A Standard API for Particle Tracing

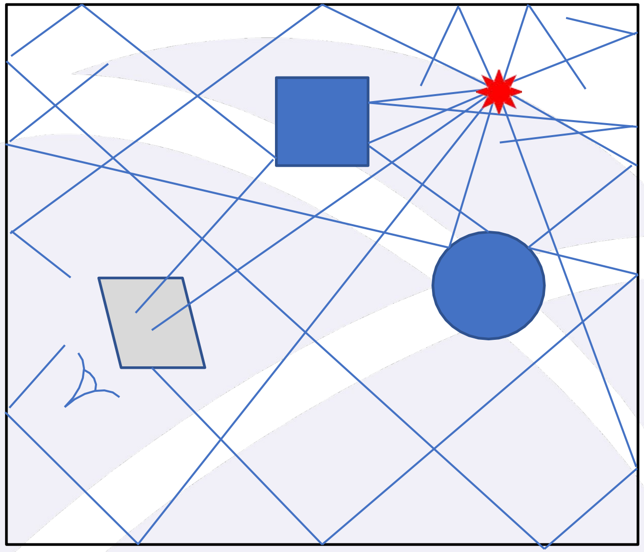
Pascal Grosset, CCS-3

September, 14

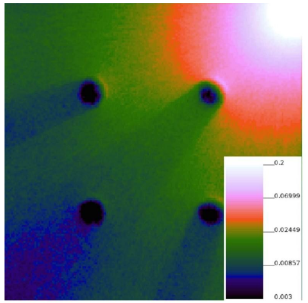
'LA-UR-21-29041

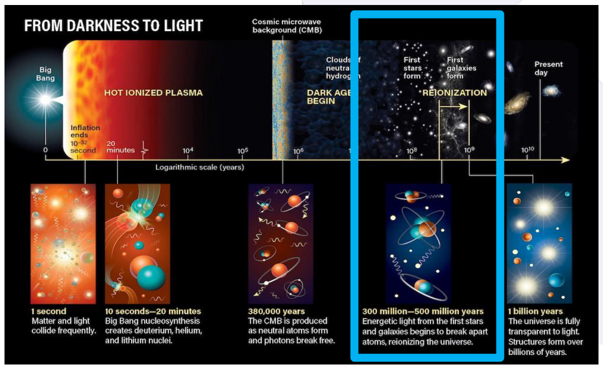
Particle Tracing

- Aim:
  - Follow particles (electrons, neutrons, photons, …) as they interact with their environment



- Uses:
  - Movie & Games industry (photons)
  - Scientific Applications: e.g.
    - Ionization of the universe
    - MCNP





1 second Matter and light collide frequently.

10 seconds—20 minutes Big Bang nucleosynthesis creates deuterium, helium, and lithium nuclei.

380,000 years The CMB is produced as neutral atoms form and photons break free.

300 million—500 million years Energetic light from the first stars and galaxies begins to break apart atoms, reionizing the universe.

1 billion years The universe is fully transparent to light. Structures form over billions of years.

https://astronomy.com/magazine/news/2021/01/the-beginning-to-the-end-of-the-universe-the-cosmic-dark-ages

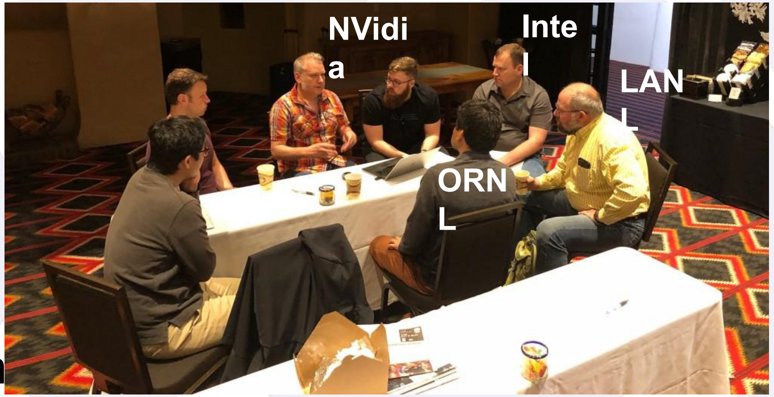
Los Alamos NATIONAL LABORATORY

Why this project? Why Now?

