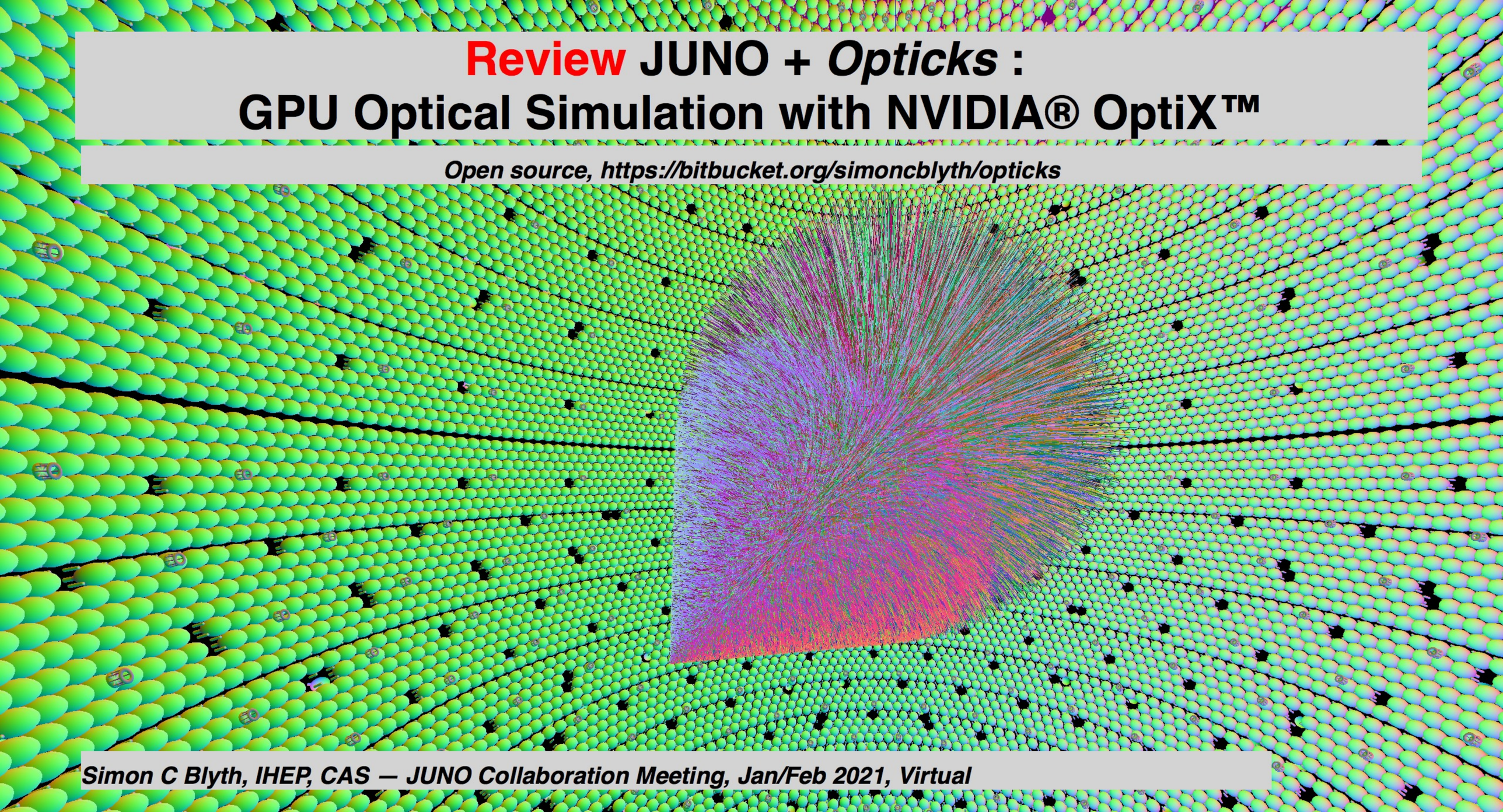
Review JUNO + *Opticks* : GPU Optical Simulation with NVIDIA® OptiX™

Open source, https://bitbucket.org/simoncblyth/opticks

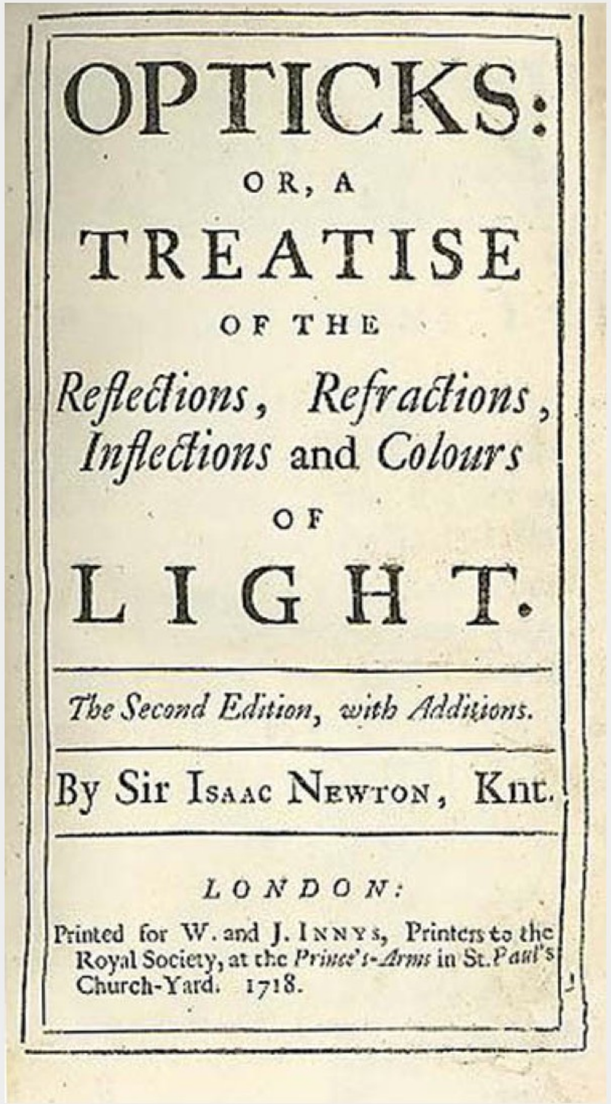


Simon C Blyth, IHEP, CAS — JUNO Collaboration Meeting, Jan/Feb 2021, Virtual

Talk

This talk will review progress with Opticks and with its use in JUNO.

Outline



OPTICKS:

OR, A

TREATISE

OF THE

Reflections, Refractions, Inflections and *Colours*

OF

LIGHT.

The Second Edition, with Additions.

By Sir Isaac Newton, Knt.

LONDON:

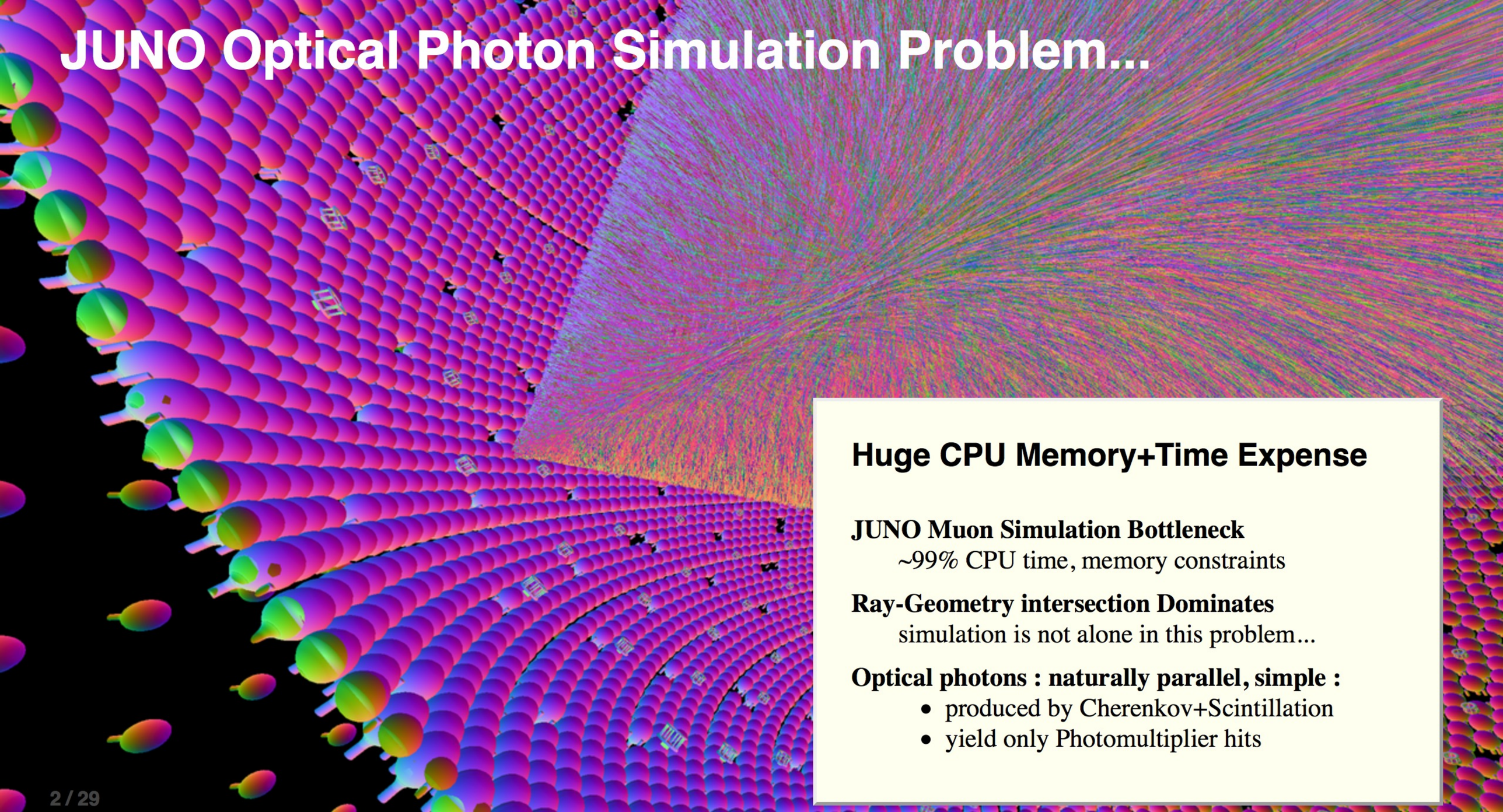
Printed for W. and J. Innys, Printers to the Royal Society, at the *Prince's-Arms* in St. *Paul's* Church-Yard. 1718.

