INTRO TO ACCELERATED RAY TRACING (RTX) WITH OPTIX 7

03.11.2021

RAMONA HOHL, COMPUTE DEVTECH ENG

RAY TRACING:
MODELING LIGHT BEHAVIOR TO GENERATE IMAGES

Images courtesy: Zhelong Xu, Adobe Substance, LeeGriggs, Autodesk Arnold, Moonlicht Studios, Chaos V-Ray, Madis Epler, Otoy Octane, Oly Stingel, Redshift, Siemens Digital Industries Software

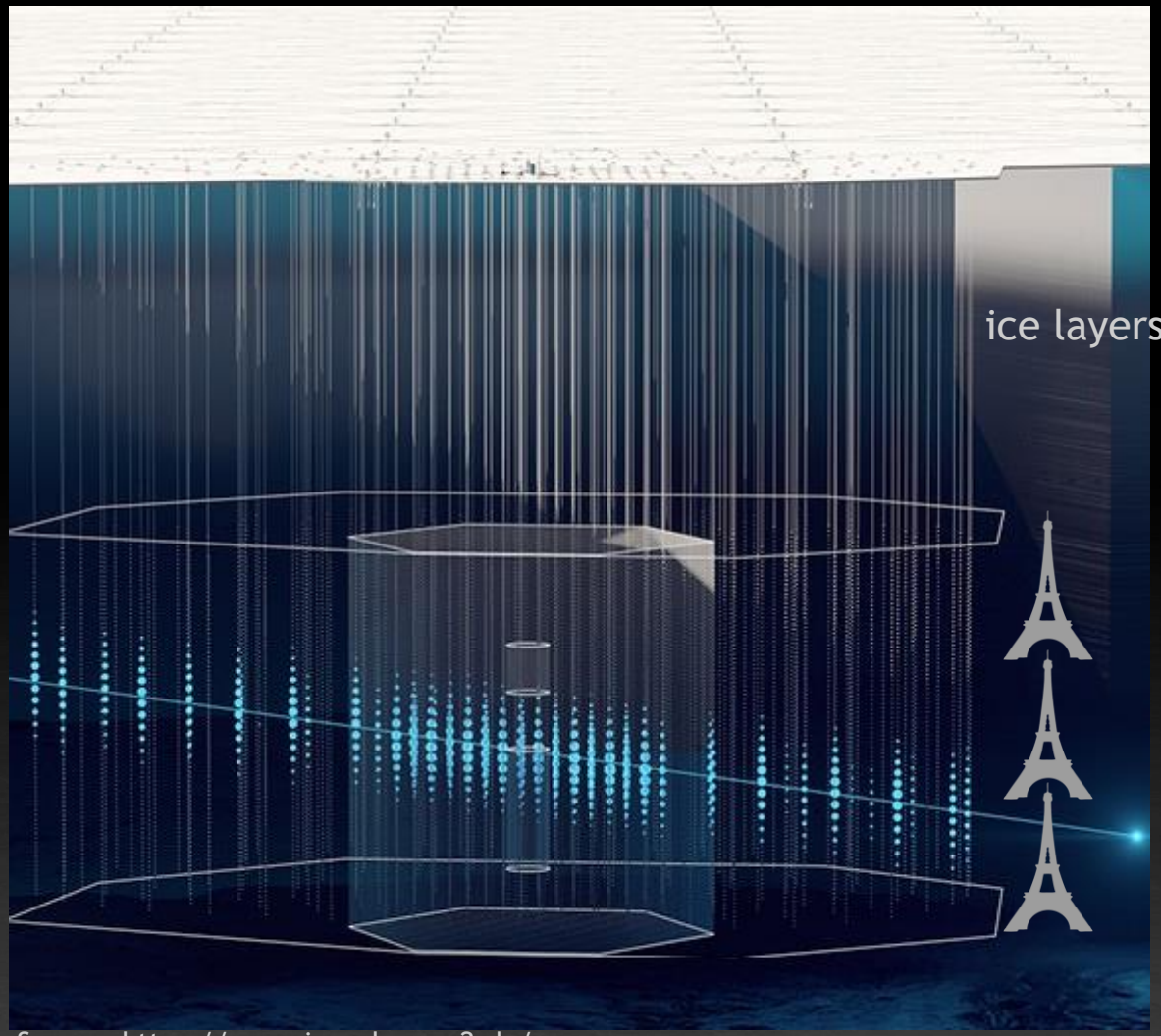
THE ICECUBE NEUTRINO OBSERVATORY

Image source: https://icecube.wisc.edu/gallery/detector/#modulagallery-7032-2074

ICECUBE SIMULATOR

Photon propagation through anisotropic layers of ice and collision detection

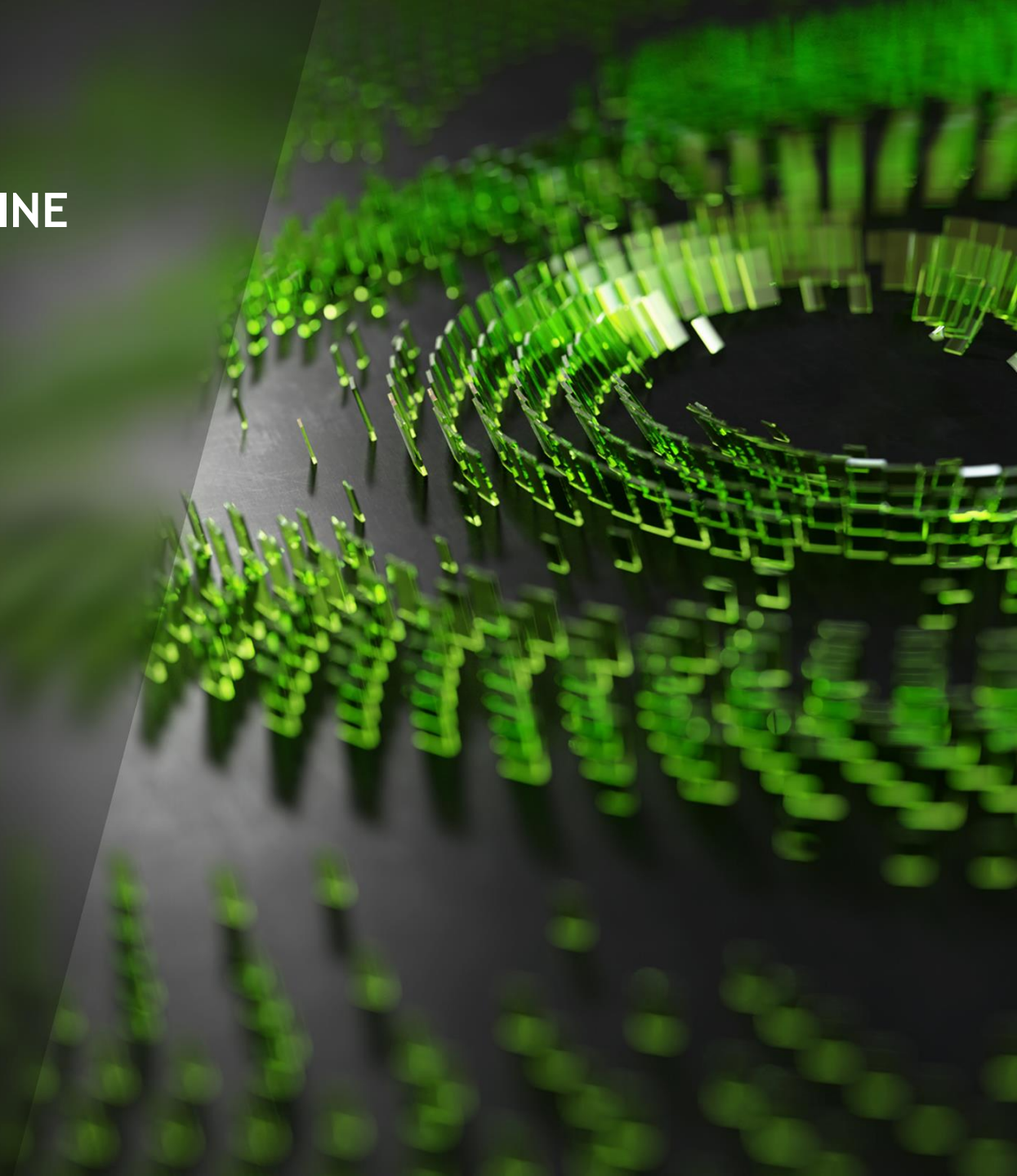
- CUDA model
  - 1D traversal of tilted ice model
  - Exploits simple detector structure
- IceCube-Gen2
  - More detector geometries
  - More sensor shapes
- Exceeds capabilities of current simulator
- What does OptiX enable?
  - Flexible in adding new geometries
  - Quicker R&D iteration