- There is interest for scientific ray tracing
  - SOLAR Ray tracing initiative
  - Intel
  - Nvidia

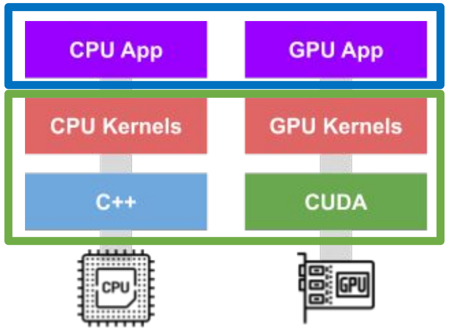
- There is interest in common ray tracing API
  - ANARI for common rendering API

- Goal: Leverage the expertise of the graphics world to help scientists focus on the application code



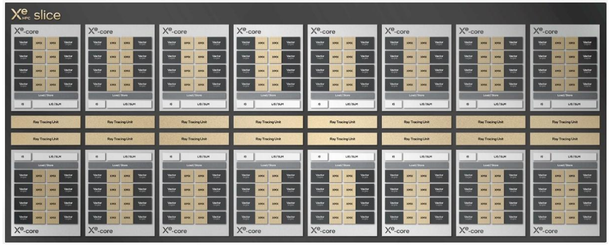
Problem Statement

- Scientific Applications Design:



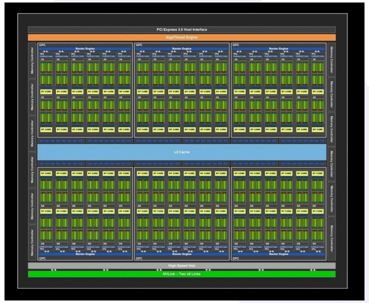
Requires Domain knowledge

Requires System knowledge



GPU with dedicated Ray Tracing (RT) Cores

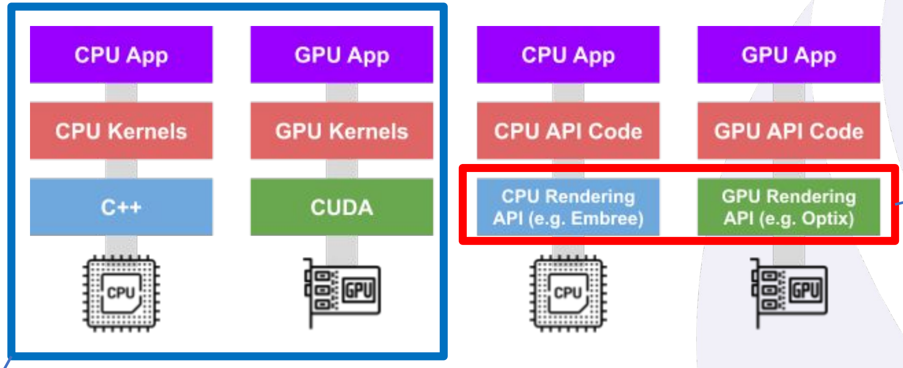
http://3s81si1s5ygj3mzby34dq6qf-wpengine.netdna-ssl.com/wp-content/uploads/2021/08/intel-arch-day-xe-hpc-slice.jpg



https://images.idgesg.net/images/article/2018/09/lkdfghpzhx-17-100771740-orig.jpg?auto=webp

Problem Statement

- Scientific Applications Design:



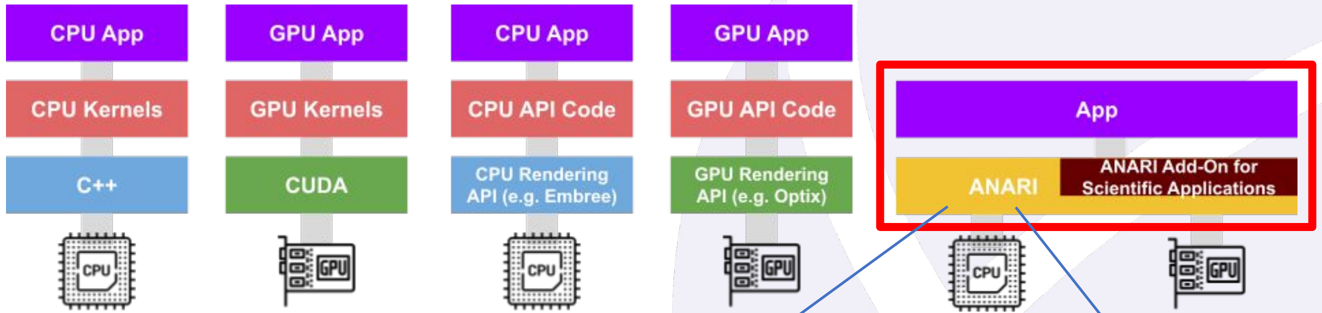
Lot of expertise in the graphics world on this!

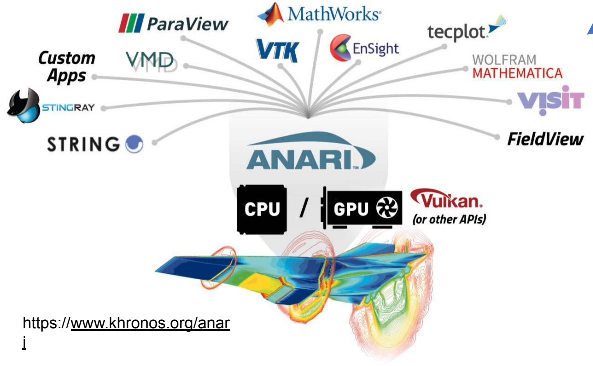
Current Applications

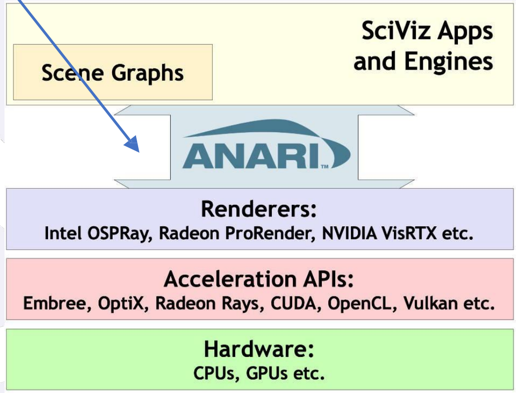
1. **Continuously updated optimized code for each platform.** Companies such as Intel and NVidia expend great effort to ensure that their APIs are optimized for the hardware that they release. Being able to leverage these at no cost in particle tracing code would effortlessly speed up code.

Proposed Solution

- Scientific Applications Design:



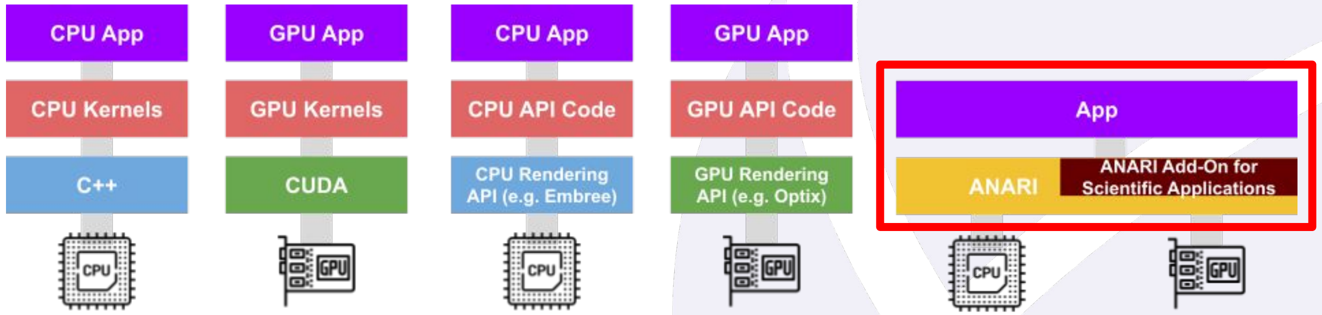


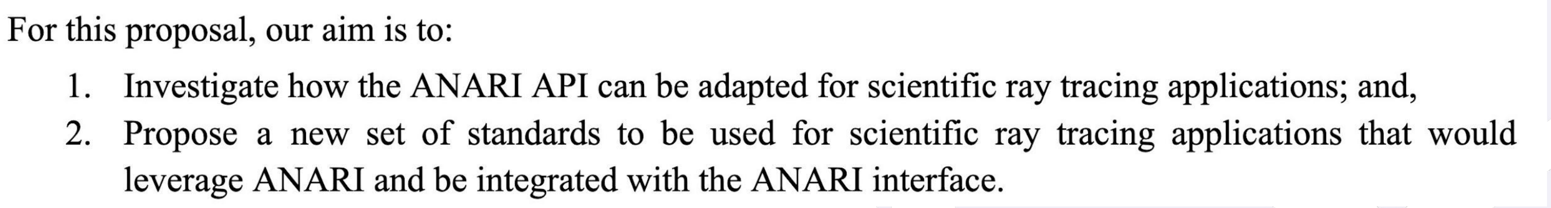

SciViz Apps and Engines
Scene Graphs
ANARI
Renderers:
Intel OSPRay, Radeon ProRender, NVIDIA VisRTX etc.
Acceleration APIs:
Embree, OptiX, Radeon Rays, CUDA, OpenCL, Vulkan etc.
Hardware:
CPUs, GPUs etc.

https://www.khronos.org/anari

Proposed Solution

- Scientific Applications Design:



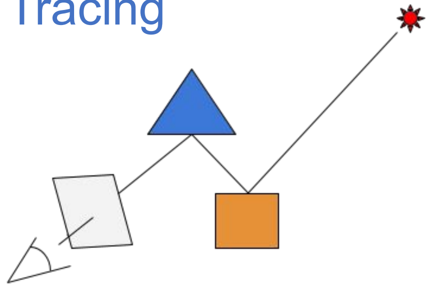

For this proposal, our aim is to:

1. Investigate how the ANARI API can be adapted for scientific ray tracing applications; and,
2. Propose a new set of standards to be used for scientific ray tracing applications that would leverage ANARI and be integrated with the ANARI interface.

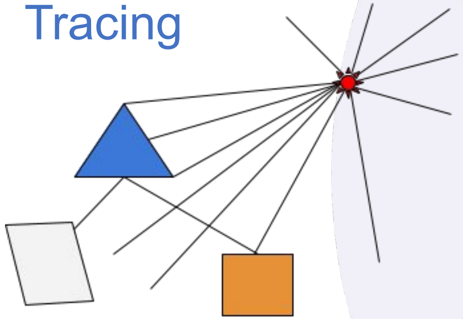
Approach[i]

1. Analysis of Particle Tracing as needed by scientists compared to Computer Graphics needs

Backward Ray Tracing



Forward Ray Tracing



Needed:

-Emit rays from source

- No need for a screen/framebuffer

- CSG for defining geometry

- Tally intersections

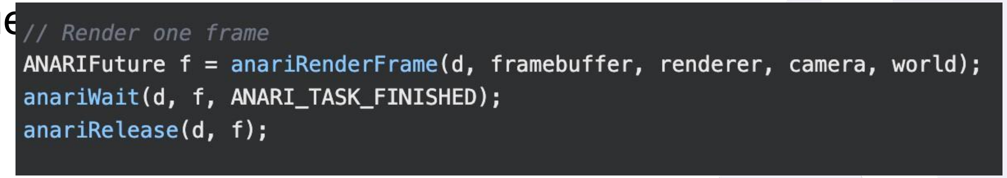
Not Needed:

-Textures

-shadow rays

Approach(ii)