Outline Talk

After a brief introduction I will cover:

- integration of Opticks with Geant4,
- its increasingly widespread usage
- progress on integration of Opticks with JUNO Offline
- and the next steps.

JUNO Optical Photon Simulation Problem...



Huge CPU Memory+Time Expense

JUNO Muon Simulation Bottleneck
~99% CPU time, memory constraints

Ray-Geometry intersection Dominates
simulation is not alone in this problem...

Optical photons : naturally parallel, simple :

- produced by Cherenkov+Scintillation
- yield only Photomultiplier hits

JUNO Optical Photon Simulation Problem... Talk

- I am sure your are all painfully familiar with the optical photon simulation problem, not just in processing time but also the memory for all those millions of photons
- Most of the time is taken finding intersections between photons and geometry.
- This ray tracing bottleneck is exactly the same as faced by ray trace image rendering in computer graphics.

NVIDIA® OptiX™ Ray Tracing Engine -- http://developer.nvidia.com/optix

OptiX makes GPU ray tracing accessible

- **accelerates** ray-geometry intersections
- simple : single-ray programming model
- "...free to use within any application..."
- access RT Cores[1] with OptiX 6.0.0+ via RTX™ mode

NVIDIA expertise:

- ~linear scaling up to 4 GPUs
- acceleration structure creation + traversal (Blue)
- instanced sharing of geometry + acceleration structures
- compiler optimized for GPU ray tracing

`https://developer.nvidia.com/rtx`

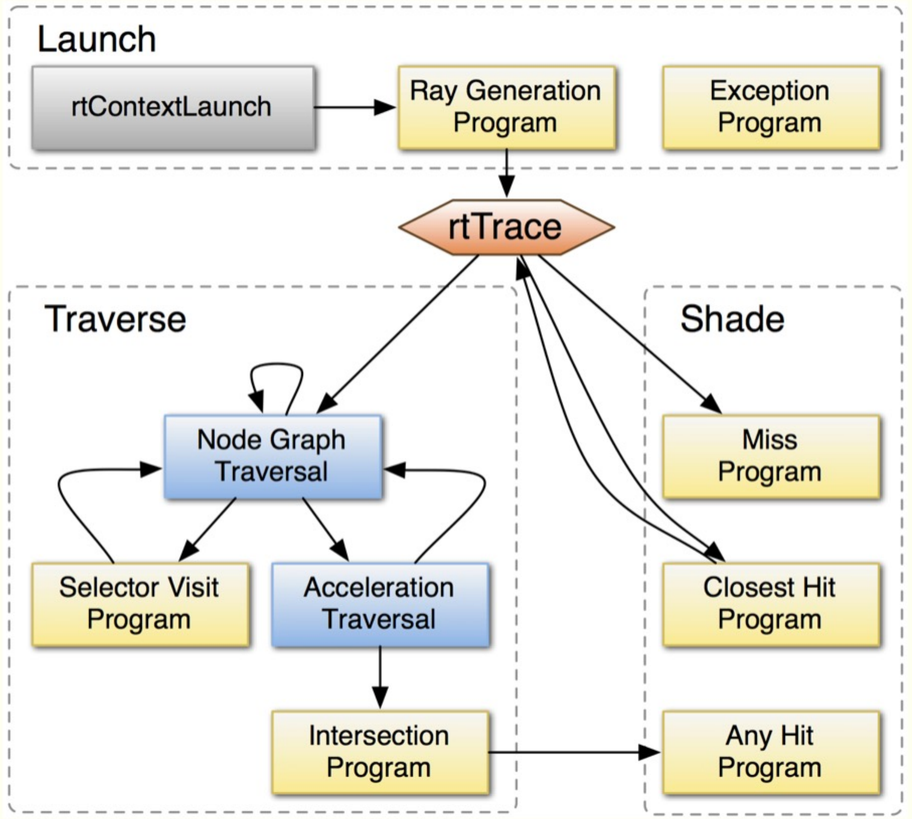
User provides (Yellow):

- ray generation
- geometry bounding box, intersects

[1] Turing+ GPUs eg NVIDIA TITAN RTX

OptiX Raytracing Pipeline

Analogous to OpenGL rasterization pipeline:



Talk

- NVIDIA OptiX provides GPUs accelerated ray-geometry intersection
- but to benefit from it the geometry must be translated into GPU appropriate forms totally different to the Geant4 geometry model

NVIDIA OptiX 7 : Entirely new thin API

NVIDIA OptiX 7 Features

- Minimal host state
- GPU memory *managed by application*
- GPU launches : explicit, asynchronous (CUDA streams)
- All host functions are thread-safe
- Multi-GPU *managed by application*

"managed by application" -> difficult development

- lots of learning + expert assistance needed

Disadvantage

Demands much more developer effort than OptiX 6

Advantage

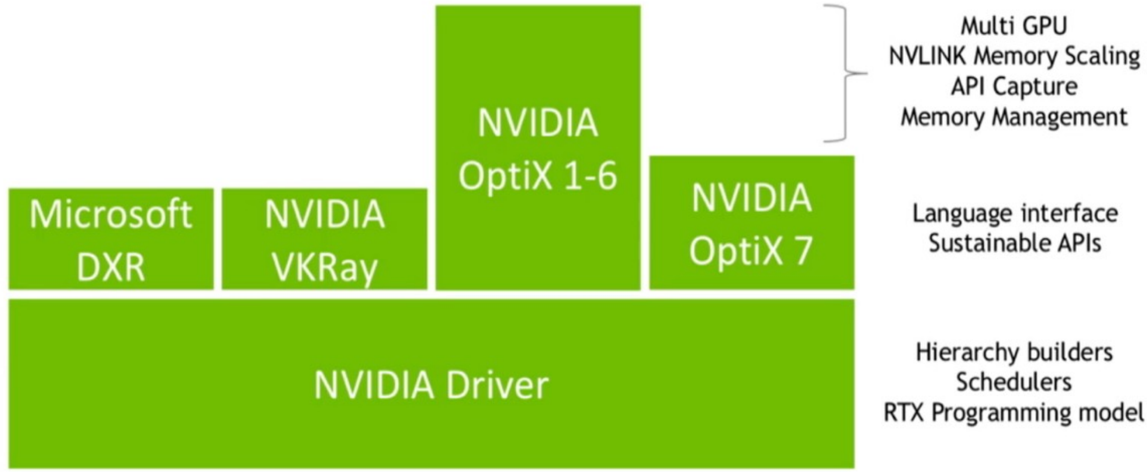
More control/flexibility over everything.

- thread-safe : important for distributed Server/Client Opticks

NVIDIA OptiX 7 : Entirely new API

- introduced August 2019
- low-level CUDA-centric thin API (Vulkan-ized)
- ~~near perfect scaling to 4 GPUs, for free~~
- Major effort needed to support in Opticks

Introducing OptiX 7

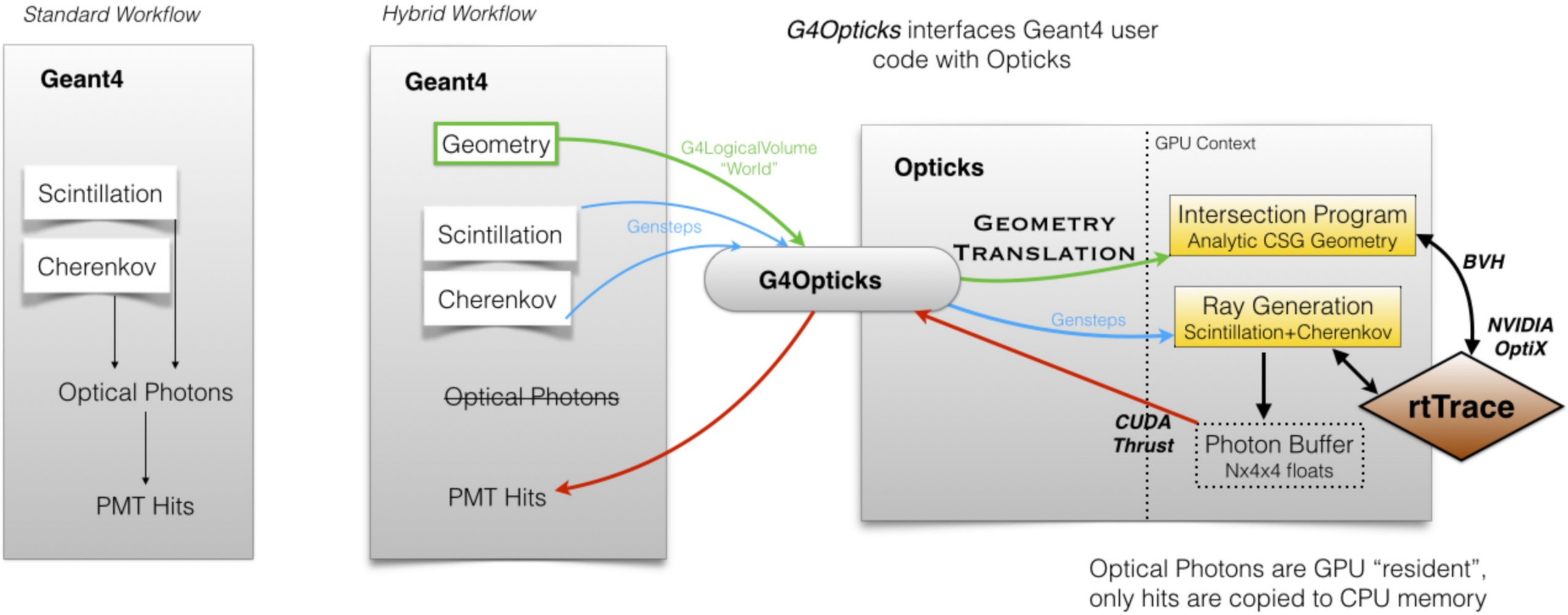


NVIDIA OptiX 7 : Entirely new thin API Talk

- Opticks current works with NVIDIA OptiX 6
- OptiX 7 is an entirely new (and much slimmer) API
- migration is unavoidable

Geant4 + Opticks Hybrid Workflow : External Optical Photon Simulation

https://bitbucket.org/simoncblyth/opticks



Talk

SMALL This compares the standard workflow on the left to the hybrid approach on the right

In the hybrid workflow, the generation of photons is moved from the CPU to the GPU using "Gensteps" as a bridge. The Genstep parameters include a number of photons, and the line segment along which to generate them and anything else needed in the generation loop.

Of course a port of the generation loop to CUDA is needed too.

