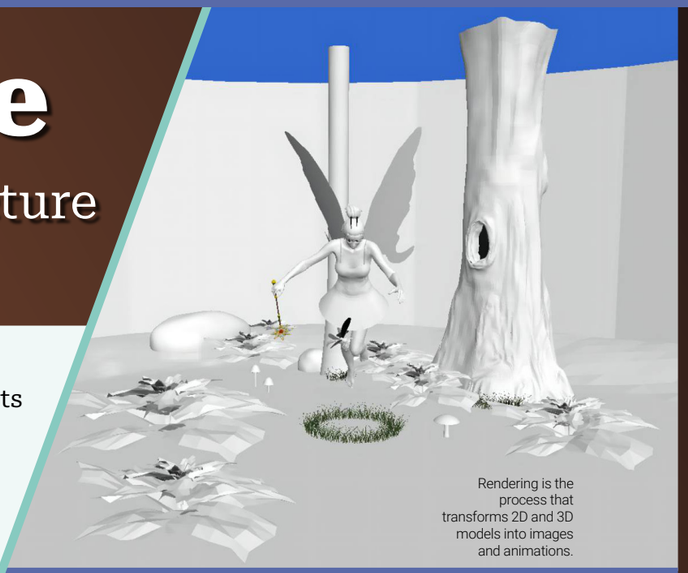
RT Engine

A ray-tracing architecture for mobile devices

- Animation, simulators, video games, architectural design, and TV special effects employ rendering to generate images and animations.
- Rendering algorithms fall into two groups: rasterization and ray tracing.
- Producing a high-quality visual experience is challenging for rasterization but realisable for ray tracing.
- Ray tracing using mobile devices is difficult due to its hardware, computing power, and memory bandwidth requirements.
- Run Yan and colleagues have designed an innovative ray-tracing engine that accelerates ray tracing, making it accessible for mobile platforms.



Rendering is the process that transforms 2D and 3D models into images and animations.

Animation, simulators, video games, and architectural design all use rendering.

Animation, simulators, video games and special effects for TV and movie employ rendering, the process that transforms 2D and 3D models into images and animations. Rendering is a fundamental area of computer graphics, and its algorithms fall into two categories: rasterization and ray tracing.

Rasterization

Rasterization involves mapping geometry in the scene into pixels. Onscreen 3D models of objects are created using a mesh of virtual triangles, or polygons, that create 3D models of objects. Computer algorithms convert the triangles into pixels on a 2D screen. Rasterization is very effective in hardware acceleration and most graphics processing units (GPUs) currently use rasterization to generate 3D photos.

Ray tracing

Ray tracing generates photorealistic 3D images. Modern movies, graphics applications, and video games rely on ray tracing to generate and enhance special effects. Ray-tracing techniques generate high-quality 3D images by simulating the optical properties of light, such as reflection, shadow, and refraction. In nature, a ray of light is emitted from a light source and travels until its progress is interrupted by a surface. Ray-tracing algorithms imitate this, extending rays into a scene and then bouncing them off surfaces, towards light sources to approximate the colour value of pixels and build the image.

> **Modern movies, graphics applications, and video games rely on ray tracing to generate and enhance special effects.**

A more efficient hardware architecture for ray tracing

Master's student Run Yan and colleagues at the National University of Defense Technology, Changsha, China, have designed a more efficient hardware architecture for ray tracing. Their innovative RT engine (ray-tracing engine) is a graphics accelerator that accelerates ray tracing, making it accessible for mobile platforms.

Yan describes the overall architecture of the new RT engine. The research team chose to create domain specific architecture. This hardware-centric approach tailors the architecture to the individual problem. It offers performance and efficiency gains, speeding up applications.

The researchers employ the bounding volume hierarchy (BVH), the standard acceleration structure for ray-tracing algorithms. (The BVH decomposes the objects in the scene into smaller object sets).

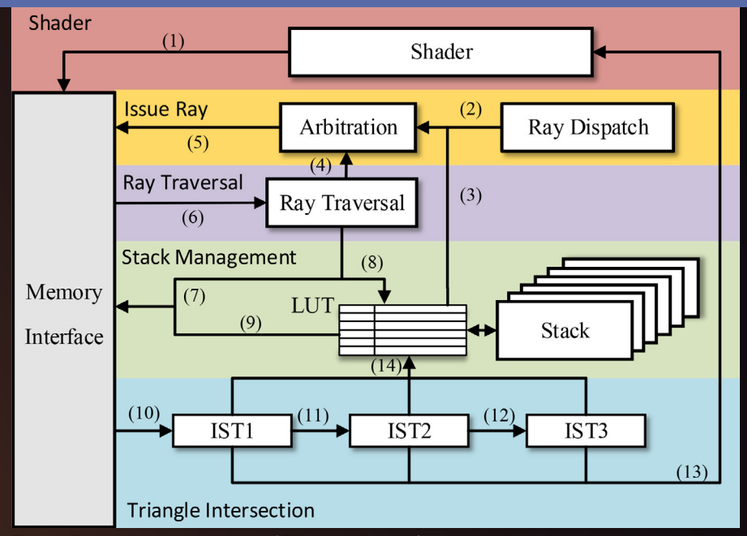


Figure 1. The overall system architecture of RT engine. It includes five parts: the shader, issue ray, ray traversal, stack management, and triangle intersection.