Source: https://www.icecube-gen2.de/

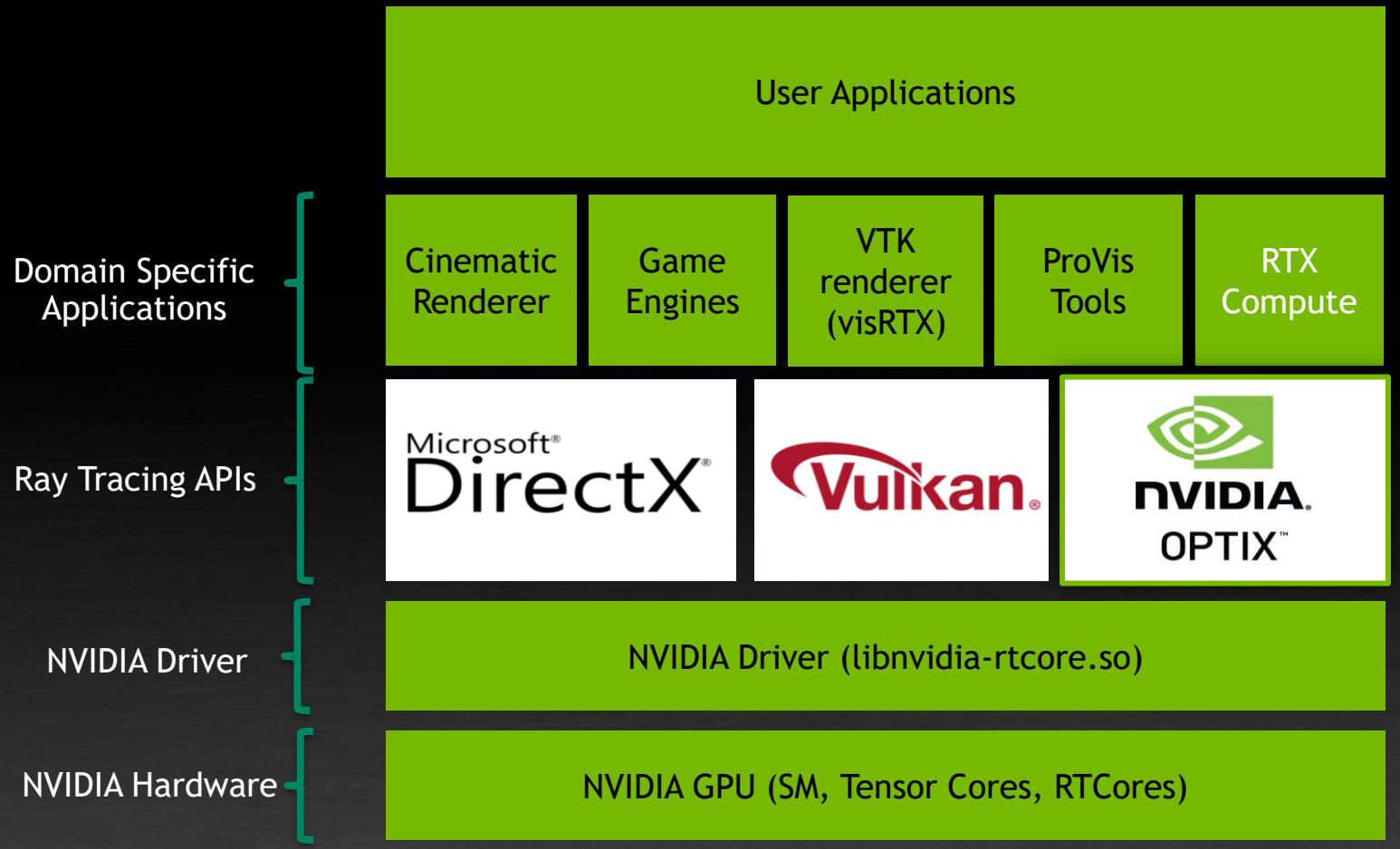
OUTLINE

- Motivation
- Ray tracing & OptiX
- RT Cores
- OptiX7
- Can an application benefit?
- IceCube with OptiX - a case study



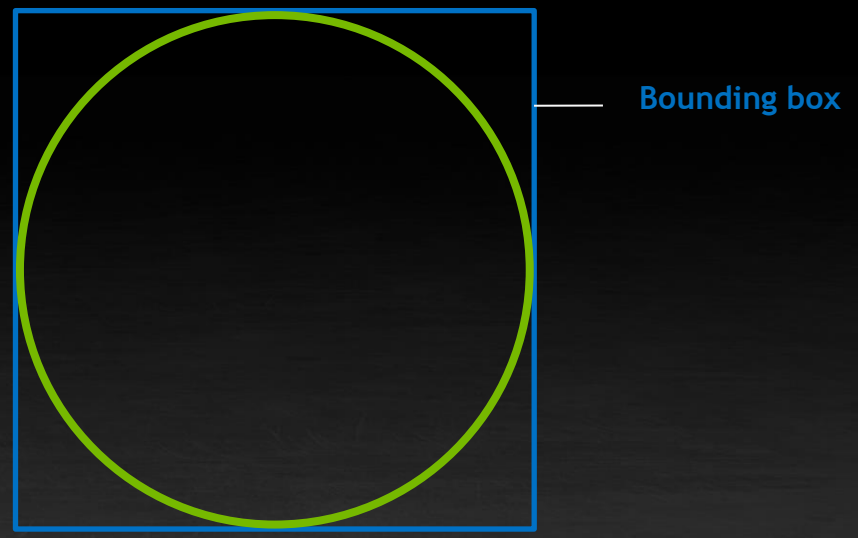
WHAT IS RTX ?