The hybrid approach means:

- the optical photon simulation is entirely offloaded to the GPU
- only PMT hits need CPU memory
- actually I have recently taken this further, reducing the CPU memory by a factor of the collection efficiency by making the collection efficiency decision on the GPU
- the green arrows represent the geometry translation
    - most of the work and problems of Opticks are with the geometry translation

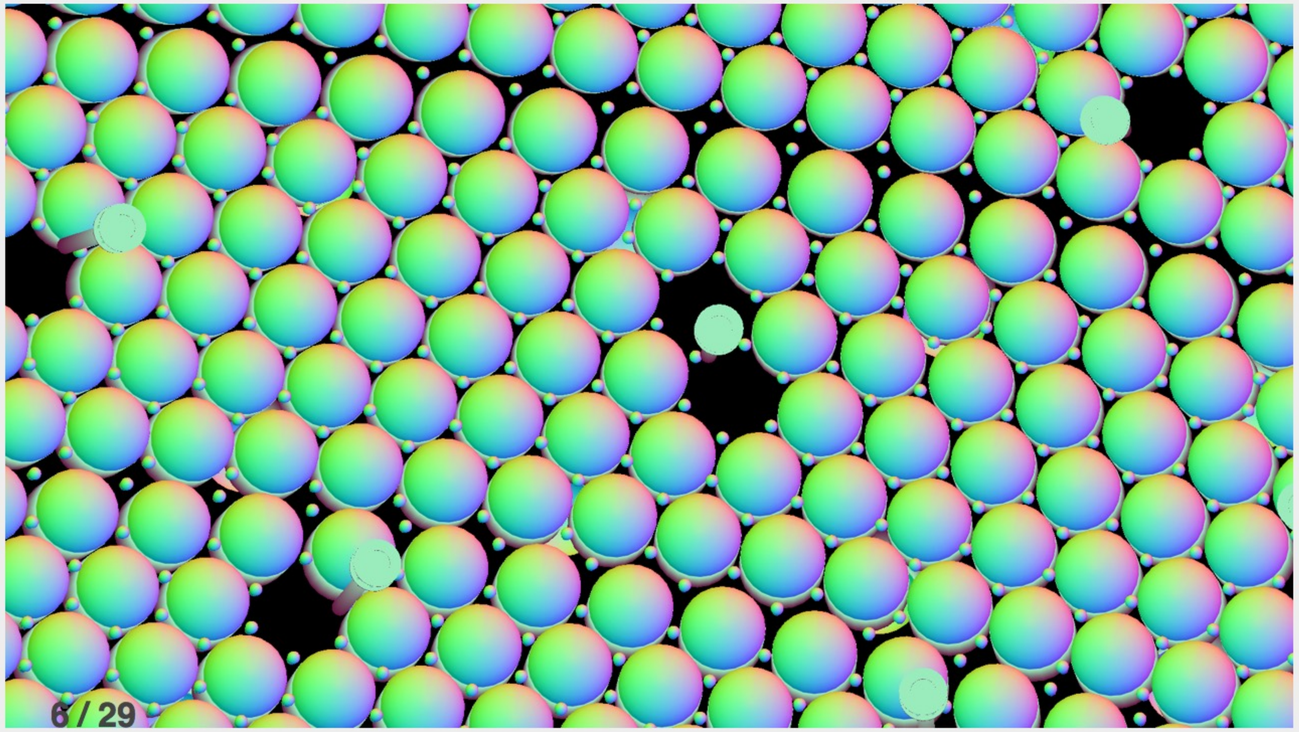
Opticks : Translates G4 Geometry to GPU, Without Approximation

G4 Structure Tree -> Instance+Global Arrays -> OptiX

Group structure into repeated instances + global remainder:

- auto-identify repeated geometry with "progeny digests"
  - JUNO (SVN trunk) : 9 distinct instances + 1 global
- instance transforms used in OptiX/OpenGL geometry

instancing -> huge memory savings for JUNO PMTs



Materials/Surfaces -> GPU Texture

Material/Surface/Scintillator properties

- interpolated to standard wavelength domain
- interleaved into "boundary" texture
- "reemission" texture for wavelength generation

Material/surface boundary : 4 indices

- outer material (parent)
- outer surface (inward photons, parent -> self)
- inner surface (outward photons, self -> parent)
- inner material (self)

Primitives labelled with unique boundary index

- ray primitive intersection -> boundary index
- texture lookup -> material/surface properties

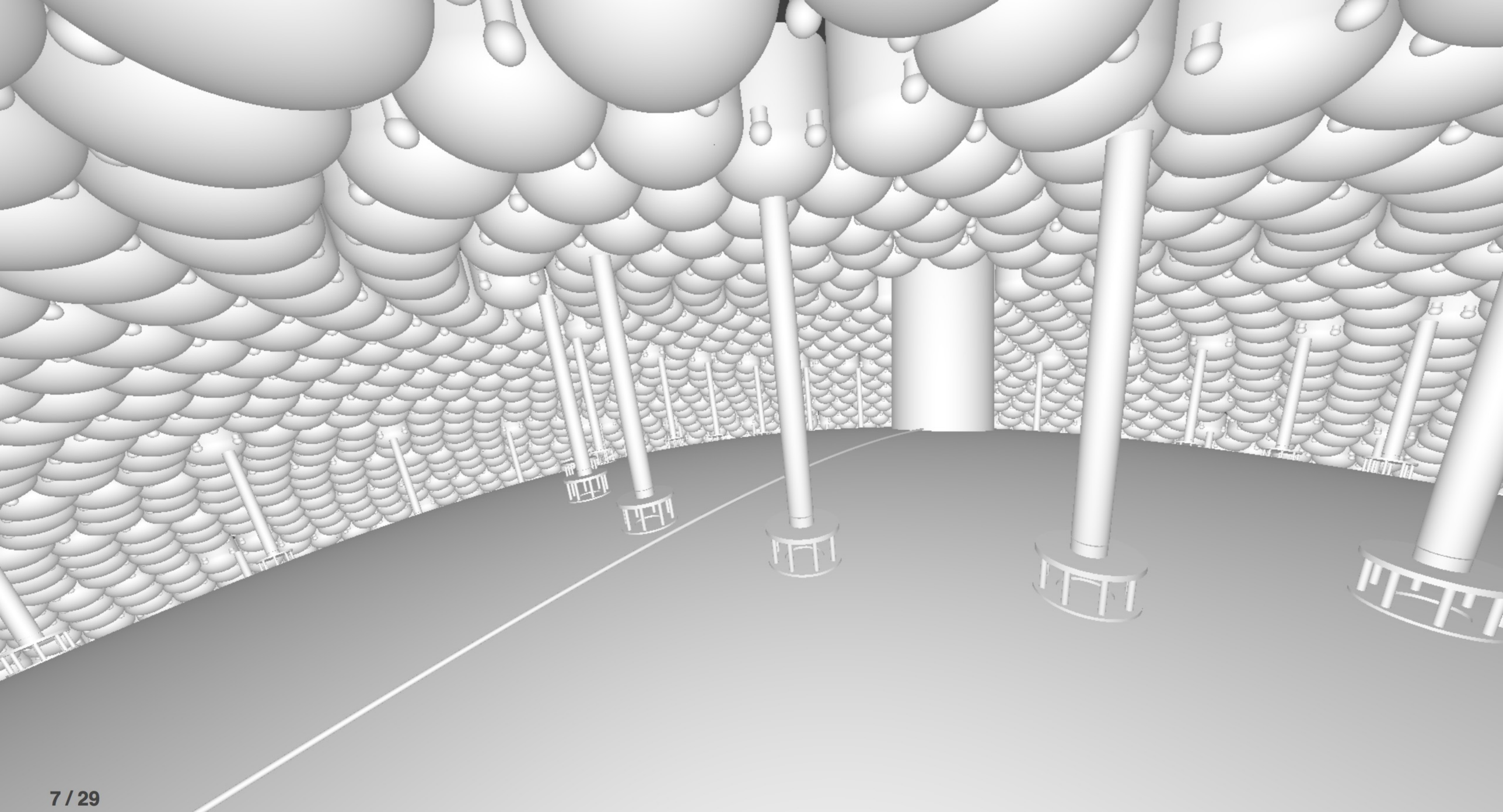
simple/fast properties + reemission wavelength

Opticks : Translates G4 Geometry to GPU, Without Approximation Talk

The translation of geometry structure starts by identifying repeated geometry and its transforms using geometry tree digests for every node.

Effectively the original JUNO geometry of hundreds of thousands of volumes is factorised down to only a handful of repeated assemblies of volumes with arrays of thousands of 4x4 transforms representing the position and orientation of each assembly.

This factorisation of the geoometry allows the use of **instancing** with OptiX and OpenGL which leads to a huge memory savings and performance improvement for simulation and visualization.



Talk

The upshot is that full Geant4 detector geometries can be automatically translated into NVIDIA OptiX geometries.

This is an OptiX ray trace image from the chimney region at the top of the JUNO scintillator.

2020 : Leap in Opticks Awareness : Meetings, Publications >>> New Users

(Feb 2020) Workshop on Efficient Computing for HEP, Edinburgh

- https://indico.ph.ed.ac.uk/event/66/ ❏ "Full Simulation of HEP Detectors with Geant4", Ben Morgan

(May 2020) HSF (HEP Software Foundation) Meeting : GPUs in Simulations

- https://indico.cern.ch/event/921244/ ❏ "Opticks Development Experience : Problems and Successes", Simon Blyth

(10+24 June 2020) HSF Meeting : R&D on accelerators in Simulations 3/9 talks discussed Opticks

- https://indico.cern.ch/event/925887/ ❏ https://indico.cern.ch/event/930881/ ❏
- "e/γ (GPU) calorimeter simulation.. ", John Apostolakis, "Prototyping simulation .. on GPUs", Andrei Gheata
- "Prospects for (LHCb) RICH detector simulation using OptiX in GPUs", Sajan Easo

(Aug 2020) HSF Publication : HL-LHC Computing Review: Common Tools and Community Software

- https://zenodo.org/record/4009114 ❏ Graeme Stewart,... HL-LHC Computing Review discusses Opticks

(Sep 2020) Geant4 : R&D Task Force Meeting 3+ Geant4 teams pursuing GPUs, 1 based on Opticks

- https://indico.cern.ch/event/942142/sessions/363813/#20200915 ❏

1. "G4Opticks for liquid Argon TPCs", Fermilab Geant4 Team, Hans Wenzel
2. "VecGeom@GPU ❏", CERN Geant4 Team, Andrei Gheata

2020 : Leap in Opticks Awareness : Meetings, Publications >>> New Users

Talk

There has been quite a leap in Opticks awareness over the past year.