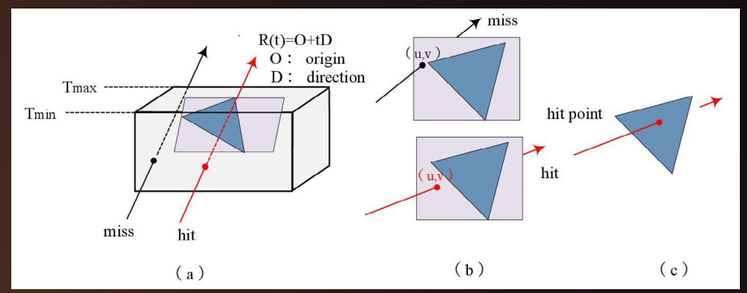


Figure 2. Triangle intersection test. (a) Ray-plane test, (b) barycentric test, and (c) final hit point calculation.

The BVH traversal algorithm takes up less memory, therefore reducing hardware and design costs.

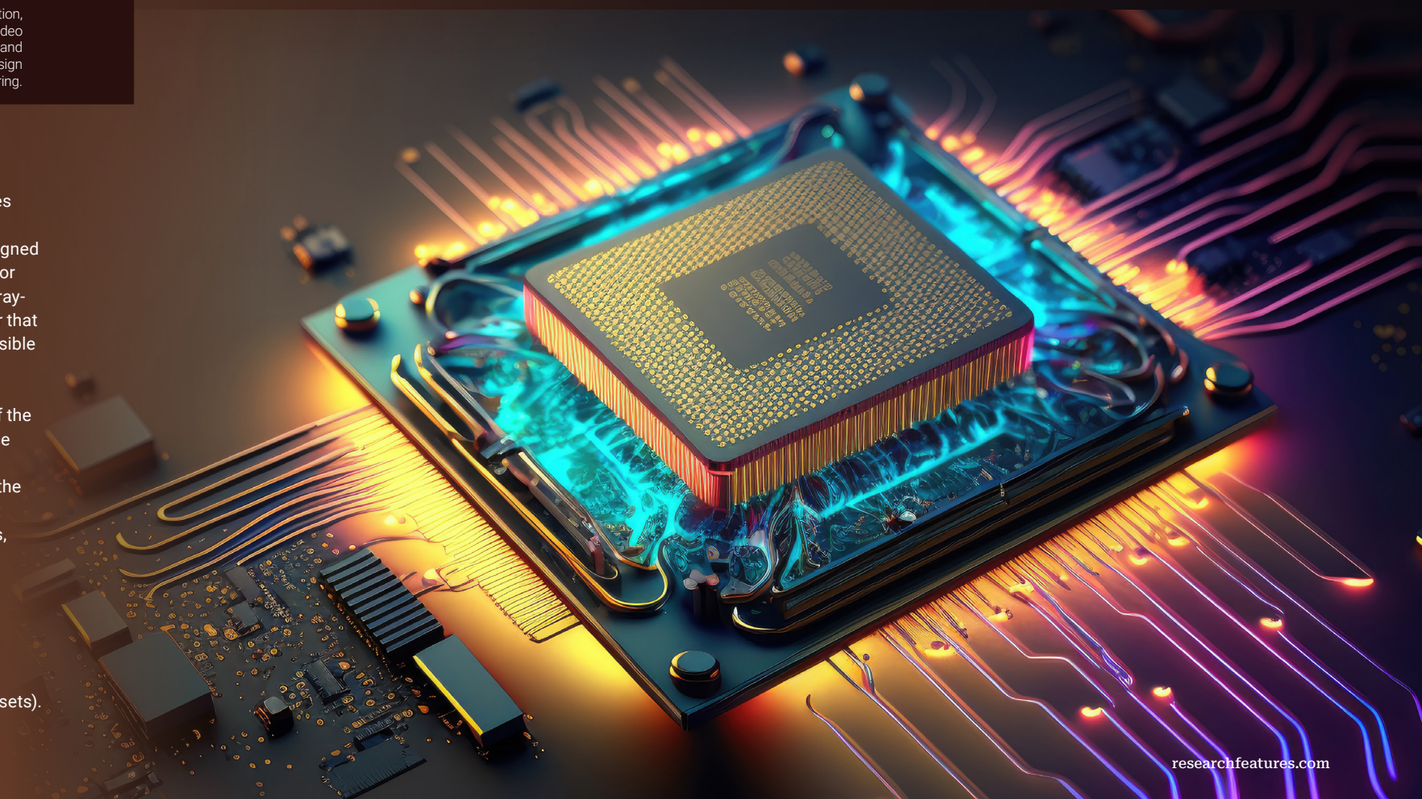
Yan and colleagues use per-ray traversal that allows each ray to traversal independently. They chose triangles as the primitive type. They also employ first-hit traversal for computing the radiance at a shading point. This widely used technique finds the nearest object to the origin of a ray.