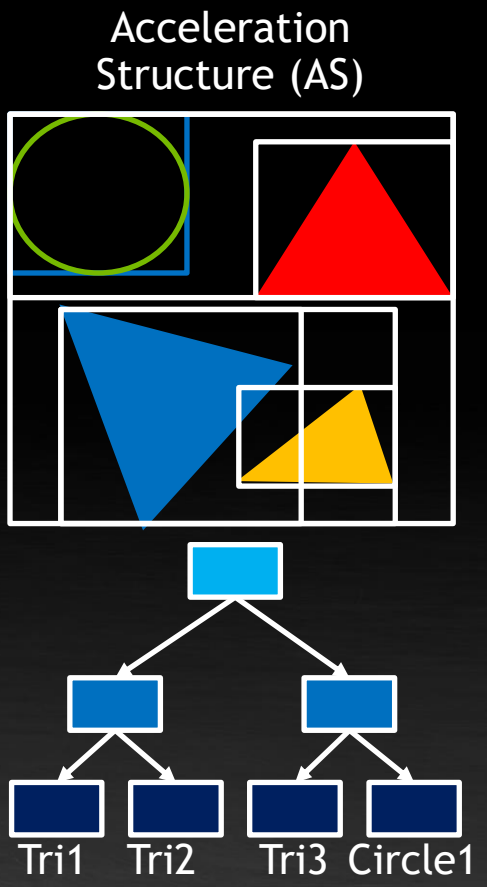
NVIDIA Ray-Tracing Stack



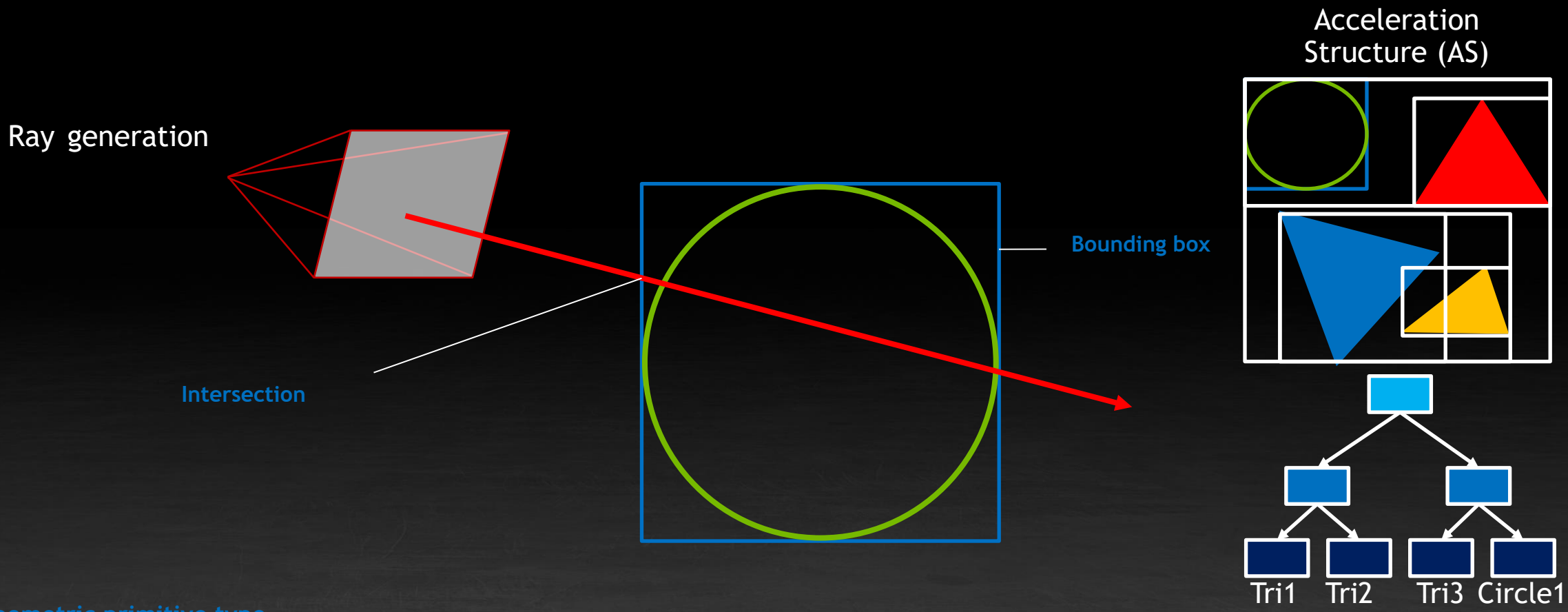
Core components in ray tracing

Core components in ray tracing



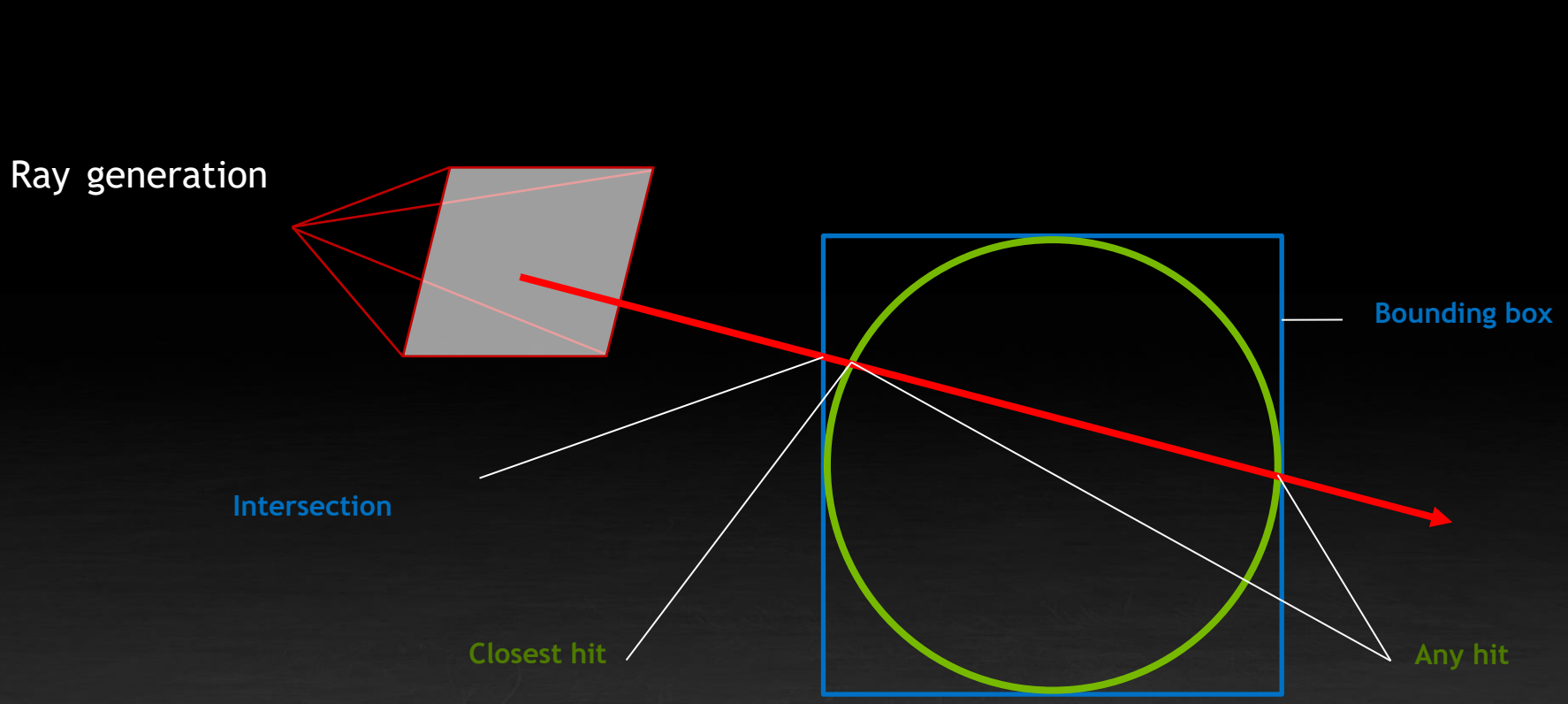


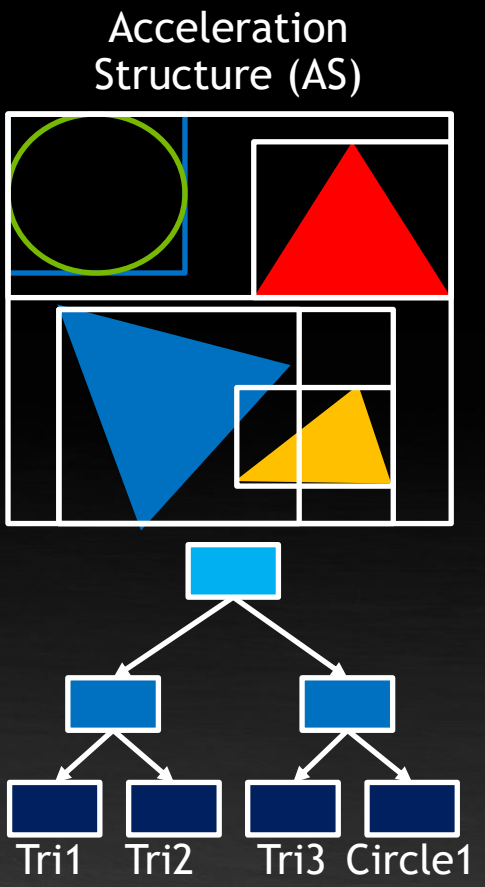
Core components in ray tracing



* per geometric primitive type
* per entry point
* per ray type

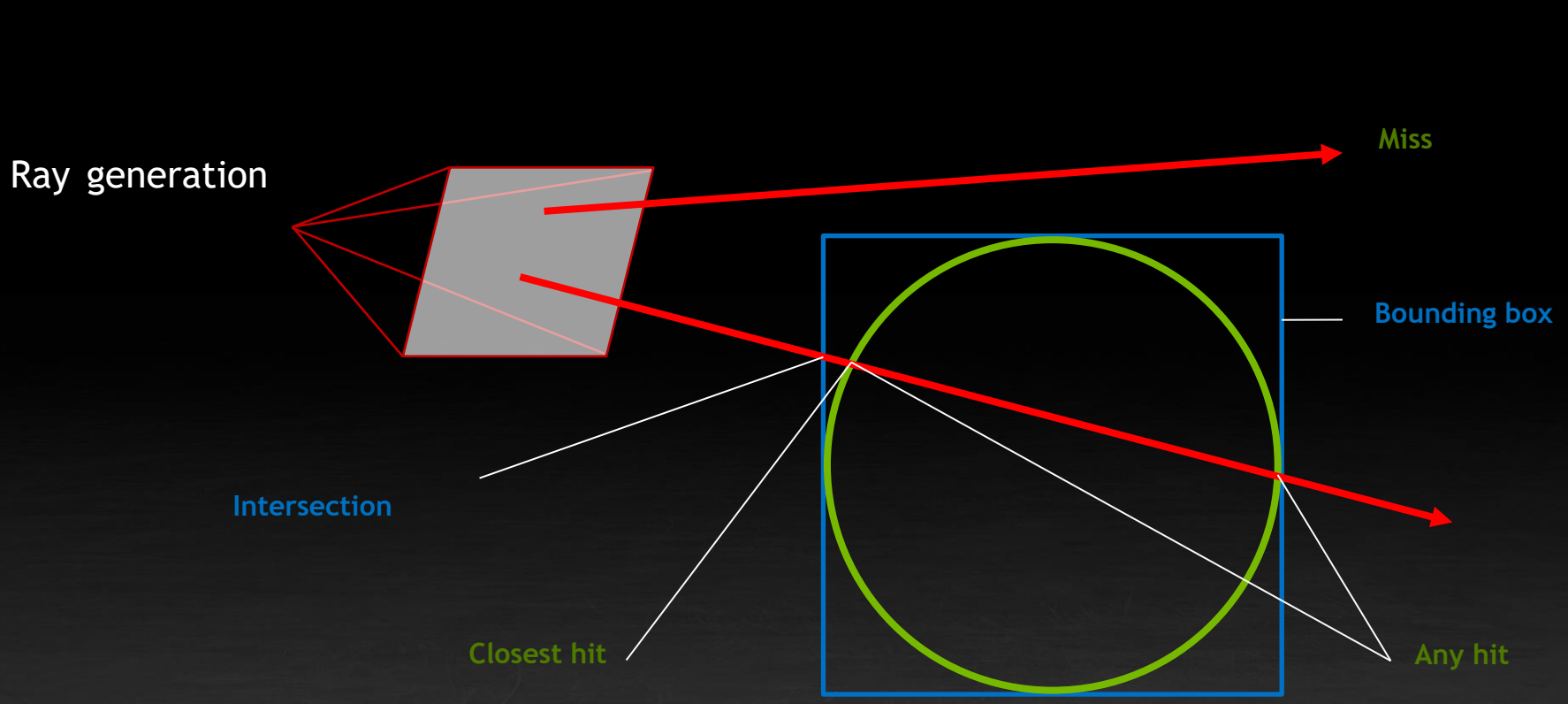
Core components in ray tracing

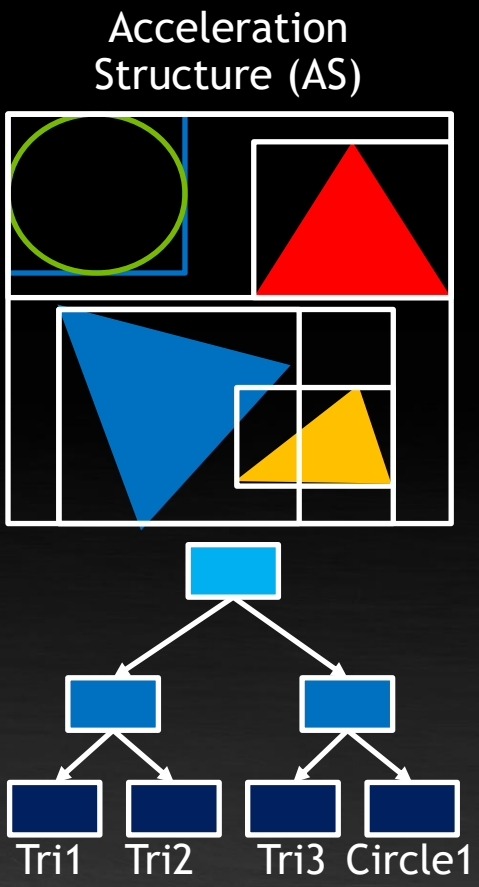