- a few of the talks and publications discussing Opticks are listed here.
- it is being actively evaluated by an increasing number of groups

Why the leap ?

- i guess my plenary presentation at the CHEP conference at the end of 2019 is the main cause
- also using GPUs with Geant4 has become a hot topic over the past year, with several high profile projects (Celeritas, AdePT) getting started

2020 : Leap in Opticks Awareness >>> Some Users Getting Serious

(Aug 2020) Shanghai SJTU : Next-gen Neutrino Telescope Simulation Workshop

- https://indico-tdli.sjtu.edu.cn/event/238/sessions/109/#20200813
- "Opticks : GPU/Graphics backgroup + Application to underwater neutrino telescope simulations ?", Simon Blyth
- SJTU team evaluating Opticks for deep ocean expt simulations (Fan Hu, Donglian Xu)

(Aug 2020) Snowmass 2021 : Three LoI based on Opticks : Multi-expt Dark Matter group invited me to sign LoI

1. "Opticks : GPU photon simulation via NVIDIA OptiX", Simon Blyth
2. "Fast simulations for Noble Liquid experiments", XENON, DARWIN, LZ teams, Simon Blyth
3. "Simulating Optical Photons in HEP experiments on GPUs", Fermilab Geant4 team

(Aug 2020) NERSC Users Group (NUG) meeting : LZ aiming to deploy Opticks on Perlmutter supercomputer

- https://www.nersc.gov/users/NUG/annual-meetings/nug-2020/
- https://www.nersc.gov/assets/Uploads/1400-Opticks-on-CoriGPU.pdf
- "Opticks on CoriGPU", O.Creaner, 5 authors from : NERSC/LBL/LZ/Stanford/Bristol
- Opticks in Shifter/Docker container -> CoriGPU -> Perlmutter Cray/NVIDIA supercomputer
- Perlmutter : > 6000 NVIDIA A100 (Ampere Architecture GPU)

https://insidehpc.com/2020/05/perlmutter-supercomputer-to-include-more-than-6000-nvidia-a100-processors/

2020 : Leap in Opticks Awareness >>> Some Users Getting Serious Talk

Several groups that have been evaluating Opticks for years are now starting to get serious.

The LZ(LUX-Zepelin) dark matter experiment is the most advanced

- they aim to get Opticks going on the new Perlmutter supercomputer at NERSC

Groups.io Find or Create a Group Help Log In Sign Up

Home
Messages
Hashtags

Opticks opticks@groups.io

Opticks users forum. Opticks is an open source software package providing
simulation, enhancing Geant4 simulations of particle physics detectors with t
Links to presentations and papers on Opticks and how to get and install the

- https://simoncblyth.bitbucket.io
- https://bitbucket.org/simoncblyth/opticks

Group Information

https://bitbucket.org/simoncblyth/opticks
25 Members
33 Topics, Last Post: Jan 20
Started on 6/23/18
Feed

Email Group + Web Archive

`https://groups.io/g/opticks`

25 Members, 33 Topics

Year	2018	2019	2020
New Members	5	9	11

- `opticks+subscribe@groups.io`

Talk

The Opticks Forum and email list is the main way that I support users.

- there are now 25 members
- unfortunately some users still tend to email me directly

G4OpticksTest : Demonstrates Geant4 + Opticks hybrid workflow

(Nov 2020) HSF WLCG Virtual Workshop :
https://indico.cern.ch/event/941278/timetable/#20201123.detailed

- Hans Wenzel, Fermilab Geant4 Team

Worked closely with Hans for several years

- Developed Geant4-Opticks integration API **G4Opticks**
- Recently reported two Geant4 optical surface issues #2305 #2311

Plan for G4OpticksTest

- advanced example distributed with Geant4
- explore use with G4Tasking with 10.7

Using Geant4 and Opticks[a] to simulate liquid Argon TPC's

Hans Wenzel, Krzysztof Genser ,Soon Yung Jun, Alexei Strelchenko
Fermilab
HSF/WLCG workshop
Nov 19-24th 2020

GEANT4 A SIMULATION TOOLKIT

11 / 29

Summary and Plans

Summary:

- We made Simon Blyth's Opticks work with Geant4 10.6.p03.
- A prototype application: G4OpticksTest is available and we plan to make it an advanced Geant4 example. Various geometries are available as gdml.
- Preliminary benchmarking looks very promising.

Plans:

- Implement current Geant4 Scintillation process on GPU.
- Implement wavelength shifting process on GPU.
- Develop Geant4 Task application with Opticks where:
  - Gensteps chunks are collected in-situ during the tracking/stepping loop, once a predetermined chunk size is reached the optical photon propagation is offloaded to the GPU (device), while the rest of simulation and gensteps collection continues on the CPU (host).

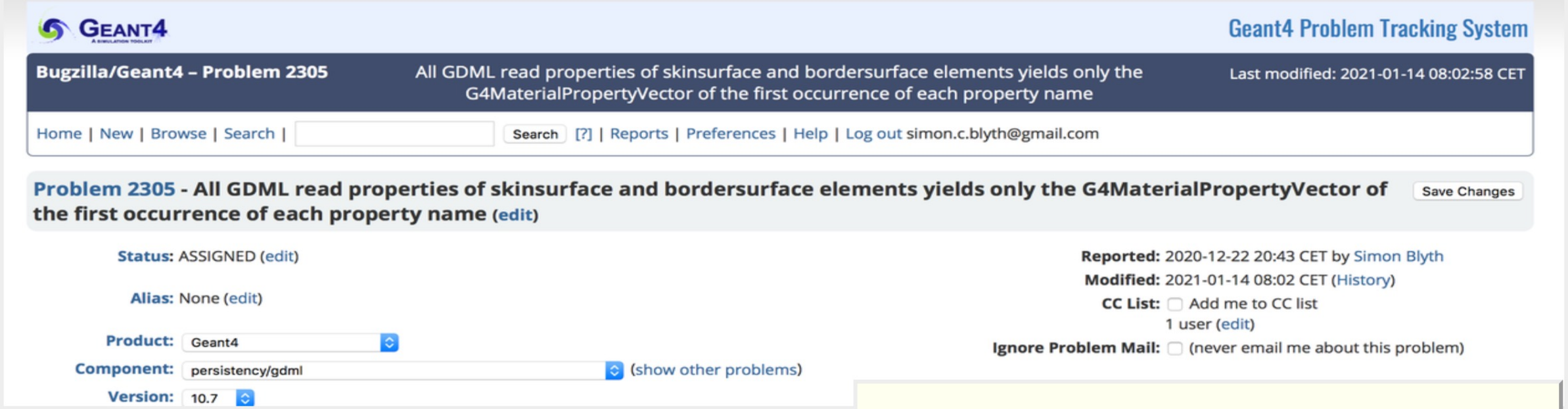
special thanks to Simon Blyth!!

Fermilab

G4OpticksTest : Demonstrates Geant4 + Opticks hybrid workflow Talk

The Fermilab Geant4 Team has been active for several years now with Opticks.

- the inset slides are from Hans Wenzel
- my work with them has shaped the development of the G4Opticks interface.
- also attempts to work with recent Geant4 releases have revealed some Geant4 optical bugs



Geant4 Bug 2305

https://bugzilla-geant4.kek.jp/show_bug.cgi?id=2305

Five Geant4 releases are **unusable**:

- 1060, 1061, 1062, 1063
- 1070
- **fix planned for 1071**

Unfortunately it took me five releases to notice this showstopper

Any GDML using expt. with > 1 optical surface, should not use these five releases of Geant4.

Important to test ahead, checking lastest Geant4:

- natural way to contribute to Geant4 and raises profile
- avoids : surprises, excluded releases
- ease future "official" updating

Talk

If you use GDML and have more than 1 optical surface then I would suggest you should not use these five releases.

It is unfortuate that I did not test Opticks with these Geant4 releases until recently. This optical surface bug tripped a consistency assert in the Opticks geometry translation, so it is immediately apparent when testing Opticks with the Geant4 release.

I would have been able to catch the problem much sooner.

It really demonstrates the importance of testing the latest version of critical externals even when you have no intention to officially updating to them in the near future.

Plus it is a natural way to contribute to your externals and ensures that they will be ready for you when you do need to update.

Problem noticed in first release

Geant4 team seem willing to follow my suggestion that will avoid any lasting impact.

Geant4 Bug 2311

https://bugzilla-geant4.kek.jp/show_bug.cgi?id=2311

Effected Geant4 releases: 1070+

bordersurface std::vector -> std::map API Change

- made Opticks name sort workaround for 1070
- have suggested **creation order index** to improve workaround > 1070

Talk

This bug is actually a change in API.

This Geant4 change in 10.7 was noticed at the first release, and fortunately an Opticks workaround is possible and it looks likely that the next Geant4 release 1071 will be free of both this workaround and the optical surface bug too.

UK SWIFT-HEP grant : SoftWare InFrastructure and Technology for HEP

SWIFT-HEP : Newly funded (1.5M GBP over 3 yrs)

University of Manchester (UOM), Adam Davis

- "... we identified Opticks as a major contribution area ..."

(Jan 2021) SwiftHep/ExcaliburHep workshop presentation

Ben Morgan, University of Warwick + Geant4

- https://indico.cern.ch/event/976081/

UOM : Plan 0.5 FTE over 12 months

- integration of Opticks within Geant4
- widening Opticks applicability
- "... also plan to hire software engineer to assist..."

details yet to be determined

WARWICK
THE UNIVERSITY OF WARWICK

Detector Simulation in ExCALIBUR-HEP and towards SwiftHEP

Ben Morgan

14 / 29

SwiftHEP/ExCALIBUR-HEP Workshop
Online, 14th January 2021

Simulation in SwiftHEP

1. **Build on R&D from ExCALIBUR-HEP on algorithms for EM Physics on GPU**
   - *Continue and strengthen collaboration with AdePT and Celeritas projects*
   - *Extend "Acceleritas" example to fully demonstrate, profile and validate hybrid CPU/GPU Geant4 application*
2. **Contribute to development of example application using Geant4+Opticks for hybrid CPU/GPU optical photon simulation**
   - *In collaboration with FNAL and Geant4 R&D Task Force*
   - *Build UK links with, and expertise in, Opticks*

UK SWIFT-HEP grant : SoftWare InFrastructure and Technology for HEP

Talk

There is some other news on Geant4+Opticks integration.

A group from Manchester and Warwick in the UK has got some funding part of which they want to dedicate to improving and extending the use of Opticks.