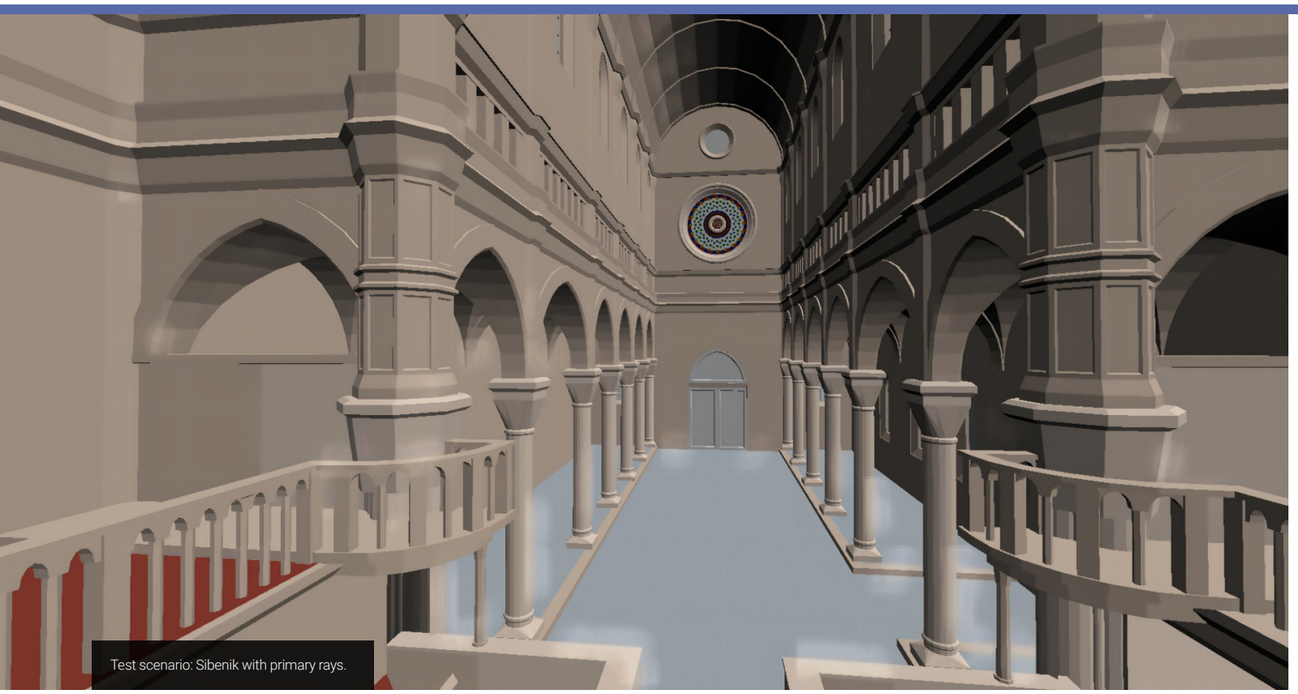
RT engine architecture

The RT engine is made up of the memory, shader, issue ray, ray traversal, stack management, and triangle intersection (figure 1). The shader provides rays, primitives and BVHs data for the system. The issue ray regulates the rays' data input according to various access requests. The ray traversal unit completes the connection between rays and BVHs.

Multiple stacks

The researchers decided to store the RT engine's data in multiple stacks. A stack is a last-in first-out data structure, where new data is placed on the top of the stack and data is taken from the top when it is removed. For instance, an internet browser's back button stores URLs in a stack. If there is only one stack in the system, then only one ray can be processed, so to achieve multiple ray parallel processing, many stacks are used to save information from multiple rays.