* per geometric primitive type
* per entry point
* per ray type

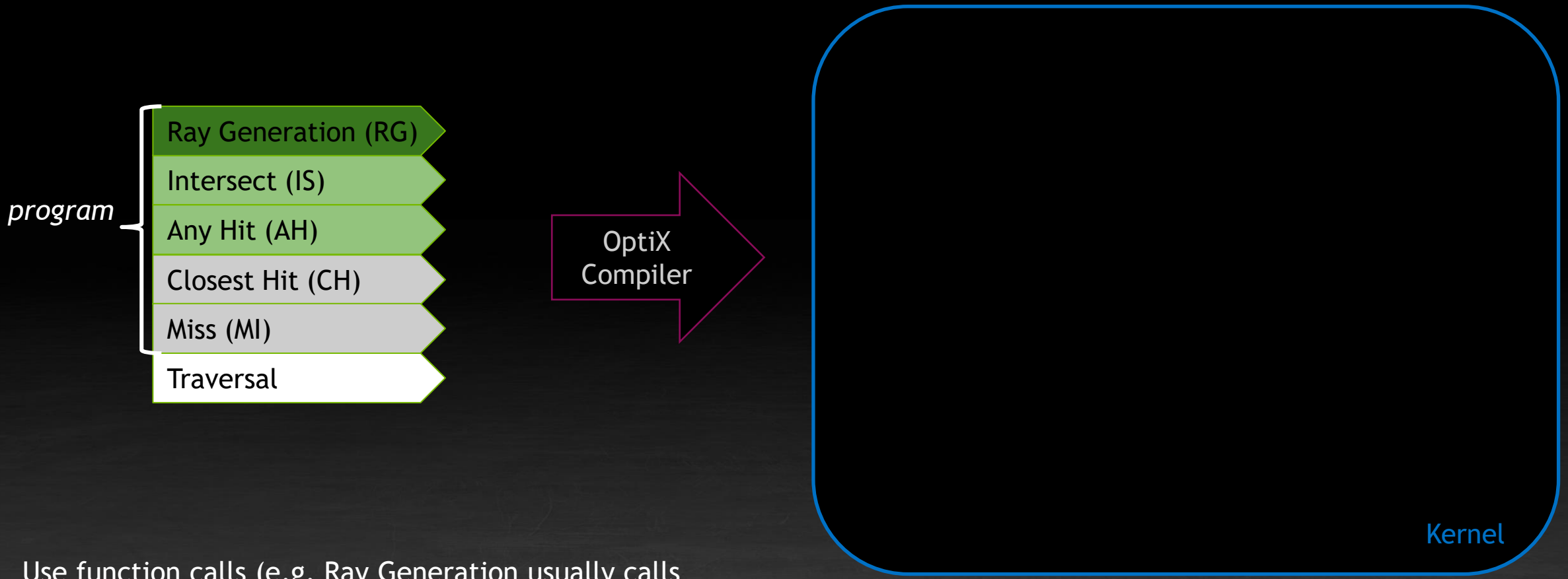
Core components in ray tracing





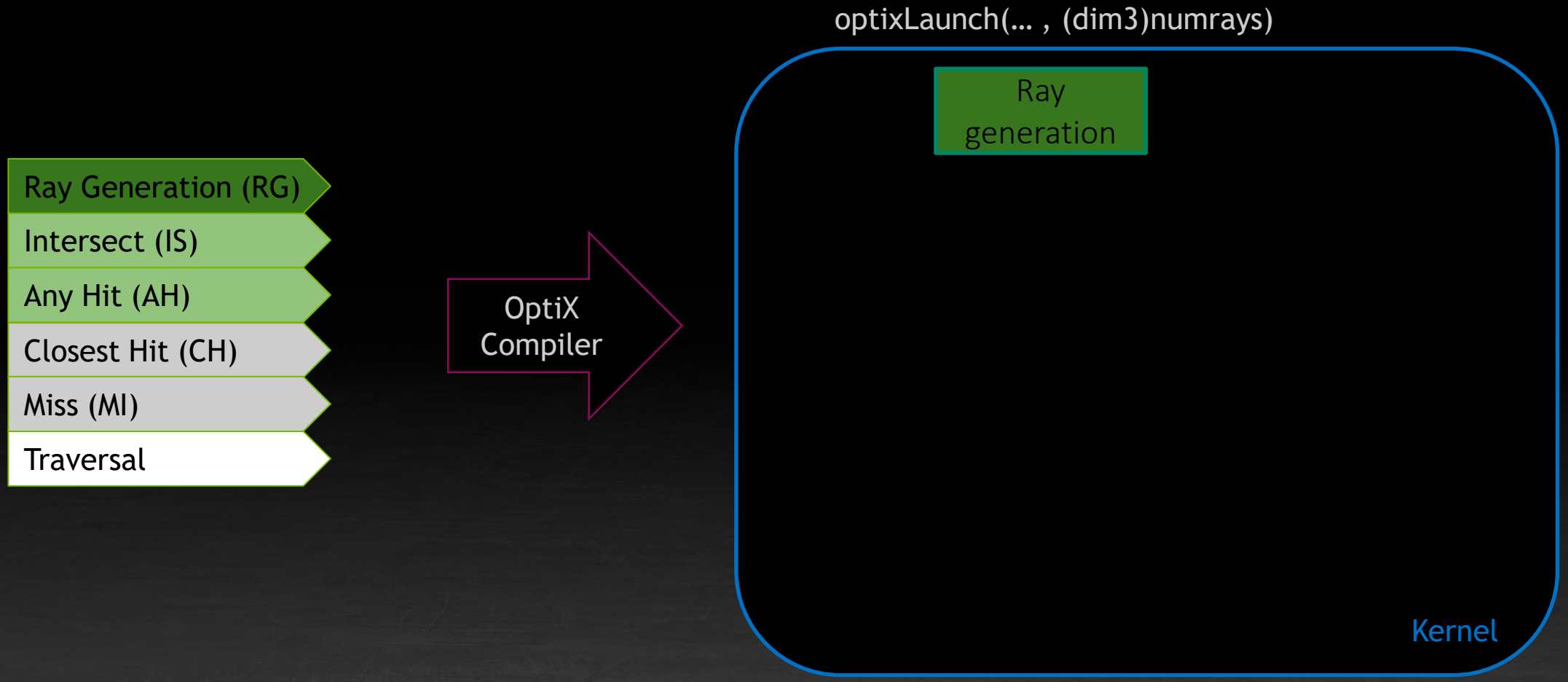
* per geometric primitive type
* per entry point
* per ray type

OPTIX GPU EXECUTION MODEL

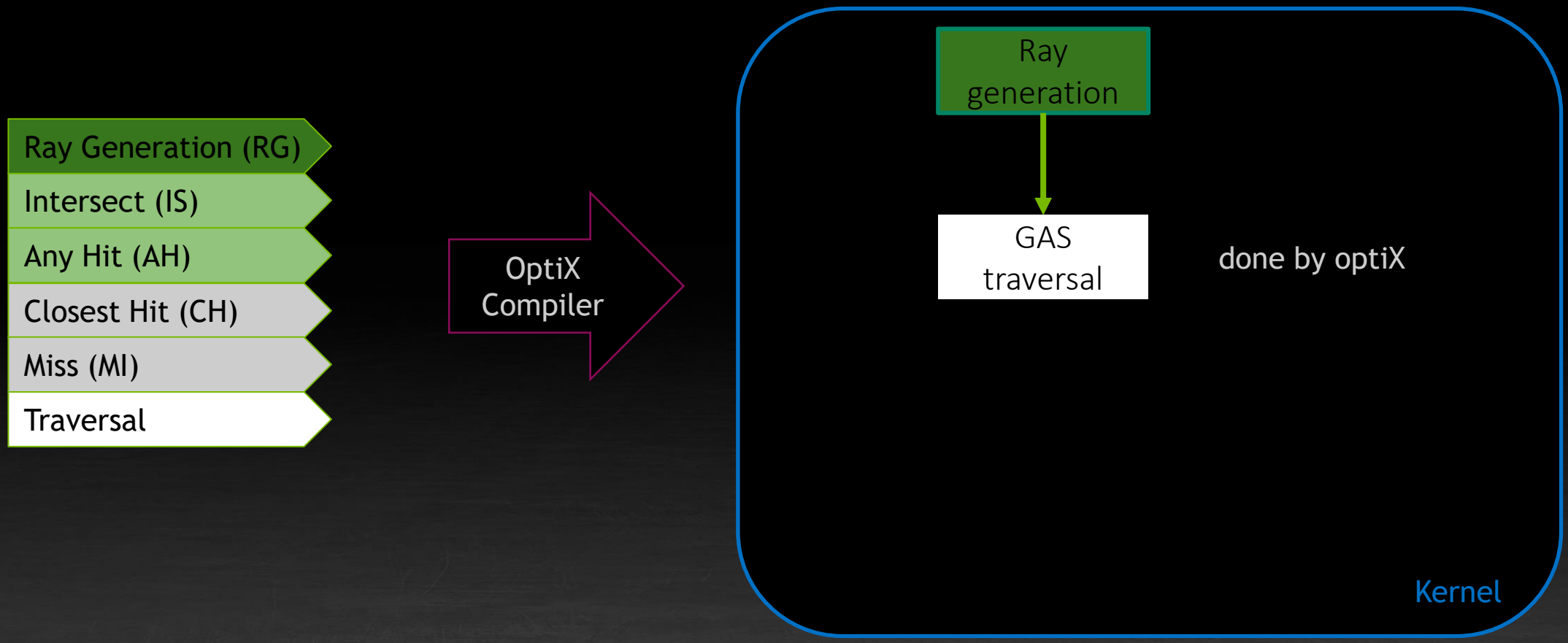


Use function calls (e.g. Ray Generation usually calls Traversal). Simple and clean 1:1 mapping of algorithmic view. Each thread executes one ray, rays may diverge