So it looks like a few more people will be working on Geant4+Opticks integration this year.

LHCb RICH (Ring Imaging Cherenkov) with Opticks : Sajan Easo

(June 2020) HSF (HEP Software Foundation) Meeting

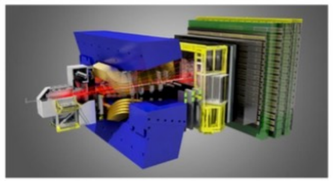
- https://indico.cern.ch/event/930881/
- "Prospects for RICH simulation using Opticks", Sajan Easo (STFC-GB)
- Standard Geant4 costs ~30% CPU time for optical photons
- successful validations

LHCb Opticks Users

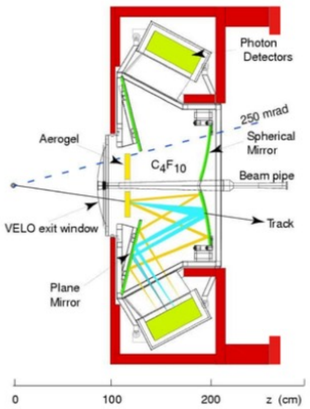
Several LHCb members are using Opticks

- Sajan Easo has reported his investigations

RICH system in LHCb



"LHCb Detector Performance", LHCb Collaboration
Int.J.Mod.Phys.A 30(2015) 1530022



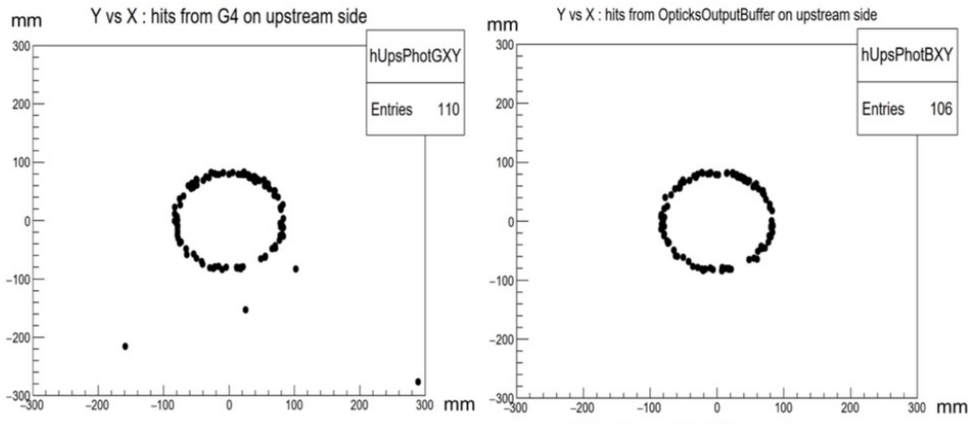
- LHCb has two RICH detectors.

- RICH simulation involves:
  - Creation of photons with Cherenkov and Scintillation process
  - Typically transporting single photons using two mirrors and a quartz window
  - Creating hits in photon detectors like MaPMTs.

- There is an ever increasing need to reduce the CPU time used up by LHCb simulations, as we attempt to simulate billions of events. During LHCb simulation, the RICH takes up approximately 30% of the overall CPU time

- Majority of the time used by the RICH system is in the transport of optical photons. Hence it is useful to focus on improving the CPU time used for this transport .
- Typically in an event with ~250 charged tracks at a luminosity of 2 x 10^{33} $cm^{-2}s^{-1}$ and an average of (41 + 24) signal hits per charged track in (RICH1+RICH2), the number of photons transported is a large number

OptiX and GEANT4



Hits from Geant4 photon transport

Hits from OptiX ray tracing

- Excellent match between the hits obtained from Geant4 and OptiX on the plane at (0,0,90)

- The 4 extra hits in the left plot are from photons which went through this plane and later got reflected back to the same plane. Their new directions were not part of the set uploaded for ray tracing in OptiX.

LHCb RICH (Ring Imaging Cherenkov) with Opticks : Sajan Easo Talk

Opticks is not restricted to neutrino and dark matter experiments.

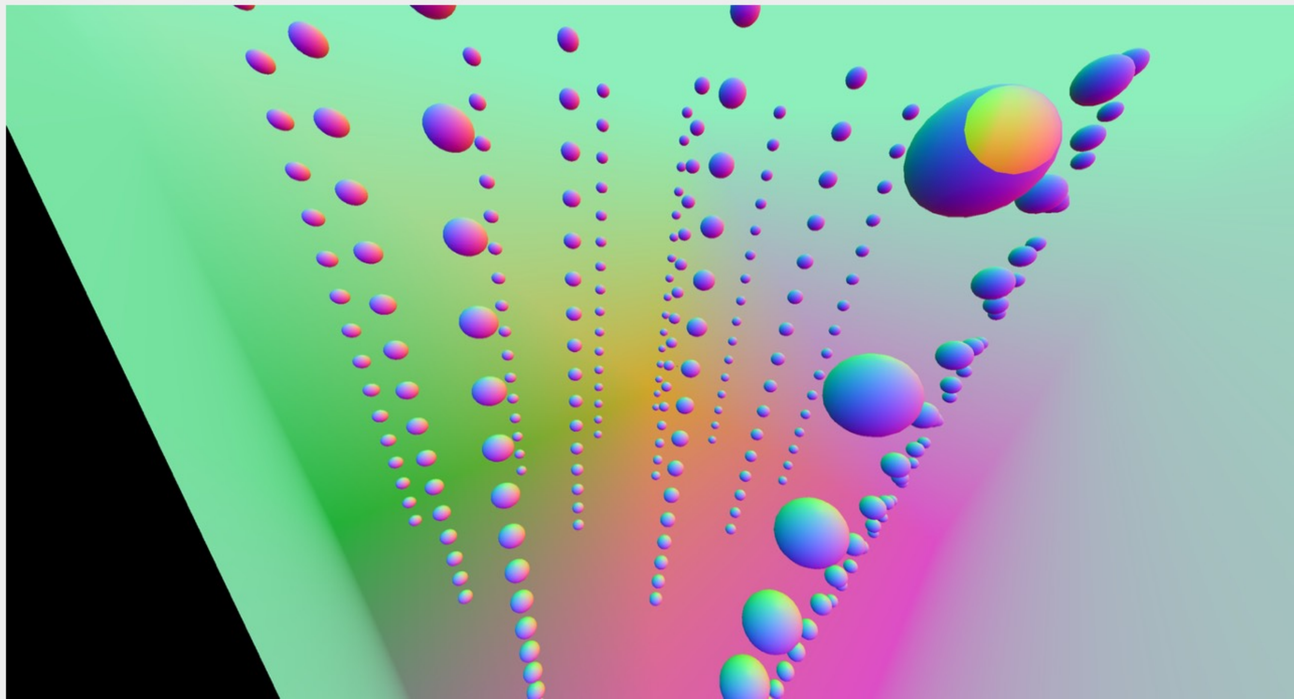
Several people from LHCb have been using it for simulation of RICH detectors (that is Ring Imaging Cherenkov)

These slides are from Sajan Easo who has done some simple validations with the RICH geometry.

Opticks+CORSIKA8 for next-generation neutrino telescope optimization ?

(Aug 2020) Shanghai SJTU : Next-gen Neutrino Telescope Simulation Workshop

- https://indico-tdli.sjtu.edu.cn/event/238/sessions/109/#20200813
- "Opticks : GPU/Graphics backgroud + Application to underwater neutrino telescope simulations ?", Simon Blyth
- SJTU team evaluating Opticks for deep ocean expt simulations (Fan Hu, Donglian Xu)
  - enormous instrumented volumes O(10) km^3 water/ice
  - strings of pods geometry
  - huge numbers of Cherenkov photons propagating across enormous volumes, how to simulate ?
  - CORSIKA 8 + Opticks : trying to setup genstep transport



Opticks+CORSIKA8 for next-generation neutrino telescope optimization ?

Talk

Another idea for using Opticks is to try to integrate it with CORSIKA for optimization of enormous deep sea neutrino telescopes.

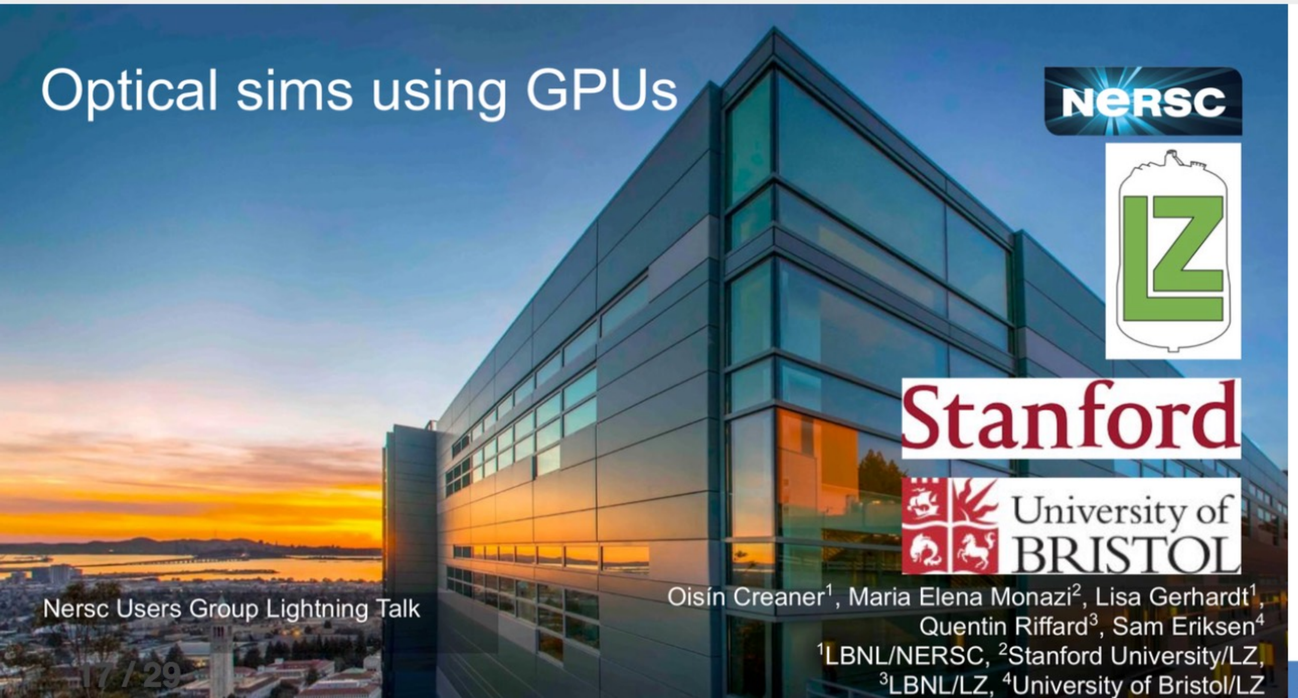
The visualization shows a string of pods geometry set up by a student of Donglian, Fan Hu.