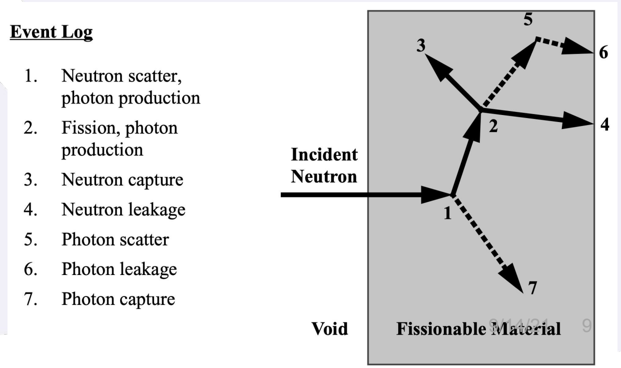
1. Implement a use case using ANARI “Reference” interface
   - Issue



```
// Render one frame
ANARIFuture f = anariRenderFrame(d, framebuffer, renderer, camera, world);
anariWait(d, f, ANARI_TASK_FINISHED);
anariRelease(d, f);
```

     - Rays can only be generated from cameras
     - Output is only through the framebuffer

   - Production and tracking of new particles
   - Tallying of interactions, fluence, …

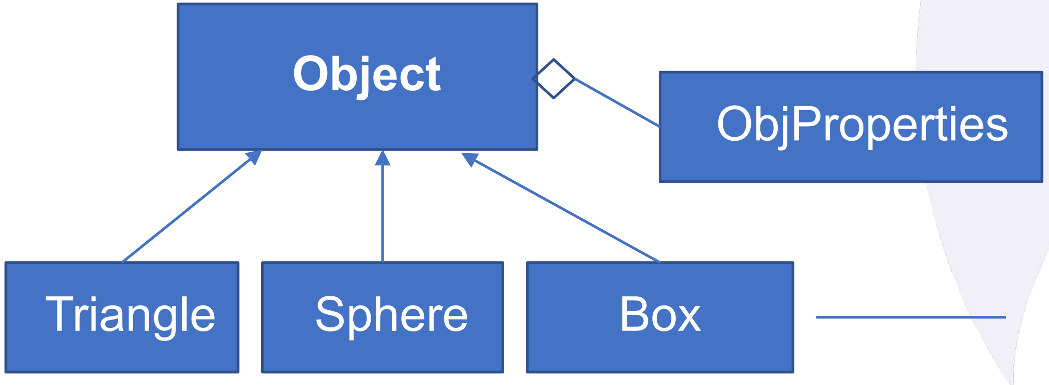


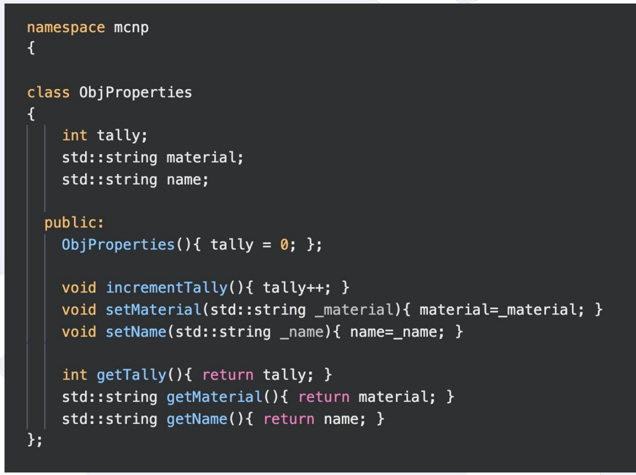
Event Log

1. Neutron scatter, photon production
2. Fission, photon production
3. Neutron capture
4. Neutron leakage
5. Photon scatter
6. Photon leakage
7. Photon capture

Approach$^{(ii)}$

2. Implement a test bed and propose use case to ANARI developers
   - Python test bed (fast to develop but too slow)
   - C++ testbed
     - Use language that scientists are familiar with
     - Isolate "non-graphics" inclusion for easy integration



