# CaTS: Integration of Geant4 and Opticks

Hans Wenzel

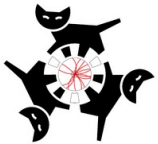
Soon Yung Jun
Krzysztof Genser

**FERMILAB-CONF-23-101**

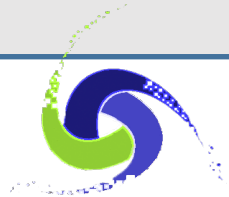
GEANT4 A SIMULATION TOOLKIT

## Outline

- Motivation:
  - The computational challenge for TPCs based on liquid Argon (LArTPCs).
  - Simulation of optical photons: an ideal application to be ported to GPU's.
- Opticks.
- CaTS is an advanced example Geant4 application.
  - CaTS workflow.
  - Performance.
- artg4tk/larg4: making Opticks available to the LArTPC experiments.
- Summary.



**CaTS: C**alorimetry **a**nd **T**racking **S**imulation



GEANT4
A SIMULATION TOOLKIT

Fermilab

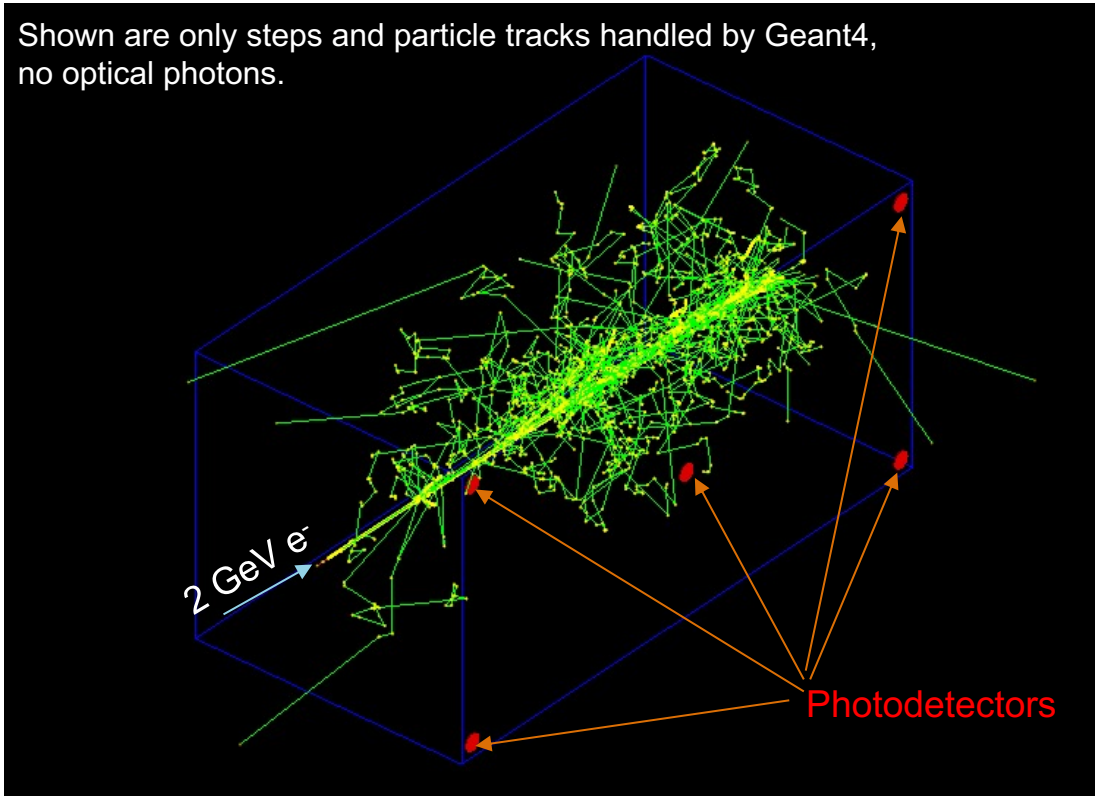
# The computational challenge for TPCs based on liquid Argon (LArTPCs):

Test Detector Geometry:
Liquid Argon: x y z: 1 x 1 x 2 m (blue)
5 photo detectors (red)
photon yield (no E-field): 50000 γ/MeV
single 2 GeV electron (shower not fully contained)
(low Z=18, low $\rho = 1.78$ g/cm$^3$).