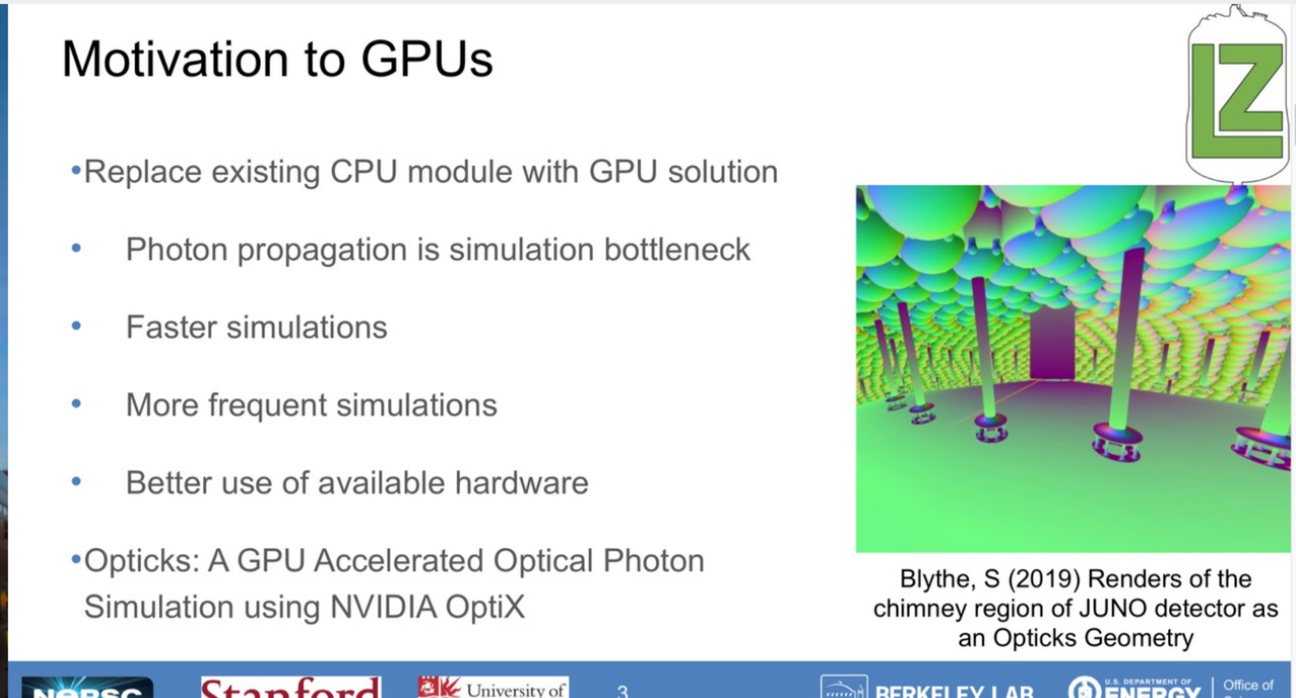
LZ+Opticks history : 2018-2021 : contacts with >5 LZ people

- Summer 2020 : LZ+Opticks simulation identified as candidate for new Perlmutter Cray/NVIDIA supercomputer at NERSC
- Autumn 2020 : LZ+NERSC proposed an NVIDIA assisted Hackathon to optimize LZ+Opticks simulation
  - BUT no point optimizing prior to migration : NVIDIA OptiX 6 -> 7 (an entirely new "Vulkan-ized" API)
  - migration to NVIDIA OptiX 7+ is unavoidable, it beings challenges and opportunities
    - multi-GPU scaling no longer provided, thread-safe OptiX 7 opens possibilities, especially for Opticks Server+Client

(Aug 2020) NERSC Users Group (NUG) meeting : LZ aiming to deploy Opticks on Perlmutter supercomputer

- https://www.nersc.gov/users/NUG/annual-meetings/nug-2020/
- "Opticks on CoriGPU", O.Creaner, 5 authors from : NERSC/LBL/LZ/Stanford/Bristol





LZ+Opticks history : 2018-2021 : contacts with >5 LZ people Talk

LZ has been evaluating Opticks over the past three years and it is now keep to scale up its usage to profit from the new NERSC Perlmutter supercomputer, with more than 6000 NVIDIA A100 GPUs

They proposed a Hackathon to optimize LZ+Opticks simulation.

- but there is little point optimizing prior to migrating to the OptiX 7 API
- after I pointed this out they indicated they could help with the migration

2021 : LZ+Opticks simulation on Perlmutter (>6000 NVIDIA A100)

(8 Jan 2021) Zoom Introduction Meeting of >12 people : LZ/NERSC(6), NVIDIA(5), Opticks(1)

- NERSC + NVIDIA teams: senior engineers + postdoc "workers"
- NVIDIA team : experienced with OptiX 6->7, multi-GPU + multi-threaded CUDA/OptiX

Prepared Opticks codebase orientation for developers

- https://simoncblyth.bitbucket.io/opticks/docs/orientation.html

From February 8:

- plan bi-weekly working meetings
- migration assistance

Adapt Opticks to all new NVIDIA OptiX 7 API

- **necessary** : otherwise forever stuck at *OptiX 6.5* (2019)
  - pre-requisite to *OpticksServer*
- re-implement *optixrap* (*OptiX* wrapper) package
- first: single-GPU only
- expect no change to *G4Opticks* API

Opportunity for Expert Assistance

- NVIDIA GPU developer team
  - real experts advising/optimizing Opticks
  - influence on NVIDIA OptiX direction(?)
- NERSC supercomputer team
  - Docker/Shifter containerization
  - large scale production optimization

Perlmutter: A System Optimized for Science

NERSC

- HPE Cray system providing 3-4x capability of "Cori"
- First NERSC system designed to meet needs of both large scale simulation and data analysis from experimental facilities
  - Includes both NVIDIA GPU-accelerated and AMD CPU-only nodes
  - HPE Slingshot high-performance network will support terabit rate connections to system
  - Optimized data software stack enabling analytics and ML at scale
  - All-flash filesystem for I/O acceleration
- Robust readiness program for simulation, data, and learning applications, and complex workflows
- Delivery begins in 2020

U.S. DEPARTMENT OF ENERGY | Office of

2021 : LZ+Opticks simulation on Perlmutter (>6000 NVIDIA A100) Talk

I was a little surprised by the number of people from NERSC and NVIDIA in the Zoom meeting. I guess NERSC wants its users to rapidly see the benefits from the hundreds of millions of dollars spent on Perlmutter.

Having NVIDIA aware in detail of how OptiX is being used for optical photon simulation could be extremely beneficial down the road.

Also it make a good story for NVIDIA to be helping with something more than making games look pretty. Plus widens the market for its GPUs.

Opticks + JUNO Progress : Efficiency Hit Culling on GPU

- Reduce CPU memory for hits by a factor of the efficiency
- only **collected hits** consume CPU memory

Culling result photon flags set on GPU:

- `EFFICIENCY_COLLECT`
- `EFFICIENCY_CULL`

ε, ε [θ, ϕ] -> GPU Tex

Separate Textures for each sensor type

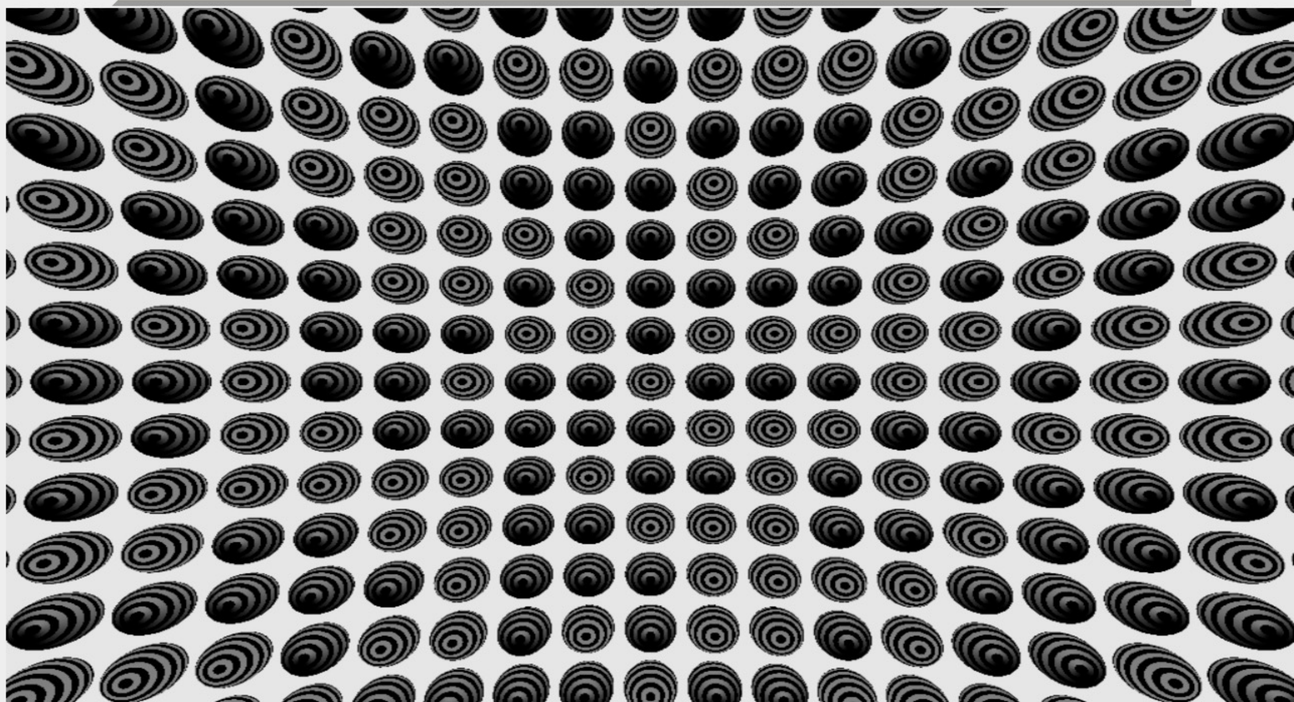
```
G4Opticks::
   setSensorData
   setSensorAngularEfficiency
```