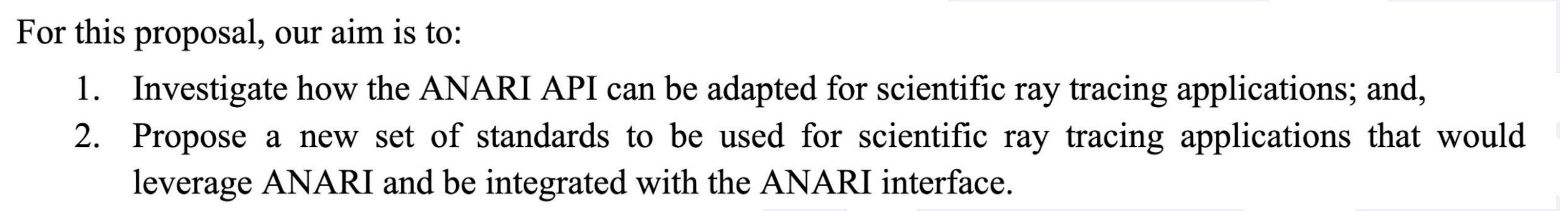


```
namespace mcnp
{

class ObjProperties
{
    int tally;
    std::string material;
    std::string name;

  public:
    ObjProperties(){ tally = 0; };

    void incrementTally(){ tally++; }
    void setMaterial(std::string _material){ material=_material; }
    void setName(std::string _name){ name=_name; }

    int getTally(){ return tally; }
    std::string getMaterial(){ return material; }
    std::string getName(){ return name; }
};
```

Approach[(ii)]

2. Implement a test bed and propose use case to ANARI developers
   - Python test bed (fast to develop but too slow)
   - C++ testbed
     - Use language that scientists are familiar with
     - Isolate "non-graphics" inclusion for easy integration
   - Work with LANL team developing MCNP for validation of the testbed
   - Ongoing work …
     - Geometry ✓
     - Rays ✓
     - Tallies . . . (add payload to rays, logging, …)
     - Validation

Outcomes


For this proposal, our aim is to:

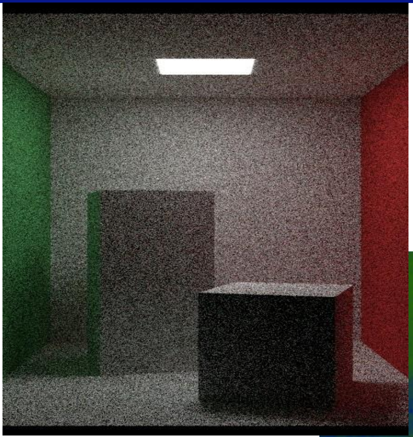
1. Investigate how the ANARI API can be adapted for scientific ray tracing applications; and,
2. Propose a new set of standards to be used for scientific ray tracing applications that would leverage ANARI and be integrated with the ANARI interface.

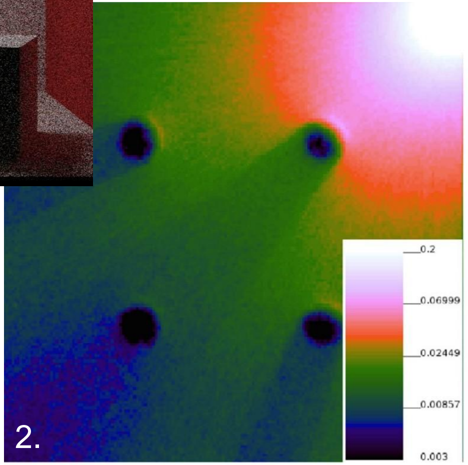
1. List of suggestions for the ANARI API to enable scientific computing
   - Approach: Sci Ray trace Fork

2. Development of a Proxy App for Vendors
   1. Connected to LANL MCNP team
   2. Connected to Intel Aurora team (App will also be released publicly – Nvidia, AMD, …
      - to stay vendor neutral

END

A Standard API for Particle Tracing





2.

1. Cornell Box
2. Jeremy E. Sweezy,A Monte Carlo volumetric-ray-casting estimator for global fluence tallies on GPUs, Journal of Computational Physics, Volume 372, 2018, Pages 426-445, ISSN 0021-9991

Project Description

Investigating how a standard API can be used to facilitate development of a RAY Tacing API for Scientific Applications

Project Outcomes

1. Identification of missing features in ANARI
   + proposing changes
2. Creation of proxy app for Vendors

PI: *Pascal Grosset*

Total Project Budget: *40K*

ISTI Focus Area: *Computer and Computational Science*