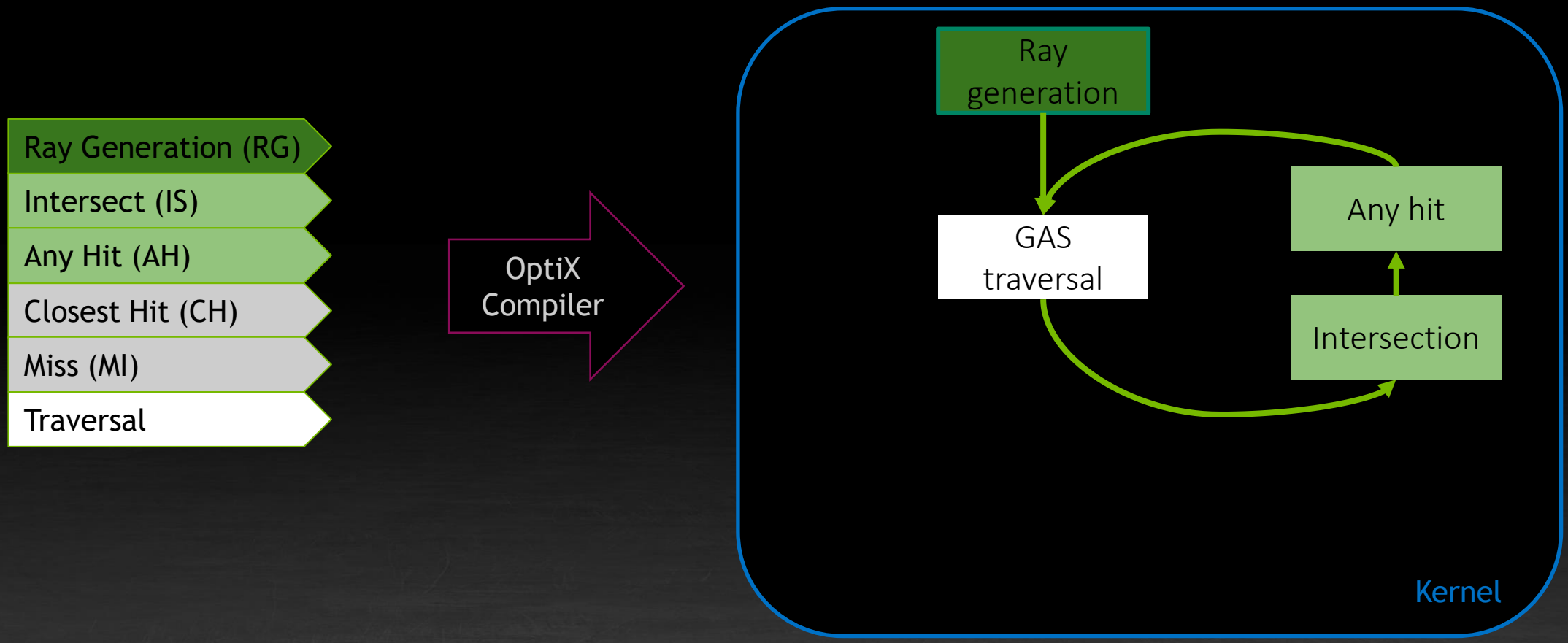
OPTIX GPU EXECUTION MODEL



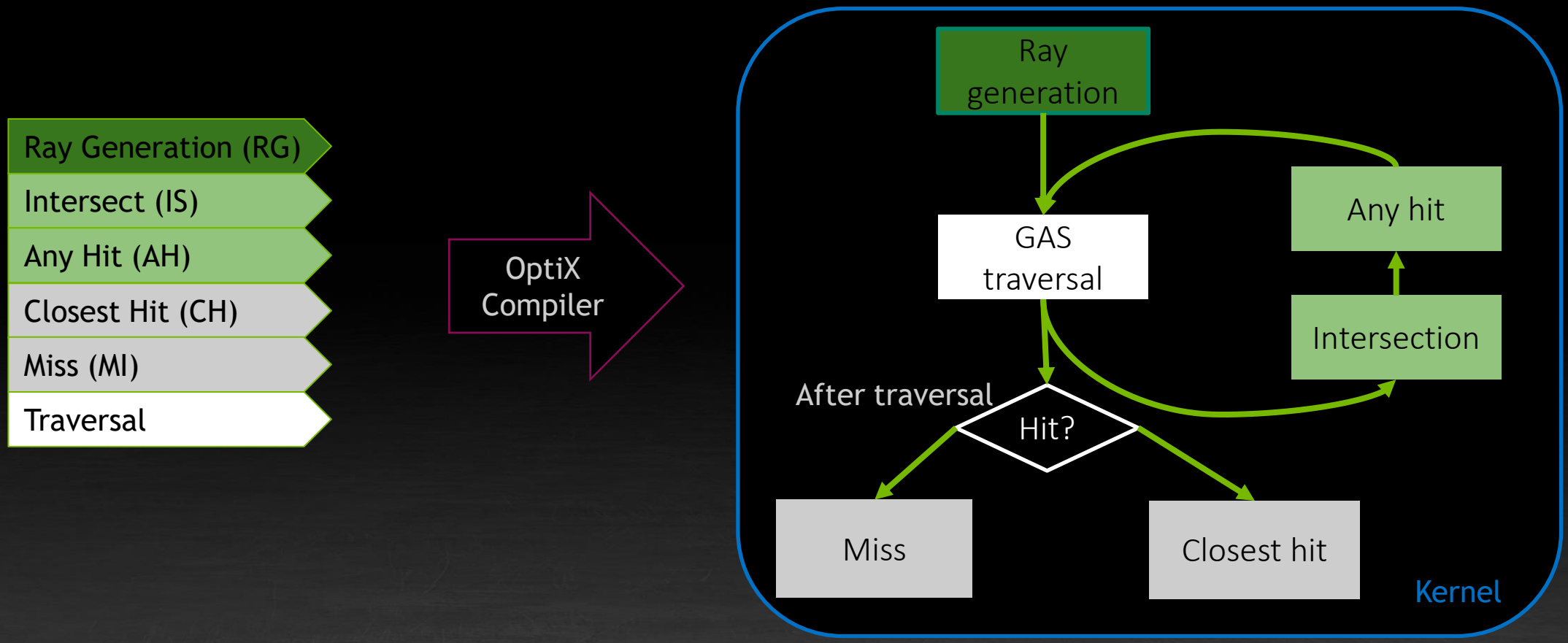
OPTIX GPU EXECUTION MODEL



OPTIX GPU EXECUTION MODEL

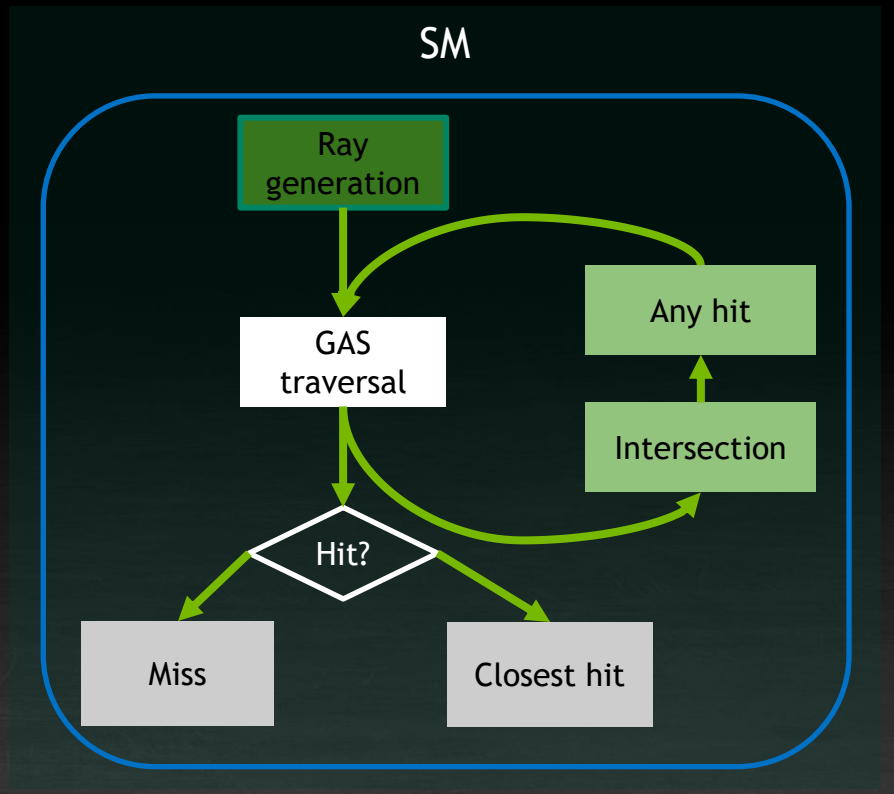


OPTIX GPU EXECUTION MODEL



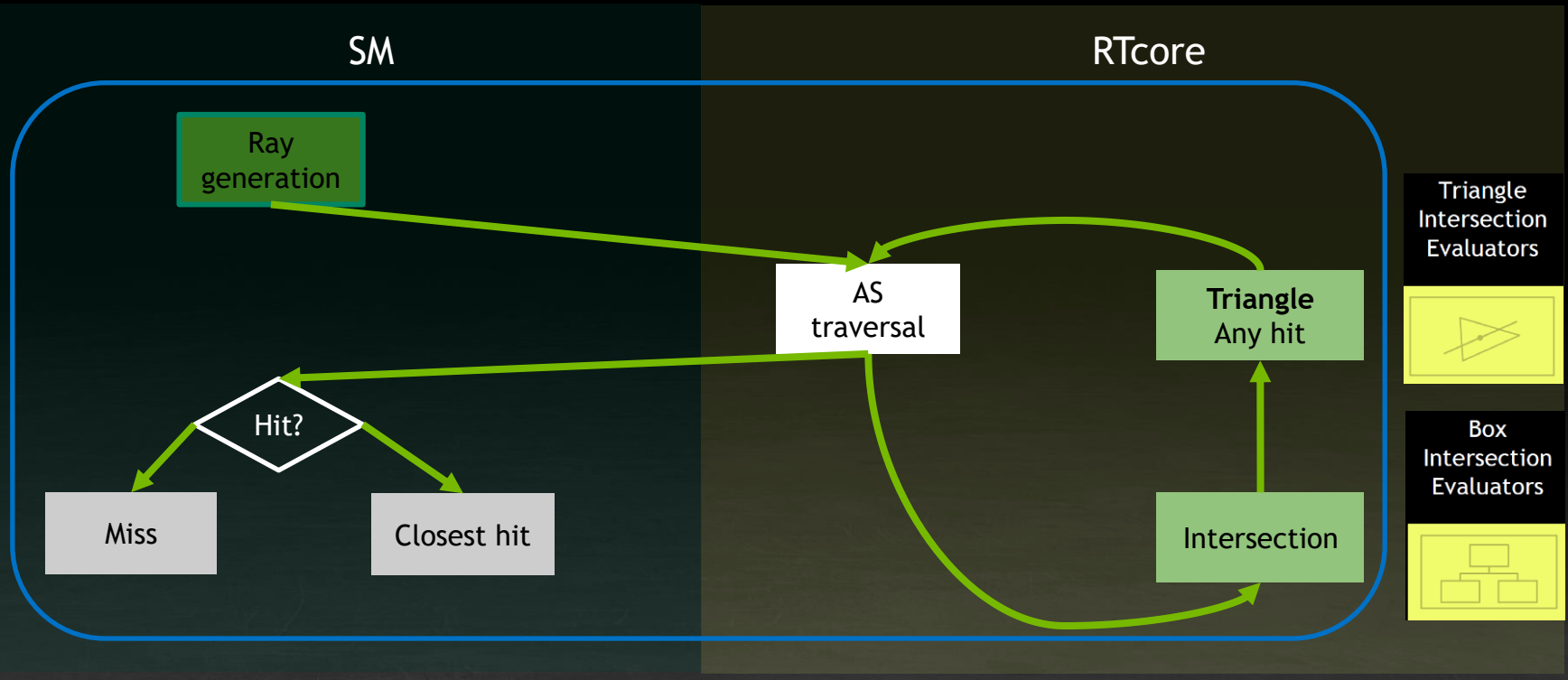
SM

RAY TRACING WITHOUT RTCORE



RAY TRACING WITH RTCORE

SM with RTcore



OPTIX 7

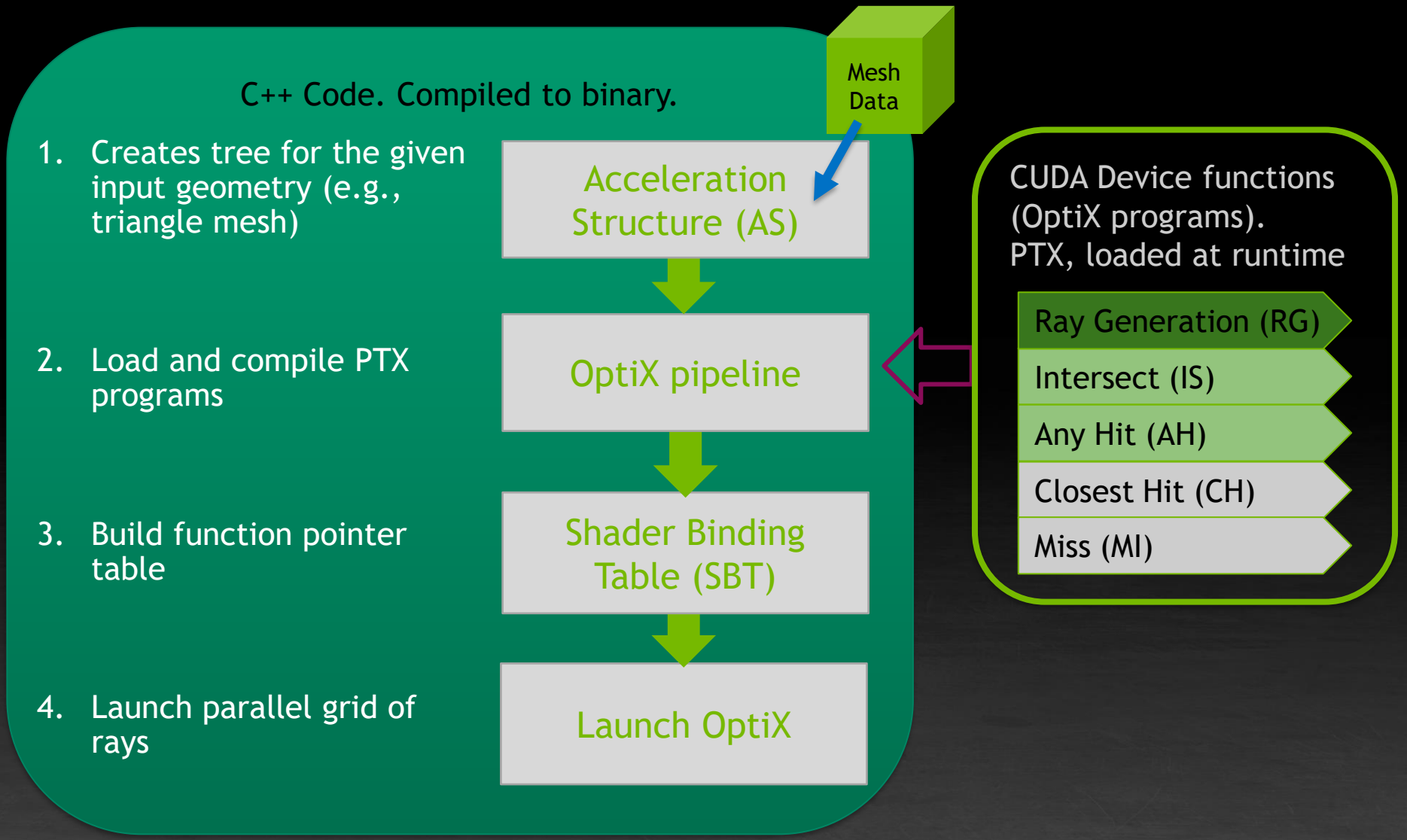
CUDA centric programming model

- Low-level CUDA centric rendering API
- Full explicit control
- Memory management (CUDA pointers, cudaMalloc)
- Acceleration structure build and management
- Fully asynchronous (CUDA streams)
- Fully thread-safe
- Profiling support with Nsight Compute

Start here:
https://developer.nvidia.com/siggraph/2019/video/sig911
https://raytracing-docs.nvidia.com/optix7/guide/index.html#introduction#