Testing angular efficiency GPU texture machinery

- Mock striped theta efficiency, two random categories
- cosine phi efficiency variation for one category

https://simoncblyth.bitbucket.io/opticks/optixrap/tests/OSensorLibGeoTest.cc.html

https://bitbucket.org/simoncblyth/opticks/src/master/notes/progress.rst



Opticks + JUNO Progress : Efficiency Hit Culling on GPU Talk

- Doing efficiency culling on GPU reduces CPU memory for hits by a factor of the efficiency
- only the collected hits needed by the electronics simulation have to use CPU memory
- BUT : it means you have to translate the efficiency information and upload it to the GPU
  - i did this with angular textures for each sensor type and efficiency buffers
- the culling decision on GPU leads to one of these flags being set, then you can copy back just the EFFICIENCY_COLLECT photons to go into the hit collection

Completing JUNO Hits : G4OpticksHit (4,4) + G4OpticksHitExtra (2,4)

void G4Opticks::getHit(unsigned i, G4OpticksHit hit, G4OpticksHitExtra* hit_extra) const*

Geant4 approach

```
// Simulation/DetSimV2/PMTSim/src/junoSD_PMT_v2.cc
junoHit_PMT* hit = new junoHit_PMT(); // junoSD_PMT_v2::ProcessHits
hit->SetPMTID(pmtid);
hit->SetWeight(1.0);
hit->SetTime(hittime);
hit->SetWavelength(wavelength);
hit->SetKineticEnergy(edep);
hit->SetPosition(local_pos);
hit->SetTheta(local_pos.theta());
hit->SetPhi(local_pos.phi());
hit->SetMomentum(local_dir);
hit->SetPolarization(local_pol);
hit->SetGlobalPosition(global_pos);
hit->SetGlobalMomentum(track->GetMomentum());
hit->SetGlobalPolarization(track->GetPolarization());
// optional extras normally from  NormalTrackInfo/NormalAnaMgr
hit->SetProducerID(producerID);
hit->SetFromCerenkov(is_from_cerenkov);
hit->SetReemission(is_reemission);
hit->SetOriginalOP(is_original_op);
hit->SetOriginalOPStartT(t_start);
hit->SetBoundaryPosition(boundary_pos); // pAcrylic/pInnerWater[1]
```

- **split**: Hit (4,4) + HitExtra (2,4) OR **combine**: HitFull (6,4) ?
- Using **split** assuming extras switched off in large productions ?

[1] set in NormalTrackInfo by `NormalAnaMgr::UserSteppingAction`

G4OpticksHit + G4OpticksHitExtra

```
struct G4OpticksHit // from photon->hit buffers
{  // global+local,... all from (4,4) photon item
    G4ThreeVector local_position ;
    G4ThreeVector global_position ;
    G4double      time ;
    G4ThreeVector local_direction ;
    G4ThreeVector global_direction ;
    G4double      weight ;
    G4ThreeVector local_polarization ;
    G4ThreeVector global_polarization ;
    G4double      wavelength ;
    G4int         boundary ;
    G4int         sensorIndex ;
    G4int         nodeIndex ;
    G4int         photonIndex ;
    G4int         flag_mask ;
    G4int         sensor_identifier ;
    G4bool        is_cerenkov ;
    G4bool        is_reemission ;
};
struct G4OpticksHitExtra // from way->hiy buffers
{  // extras from (2,4) "way" item
    G4ThreeVector boundary_pos ;
    G4double      boundary_time ;
    G4double      origin_time ;
    G4int         origin_trackID ;
};
```

Completing JUNO Hits : G4OpticksHit (4,4) + G4OpticksHitExtra (2,4) Talk

The (4,4) and (2,4) are dimensions of the buffer items. On the GPU it is more efficient to handle quad elements like float4 so Opticks does everything in quads.

One obvious way to provide the global+local information would be to double the size of the photon element. That would have halved the number of photons that can be simulated at once. So transforms are used to convert global into local on the CPU using identity information to get the right transform.

All of the *G4OpticksHit* comes from a (4,4) photon item.

Hit information is split into standard+extra to avoid paying the price for the extra info when it is not needed.

The price is a 50% increase in VRAM and a doubling of GPU to CPU copies.

If the extra info were regarded as essential then increasing the size of the photon item from (4,4) to (6,4) would be appropriate.

Opticks : Three Levels of Performance

- https://bitbucket.org/simoncblyth/opticks/src/master/notes/performance.rst ❏

1. **optical**: time+memory to simulate photons (1 CPU, *N* GPUs)
2. **node**: overall simulation performance (1 CPU, *N* GPUs)
3. **cluster**: overall batch performance creating large samples

For example with *N = 4*

- best **optical** performance
- **node** : ~same as 1 GPU per CPU (dominated by non-optical)
- **cluster** : significantly worse : as limiting concurrent jobs

Experimentation required to find optimum

- CPU only *OpticksClients* -> GPU cluster *OpticksServer*

Flexibility -> maximizes resource usage -> solves problems:

- limited numbers of NVIDIA GPUs
- GPU starvation : difficult to keep them busy

cluster level optimizations

Opticks Server + many Clients

- genstep requests, hits in response
- bring Opticks to CPU only batch nodes
- avoid GPU starvation
- header only implementation : `OpticksClient.hh`

Prototype : Network Transport of NP Arrays

- https://github.com/simoncblyth/np ❏
- depends only on Boost-Asio (Async IO)
- np_server/np_client operational

Opticks : Three Levels of Performance Talk

SMALL

Performance must be considered at three levels because choices optimum for one level will not be the optimum at other levels.

For example dedicating 4 GPUs to 1 CPU would give the best **optical** performance but at **node** level the performance might not be different to that with 1 GPU per node and at **cluster** level you would be limiting the number of concurrent batch jobs hence killing your overall throughput.

Finding the optimum approach requires experimentation.

Opticks has only optimized at **optical** level.

Optimizing at cluster level is expected to give large gains, by maximimizing the use of resources

A big problem at **cluster** level is that there is a limited number of GPU nodes

Developing a distributed Opticks is

OpticksServer + Lightweight OpticksClients

General compute server structure:

- receive/queue/reply *NumPy* arrays (eg gensteps, hits) over TCP
- decouple computation from queue/network infrastructure code

Exploration phase: find/evaluate open source basis projects

- load balancer, proxy : HAProxy, Squid
- task queue : https://taskqueues.com

Plan ? too early to set timeline

- priority depends on experience with GPU constrained Opticks

computing student/postdoc/.. welcome to assist

- needs network/MQ/server/multi-threading experience~interest

Distributed compute advantage

Currently (only on GPU nodes):

- JUNO Offline -> *G4Opticks* -> Opticks/OptiX

Instead : split CPU and GPU compute

On any CPU node:

- JUNO Offline -> *G4Opticks* -> OpticksClient

On GPU cluster:

- OpticksServer->Opticks(**OKOP**)->OptiX

---

splitting -> drastic leap in flexibility

[1] OKOP package : no Geant4, OpenGL : (Opticks structured as tree of ~20 packages according to external dependencies, only the four highest level packages depend on Geant4)

OpticksServer + Lightweight OpticksClients Talk

Called *OpticksServer* for the name-recognition

- develop as general GPU compute

Releases for Opticks+JUNO validation comparing G4+Opticks vs G4

0.1.0 : "Basic Validation" Release : Enabling Statistical hit comparison : Estimate : During February	
Completed:	**Remaining:**
• full JUNO hits : ▪ *G4OpticksHit* : global+local ▪ *G4OpticksHitExtra* : boundary_pos, origin_time, .. • GPU collection angular efficiency culling ▪ ε [θ, ϕ] -> *G4Opticks* -> GPU "angular" texture ▪ *G4Opticks::setSensorData* ▪ reduces CPU hit memory by : ε . ε [θ, ϕ]	• angular efficiency culling : make **run-time** controllable • *G4OpticksHitExtra* : make **run-time optional** ▪ photon size: (4,4)+(2,4) -> (4,4) ▪ half GPU->CPU hit copies • JUNO Offline+Opticks Bridge updates ▪ adapt to new *G4Opticks* API ▪ improve python commandline control of *G4Opticks*
students/postdocs with Geant4 geometry experience can assist with "Basic Validation" distribution comparison	

0.2.0 : "Full Validation" Release : Enabling Random Aligned step-by-step comparison : Estimate March-April	
Completed:	**Remaining:**
• *cfg4* : *CRandomEngine* control random stream ▪ -> random aligned bi-simulation • *cfg4* : *Geant4* SteppingAction emulates *Opticks* ▪ -> *OpticksEvent* arrays -> direct *NumPy* compare	• Integrate *cfg4* random control with *G4Opticks* ▪ adapt to general *G4* environment (prev: optical only)
students/postdocs with NumPy + Geant4 geometry experience can assist with "Full Validation" step-by-step comparison	

Releases for Opticks+JUNO validation comparing G4+Opticks vs G4 Talk

The plan is to make a "Basic Validation" release during February with the features I have described already. There is little remaining to do, so I am confident have the release during Febrary. This will allow basic statistical comparison of hit distributions.

The "Full Validation" release aims to provide random aligned step-by-step comparisons. Validating like this is very good at finding problems as the comparison is not clouded by statistics.

Problems with Opticks are mostly geometry problems, so experience with Geant4 geometry is important.