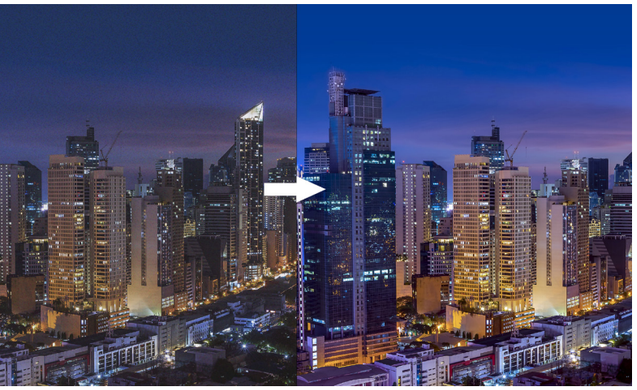
Test scenario: Sibenik with primary rays.

This design of multiple Stack Management units ensures that multiple rays are processed in parallel. It also sets a stack cache to store the data in stacks when there is insufficient storage space in multiple stacks, effectively releasing the excessive hardware storage resources used by the stacks. The RT engine also adopts a data prefetching mechanism to set caches and improving performance. This makes reading data more efficient.

Three-phase break for intersection

The triangle intersection unit carries out the three-phase break for intersection tests for the intersection of rays and triangles. It carries out three tests: the ray-plane test to find out if the ray and triangle intersect, if so it's a 'hit'. The barycentric test determines if the hit point is inside the triangle. The third phase is the final hit-point calculation. The simulation results show this method can effectively improve performance.

The RT engine can carry out efficient ray-tracing processing with limited resources, making it suitable for mobile devices.



Approximation method for reciprocal

The researchers decided to use the approximation method for the reciprocal, which combines parabolic synthesis and second-degree interpolation. Compared to other methods, this approach converges faster and has a smaller chip area. The reciprocal unit tested for every possible input. The max error is $1.18358967 \times 10^{-7}$ ($\approx 2^{-23.01}$) which is smaller than the machine epsilon (Є).

A ray-tracing engine for mobile devices

Experimental simulations were performed to evaluate the efficiency of the RT engine. Results revealed that this novel architecture's performance in terms of MRPS/mm^2 (million rays per second per millimetre) is 2.4 times greater than the best results to date for ray tracing with dedicated hardware. These results demonstrate that the RT engine can carry out efficient ray-tracing processing with limited resources, making it suitable for mobile devices.

Personal response

What initially sparked your interest in computer graphics?

Demand for computer rendering has increased in the past decade. Rasterization-based graphics rendering has been unable to meet our needs. Ray tracing based on physical principles has excellent potential in meeting the new generation of realistic graphics rendering and is expected to become the next generation of mainstream real-time 3D graphics rendering. Our group has found the hot spot of this research and is committed to promoting the development of computer graphics in the field of rendering using customised design.

What posed the greatest challenge in developing the RT engine?

In the research process, the most challenging issues are the selection of algorithms and the design of hardware architecture. First, there is much research on the algorithms based on ray tracing. We refer to the relevant literature, summarise and compare them, and finally determine the standard BVH algorithm for implementation. Then, due to the inherent characteristics of the algorithm, such as many branches and a large amount of computation, the architecture design is a heavy and complex matter. We have carried out a detailed design to make it more applicable to the mobile field.

What are your plans to extend this research?

Because the existing GPU and our design can support the calculation of ray traversal and triangle intersection tests in ray tracing, the support for dynamic construction of accelerated data structure could be better. One of the challenges of real-time ray tracing is dynamic scene support. Whenever scene primitives change, the spatial data structure used to speed up rendering is reconstructed or updated. Therefore, we plan to add the hardware to build BVH dynamically and combine it with the existing RT engine to realise real-time ray tracing.

What does a typical day at the National University of Defense Technology involve for you?

The National University of Defense and Technology has achieved a large number of independent innovation achievements represented by the 'Tianhe' series of supercomputer systems, 'Beidou' satellite navigation and positioning system key technologies, 'Tiantuo' series of micro-nano satellites, laser gyro, ultra-precision machining, maglev train, etc. The university has an intense academic atmosphere, and researchers work collaboratively in their research. I would like to thank my tutor Professor Libo Huang for his careful guidance and help with my academic research and paper writing. Every day we strive for scientific research and academic progress.

Details

e: libohuang@nudt.edu.cn
: Yaa-Ruu

Bio

Run Yan is a Master's student at the National University of Defense Technology. He specialises in theoretical and experimental computer architecture and computer graphics image processing, in particular ray-tracing acceleration.

Funding

This study was supported in part by grants from the National Natural Foundation of China (grant number 61872374 and 62102433).

Collaborators

- Professor Libo Huang
- Assistant Professor Hui Guo
- Dr Yashuai Lü

Further reading

Yan, R, et al, (2022) RT engine: An efficient hardware architecture for ray tracing, *Applied Sciences*, 12(19), 9599.

Yan, R, et al, (2023) MMsRT: A hardware architecture for ray tracing in the mobile domain, *Journal of Circuits, Systems, and Computers*, 32(11), 2350192.