https://devblogs.nvidia.com/how-to-get-started-with-optix-7/
Gets you ready to start coding:
https://developer.nvidia.com/gtc/2020/video/s21888
https://gitlab.com/ingowald/optix7course

OPTIX 7 APPLICATION LAYOUT

Flow of an OptiX 7 application

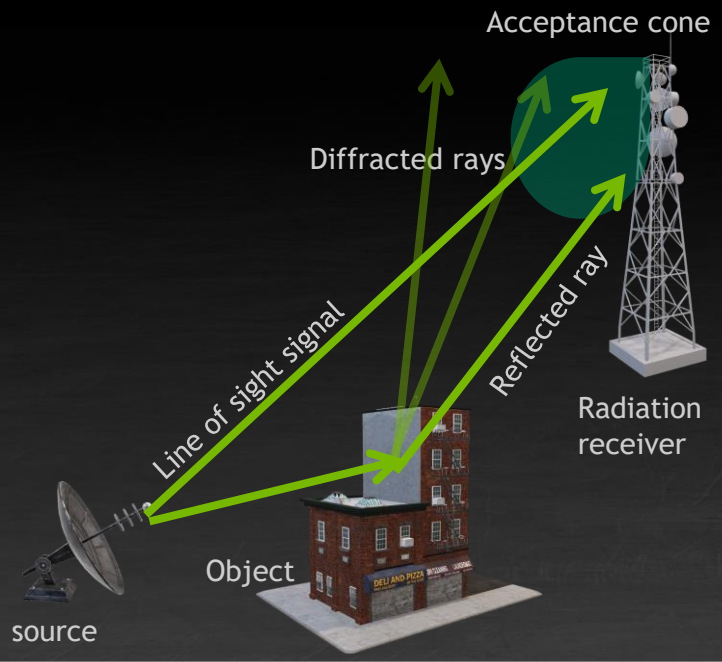


COMPUTATIONAL PATTERNS

Which application can benefit from RTX?

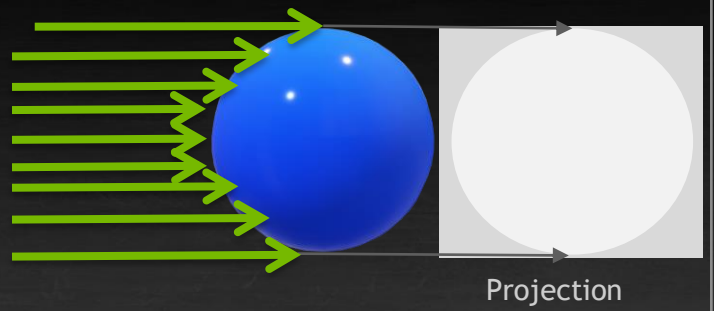
DIRECT MAPPING TO RAY TRACING

✓ Rays
✓ Geometry



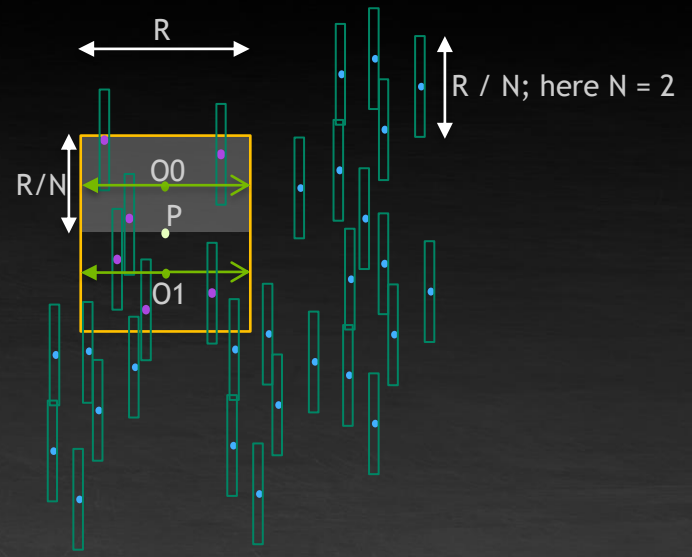
INDIRECT MAPPING TO RAY TRACING

× No explicit rays
✓ Geometry

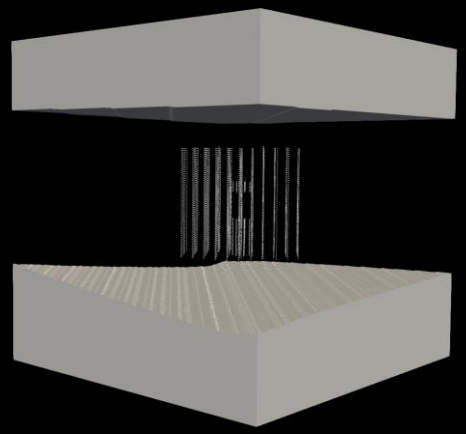


ALGORITHMIC RETHINKING

× No explicit rays
× No explicit geometry



https://github.com/NVIDIA/rtx_compute_samples - examples description at: RTCore for Compute GTC 2020