Summary and Links

Opticks : state-of-the-art GPU ray traced optical simulation integrated with *Geant4*.
Now integrated with JUNO Offline
Pioneer users welcome

Next Steps

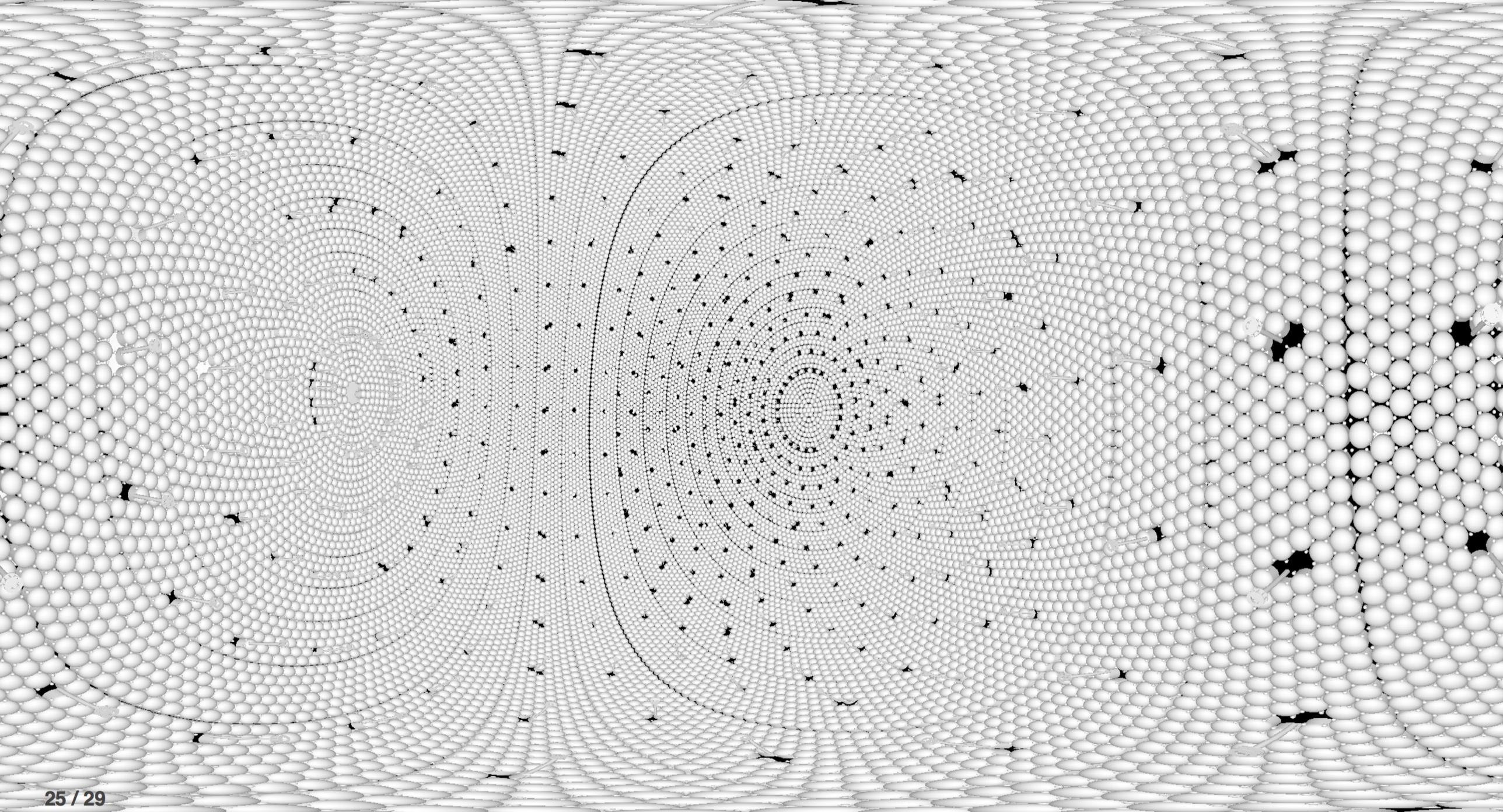
- basic validation release
- migrate to NVIDIA OptiX 7
- full validation release

- **Opticks gained many new users in 2020, some getting serious**
- JUNO Opticks users are needed to assist with validation

https://bitbucket.org/simoncblyth/opticks ❏	code repository
https://github.com/simoncblyth/opticks/releases ❏	.zip .tar.gz archives
https://simoncblyth.bitbucket.io ❏	presentations and videos
https://groups.io/g/opticks ❏	forum/mailing list archive
email:opticks+subscribe@groups.io	subscribe to mailing list

Summary and Links Talk

While it is great that Opticks is gaining many new users with lots of interest and activity from Geant4 and also now NVIDIA the downside of is that it takes times to support them all.



Talk

Background slides start from here

Integration of JUNO + Opticks : Opticks is a JunoENV External

Add Opticks to an existing installation (SVN trunk r3990+):

```
bash junoenv libs all opticks
```

- pre-requisites : NVIDIA Driver, CUDA, OptiX

Four mandatory envvars :

```
export CMTEXTRATAGS=opticks
# -> Opticks ON in JunoENV + CMT
export OPTICKS_COMPUTE_CAPABILITY=70  # -> nvcc flags
export OPTICKS_CUDA_PREFIX=/usr/local/cuda-10.1
export OPTICKS_OPTIX_PREFIX=/usr/local/OptiX_650
# location of CUDA and OptiX installs
```

Build all JUNO externals, including Opticks (takes ~1hr, 13G) :

```
jlibs(){
    mkdir -p $JUNOTOP && cd $JUNOTOP
    svn co https://juno.ihep.ac.cn/svn/offline/trunk/installation/junoenv && cd $JUNOTOP/junoenv
    local libs=$(bash junoenv libs list | perl -ne 'm, (\S*)@, && print "$1\n"' -)
    for lib in $libs ; do
        bash junoenv libs all $lib || return 1
    done
}
```

- https://juno.ihep.ac.cn/mediawiki/index.php/Offline:Installation#Install_Opticks_using_JunoENV

Opticks Build Generalized

Builds against CMAKE_PREFIX_PATH

- Boost, CLHEP, XercesC, Geant4
- -> Opticks uses JUNO externals

opticks-config (**pkg-config** .pc file based)

- integrates with non-CMake projects, eg CMT

releases on github : latest `v0.1.0-rc2`

https://github.com/simoncblyth/opticks/releases/tag/v0.1.0-rc2

Integration of JUNO + Opticks : Opticks is a JunoENV External Talk

Now I transition to progress with integrating Opticks with the Offline.

- **Opticks is now a JunoENV external**
- that means you can install Opticks with that one line
- actually to install all Offline externals you can use the bash function below

Opticks has pre-requisites : CUDA, OptiX and the NVIDIA GPU Driver that need to be separately installed. It is not appropriate to automate this, as low level system drivers are involved users need to follow NVIDIA instructions for your system.

Where these are installed are communicated via envvars.

As Opticks is an optional external, it is necessary to switch it on somehow. The CMTEXTRATAGS envvar is used to do this, when that contains the string "opticks" it signals JUNOEnv and CMT to enable Opticks in the builds.

More details are in the wiki at this link.

Integration of Offline + Opticks : Where WITH_G4OPTICKS used ?

Offline Opticks : `#ifdef WITH_G4OPTICKS ... #endif` + `if(m_opticksMode > 0){ ... }`

```
cd $JUNOTOP/offline/Simulation/DetSimV2 ; find . -type f -exec grep -l WITH_G4OPTICKS {} \+

./DetSimOptions/src/DetSim0Svc.cc                    ## initialize/finalize
./DetSimOptions/src/LSExpDetectorConstruction.cc     ## translate geometry to GPU

./PhysiSim/src/DsG4Scintillation.cc     ## collect Scintillation gensteps, excluding reemission
./PhysiSim/src/LocalG4Cerenkov1042.cc   ## collect Cerenkov gensteps
./PhysiSim/src/DsPhysConsOptical.cc     ## use LocalG4Cerenkov1042 rather than G4Cerenkov

./PMTSim/src/PMTSDMgr.cc           ## setup separate PMTHitMerger m_pmthitmerger_opticks
./PMTSim/src/junoSD_PMT_v2.cc      ## bulk hit creation+collection at EndOfEvent
```

`junoSD_PMT_v2::Initialize`

- setup extra **hitCollection_opticks** (temporarily doing both GPU+CPU sim, with separate hits + merger)

`junoSD_PMT_v2::EndOfEvent_Opticks`

- invoke `G4Opticks::propagateOpticalPhotons` with the gensteps collected
- bulk populate **hitCollection_opticks**

- https://juno.ihep.ac.cn/mediawiki/index.php/Offline:Installation#building_the_offline_-DWITH_G4OPTICKS

Integration of Offline + Opticks : Where WITH_G4OPTICKS used ? Talk

Not everyone has an NVIDIA GPU, so Opticks is optional.

Because of this all usage is "hidden" behind:

1. WITH_G4OPTICKS compilation macro
2. opticksMode runtime switch

The Opticks integration changes code in 3 packages

- DetSimOptions
- PhysiSim
- PMTSim

DetSimOptions
: handles lifecycle + passing the geometry world pointer to Opticks for translation

PhysiSim
: collects Gensteps from Scintillation and Cerenkov processes

Integration of Offline + Opticks : tut_detsim.py options

tut_detsim.py options relevant to Opticks usage : TODO: user friendly Opticks control interface

`--opticks-mode 0`
: do not use Opticks GPU propagation, the default

`--opticks-mode 1` (currently SVN trunk r3990)
: performs both CPU and GPU optical simulations put GPU hits into separate **hitCollectionOpticks**

`--no-guide_tube`
: change geometry skipping the guide tube torus

`--pmt20inch-polycone-neck`
: change geometry of LPMT necks avoiding use of torus

`tds` : **Example Bash Function :**

```
tds-(){ python $JUNOTOP/offline/Examples/Tutorial/share/tut_detsim.py $* ; }
tds(){ tds- --opticks-mode 1 --no-guide_tube --pmt20inch-polycone-neck --evtmax 2 gun ; }
```

- https://juno.ihep.ac.cn/mediawiki/index.php/Offline:Detsimuserguide/UsingOpticks ❏

Integration of Offline + Opticks : tut_detsim.py options Talk

tut_detsim.py options relevant to using Opticks are collected here

--opticks-mode is the main control

Currently both CPU and GPU simulations are done, with GPU hits going into a separate hit collection.

- Trying to use standard geometry with torus exits at Opticks geometry translation.
- Trying to use non-zero opticks-mode with code not compiled WITH_G4OPTICKS exits at initialization.

More details in the wiki

JUNO Geometry : Remove Torus to enable translation to GPU

tut_detsim.py options (in SVN trunk r3990)

`--no-guide_tube`
skip the guide tube torus

`--pmt20inch-polycone-neck`
"cylinder - torus" -> polycone

- NNVT_MCPPMT_PMTSolid
- Hamamatsu_R12860_PMTSolid

`--pmt20inch-simplify-csg`
simpler+faster CSG, same geometry

- avoids Inner_Separator anti-pattern

Why Avoid Torus ?

Double heavy quartic root finding

- fragile (crashes in some OptiX versions)
- precision problems in CSG combinations
- mere presence in OptiX kernel kills RTX mode performance by factor 10

Alternative Approaches To Cope With Torus

- triangulated torus
- TODO : ray marching (aka sphere tracing) signed distance function (SDF) model

JUNO Geometry : Remove Torus to enable translation to GPU Talk

Torus intersection is such a problem, that the best solution is to avoid torus : I think it is perfectly acceptable to do so, because the guide tube is not important for optical photons and the torus based PMT neck is cosmetic only in my opinion.