lighting as part of a global illumination solution. Two versions of a graphics program with a modified version of the Crytek Sponza scene are created, one built on top of the NVIDIA Falcor framework for DirectX12 for access to DXR (RTX), and one built on top of OpenGL 4.5 (Voxel Ray Marching). Our VSRM approach voxelizes the scene based on three different camera positions, and in a final render-pass employs ray marching from each visible screenspace pixel to accumulate indirect light. It is important to note we do not attempt to optimize the voxelization stage

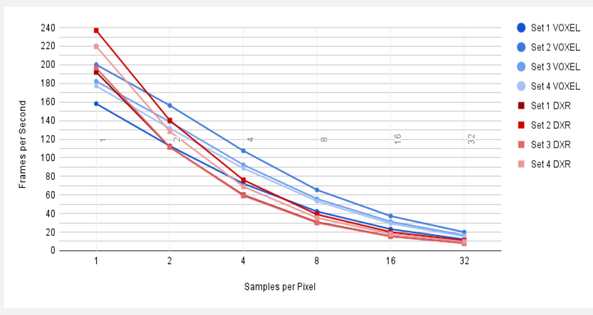
1. Voxelize the 3D scene with back-face culling and depth test disabled, orthographically render the scene from three different perspectives (e.g -x, -y, -z). Calculate direct lighting and shadows for each rasterized fragment. Using fragments' world positions, calculate the average color of each voxel and store it in a 3D texture.
2. Calculate indirect lighting: Raymarch in voxel-space from each fragment's world position in random directions generally aligned with the fragment's normal vector. If the raymarch hits a voxel, add a percentage of the voxel's color to the result; the weight of the contribution is based on the number of samples and the distance falloff. Falloff = 1.0 − (HitDistance/MaxRayDistance). Denoise the ray marching results using any generic image denoiser. In our solution, we use a rudimentary Gaussian blurdenoiser with normal and depth testing for neighboring pixels. Calculate and store direct lighting and shadows.Render an image to the screen that combines the original colors, direct lighting, and ray marching information using the following formula (we define multiplication of two vectors as a component-wise multiplication). Final = DirectLighting + (Original ∗ RayMarching)

For the **DXR approach**, we use the NVIDIA RTX platform for direct and indirect lighting calculations. 1. Raytrace from the fragment world-space position towards the world-space light position. This is known as a "shadow ray". 2. If a "shadow ray" misses all geometry, evaluate the direct lighting information at the fragment and store in a texture. Else, the point is in shadow; thus, direct lighting contribution is zero. 3. From each fragment, trace a set of indirect lighting rays in random directions generally aligned with the initial point's normal. 4. For each indirect ray hit, add the direct lighting to the indirect lighting result. The weight of the contribution is based on the number of samples and a distance falloff described above. 5. Denoise ray tracing results using any generic image denoiser. Our solution uses a rudimentary Gaussian blur denoiser with normal and depth testing for neighboring pixels. 6. Combine direct lighting and denoised ray tracing results.

RESULTS

Performance metrics were captured using an RTX 3080 GPU at 1080p resolution. Each approach was tested in four different locations in the scene using 1, 2, 4, 8, 16, and 32 rays per pixel, with the denoiser turned off. Denoising was disabled as it is generic and any denoiser of varying performance could be substituted.

In the qualitative assessment figure to the right bottom, the qualitative differences between VSRM and DXR can be seen. While both approaches are relatively similar regarding indirect light intensity and reach, VSRM sampling from voxels, which are set-sized volumes of average light, results in more light accumulation as well as less pronounced indirect light colors.
As is illustrated in the measurements to the right, VSRM has a much more linear-leaning progression with increasing number of rays compared to DXR. This is due to the screen-space voxelization as well as an in average expected ray-collision distance with upper limit. As our voxelization stage is not optimized, note, that frame-rates on VSRM can be improved based on different voxelization approaches and API capabilities. With increasing number of rays, DXR falls below 60 fps at 4 rays. These results strongly indicate VSRM has a better scalability than the DXR ray tracing for reaching high sample counts.



AFFILIATIONS

California Polytechnic State University, San Luis Obispo
BROWN

REFERENCES

[1] CRASSIN C., NEYRET F., SAINZ M., GREEN S., EISEMANN E.: Interactive indirect illumination using voxel cone tracing. In Computer Graphics Forum (2011)
[2] GARCÍA A., MURGUIA S., OLIVARES U., RAMOS F. F.: Fast parallel construction of stack-less complete lbvh trees with efficient bit-trail traversal for ray tracing. In Proceedings of the 13th ACM SIGGRAPH international conference on virtual-reality continuum and its applications in industry (2014)
[3] KALLWEIT S., CLARBERG P., KOLB C., DAVIDOVI ˇC T., YAO K.-H., FOLEY T., HE Y., WU L., CHEN L., AKENINE-MÖLLER T., WYMAN C., CRASSIN C., BENTY N.: The Falcor rendering frame-
work, 3 2022.
[4] LIU E., LLAMAS I., KELLY P., ET AL.: Cinematic rendering in ue4 with real-time ray tracing and denoising. In Ray Tracing Gems.
[5] MOON B., JUN J. Y., LEE J., KIM K., HACHISUKA T., YOON S.-E.: Robust image denoising using a virtual flash image for monte carlo ray tracing. In Computer Graphics Forum (2013)