✓ Rays
✓ Geometry

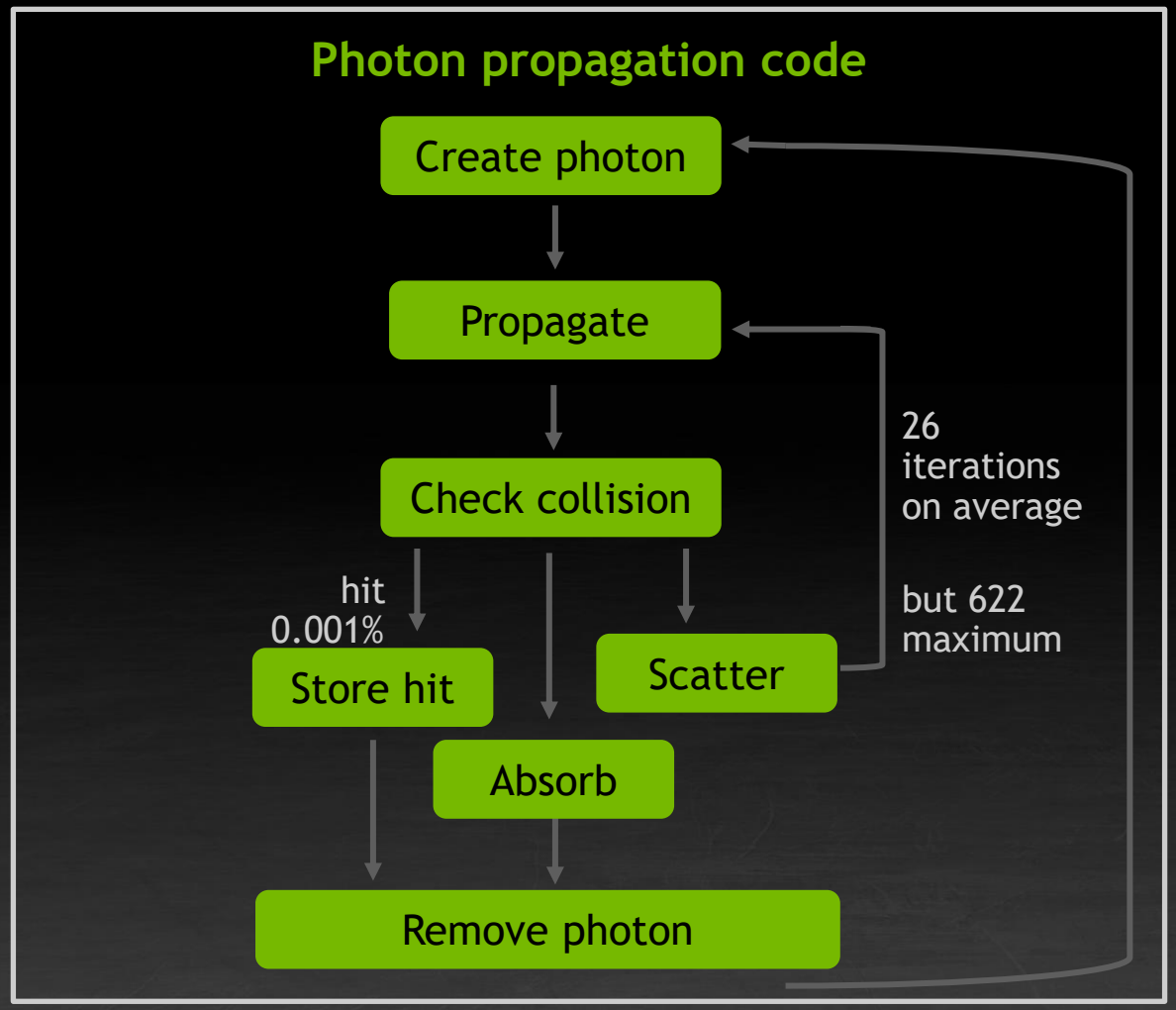
ICECUBE PHOTON PROPAGATION; COLLISION DETECTION WITH OPTIX

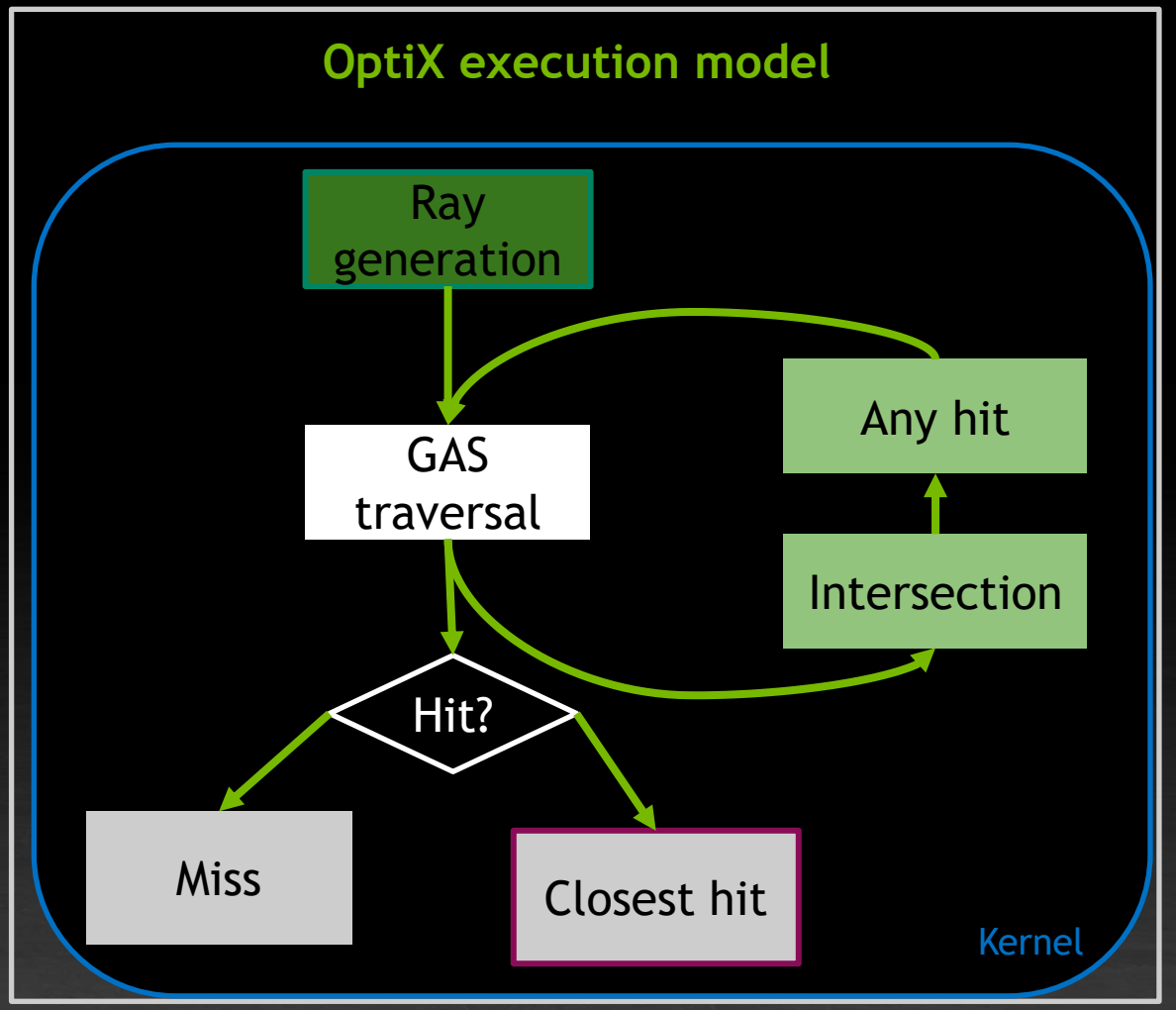
A case study

Simplified model of D-Egg of IceCube-Gen2
(made with NVIDIA Omniverse & Omniverse Paraview Connector)

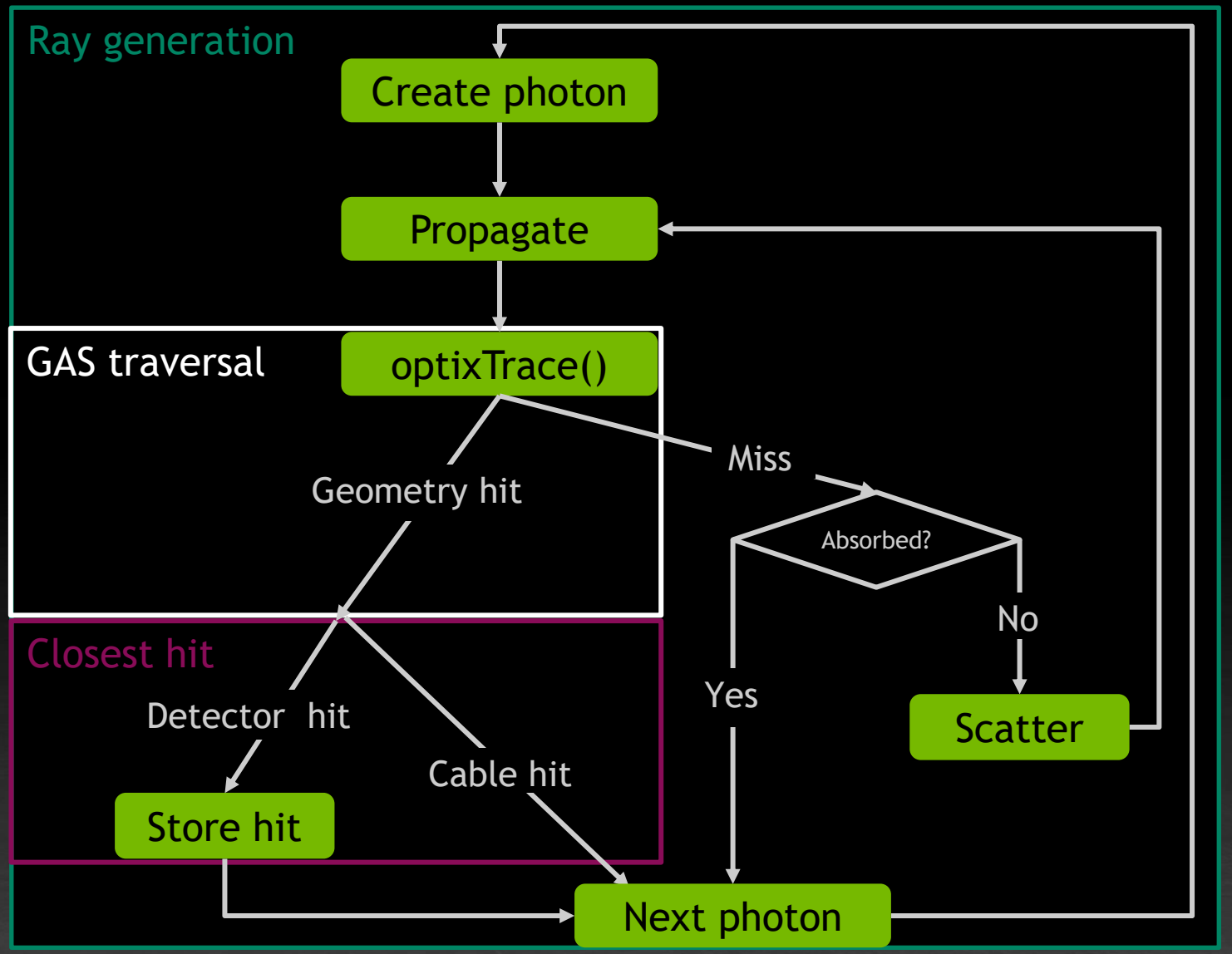
PHOTON PROPOGATION

IceCube photon propogation code vs OptiX



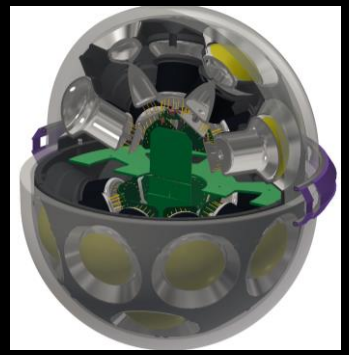


COLLISION DETECTION WITH OPTIX



ADDING GEOMETRIC COMPLEXITY COMES "FOR FREE"

Pill-shape D-Egg of IceCube-Gen2
(source: IceCube collaboration)



Spherical DOM of IceCube
(source: IceCube collaboration)

Model	#Triangles	Detector hits	Cable hits	Total hits	[ns]/ray
Sphere detectors	9’907’200	42’099	0	42’099	3.90
Sphere detectors, cables	9’926’640	40’978	132’209	173’187	3.94
Pill detectors	12’291’120	61’437	0	61’437	3.89
Pill detectors, cables	12’310’560	60’250	128’797	189’047	3.94

5160 detectors

86 cables

146,152,721 photons

1e4 eV neutrino’s energy

~570ms runtime

(*older code version)

HOW TO OPTIMIZE IT FURTHER?

OptiX concepts applied to the IceCube photon propagation

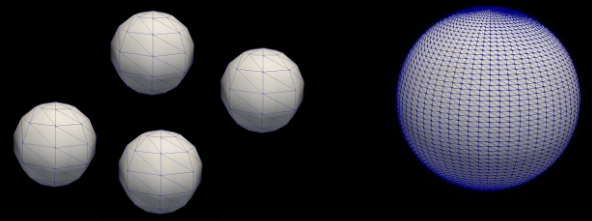
MATERIALS

Treat cable and detector hits differently

- Create separate build inputs
- Have two closest hit programs
- Watertight and single hit rule

INSTANCING

1 DOM mesh + 5160 transformation matrices



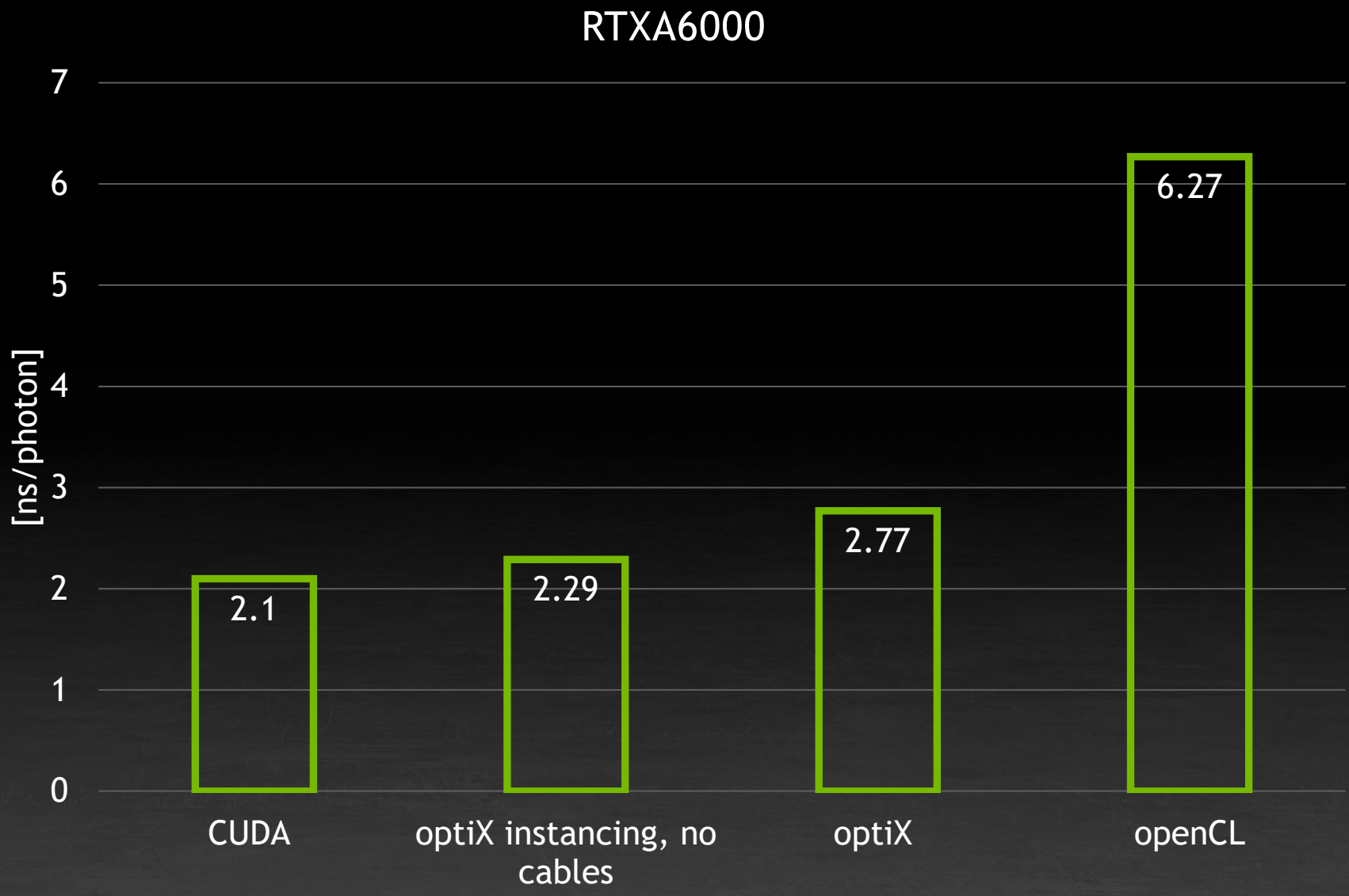
- Save memory
- -> Higher resolution
- Speed up

CUSTOM INTERSECTION

Ray sphere intersection for spherical DOMs

+ Accuracy, no tessellation
+ Less data
- No RT core triangle intersection
- Not easily applicable for all geometries
  (-> constructive solid geometry as in Opticks)

PERFORMANCE COMPARISON
for the sphere detector



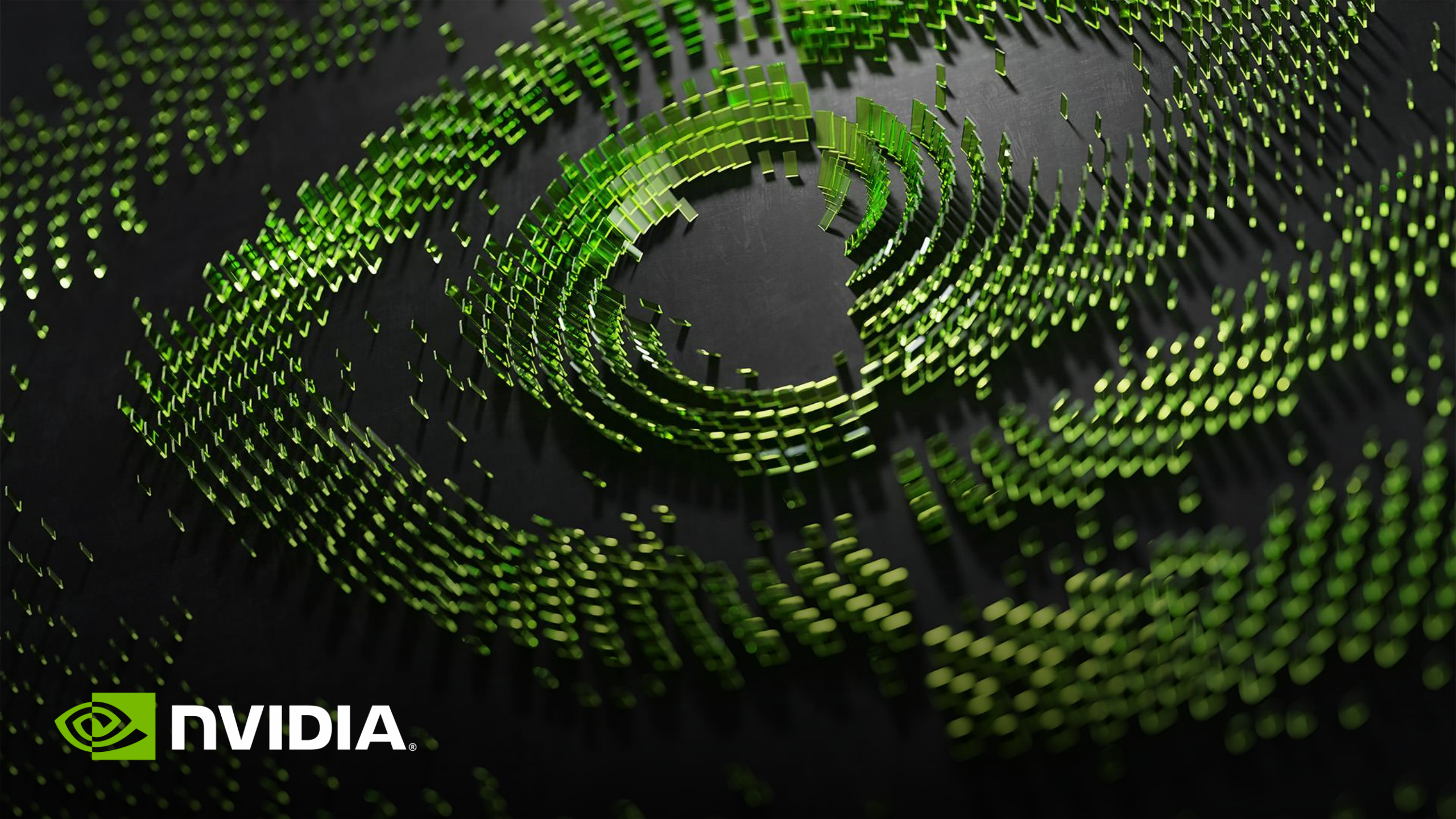
SUMMARY

- Ray tracing (RTX) is not only for graphics but helps compute problems as well
- OptiX7 is CUDA centric: CUDA with rays
- Geometry heavy cases tend to benefit the most from RTX
- IceCube photon propagation: OptiX offers flexibility with detector geometries

Acknowledgements: Vishal Mehta, Max Rietmann, Peter Messmer for their support

THANK YOU!

IN CASE OF MORE QUESTIONS; RHOHL@NVIDIA.COM


NVIDIA