- **~ 7x10$^7$** VUV scintillation photons are produced/event.
- Using Geant4 (11.1.p01) to simulate photon generation and propagation o using a single core on an Intel® Core i9-10900k@ 3.7Ghz takes :

**~ 10 minutes/event**

(Compared to **0.034 seconds/event** without optical photon simulation) → LArTPC-Experiments use look up tables and parameterizations instead of full simulation for photon response.



# Simulation of optical photons: an ideal application to be ported to GPU's.

- Only one particle type is involved (optical photon), but many of them (~$10^7$/event)→ allows for massive parallelism (low latency, no big fluctuations in computing time).
- No new particles types besides optical photons are produced.
- Only a few physics processes need to be implemented on the GPU. The processes are:
  - G4Cerenkov (generate photons),
  - G4Scintillation (Reemission) (generate photons),
  - G4OpAbsorption,
  - G4OpRayleigh,
  - G4OpBoundaryProcess,
  - G4OpWLS (not yet implemented, need it for LArTPCs).
- These processes don't need a lot of input data (collected in so called GenSteps for the Cerenkov and Scintillation processes)→ little data transfer from host to device.
- Only a small fraction of photons reach the Photodetectors and produce a PhotonHit → so very little data to transfer from device to host.
- Optical ray tracing is a well-established field → benefit from available efficient algorithms (OptiX®).
- Use NVIDIA® hardware and software (NVIDIA® CUDA, NVIDIA® OptiX ® ).

# Opticks

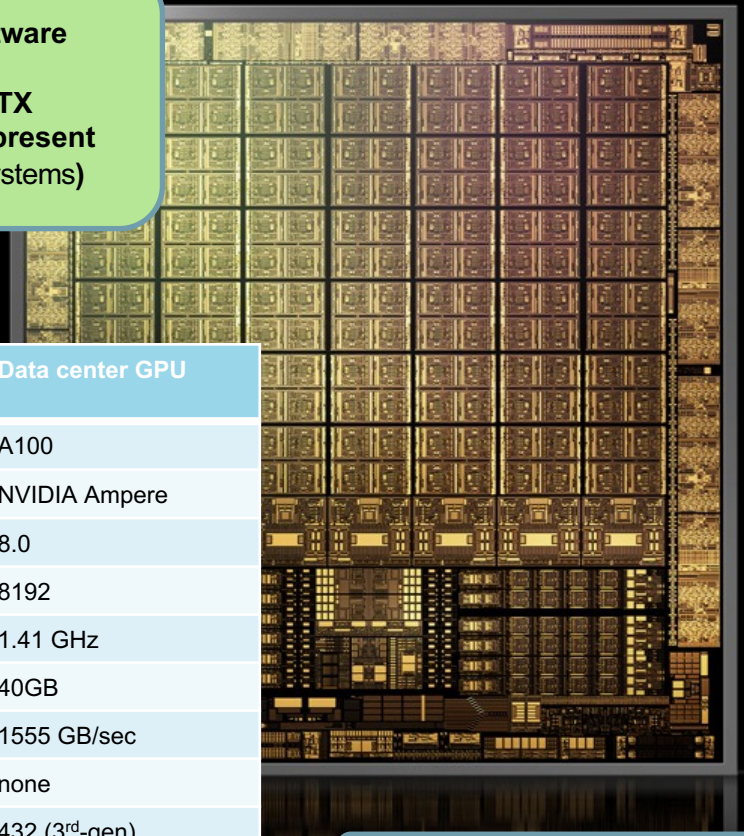
Opticks is an open-source project developed by Simon Blyth: https://bitbucket.org/simoncblyth/Opticks/
See also talk at this conference. There are 2 major versions: legacy Opticks based on OptiX ® 6 using the G4Opticks API and reengineered Opticks based on OptiX ® 7 using the G4CXOpticks API.
Opticks accelerates optical photon simulation by:

- Translating the Geant4 geometry to OptiX ® without approximation.
- Implementing the Geant4 optical processes on the GPU.
- Integrating NVIDIA GPU ray tracing (accessed via NVIDIA OptiX ®).

G4(CX)Opticks provides an API to interface Geant4 and Opticks. The Geant4 advanced example CaTS (**C**alorimetry **a**nd **T**racking **S**imulation) uses this API to implement a hybrid workflow:

- Geant4 on the CPU/host handles all particle types but the optical photons.
- The Geant4 Cerenkov and Scintillation processes are still used to calculate the number of optical photons to be generated at a given step and to provide all necessary quantities to generate the photons on the GPU.
- The information collected is the so called GenStep which are different for Cerenkov (needs e.g betainverse) and Scintillation (needs e.g. scintillation time constants).
- Copying GenSteps to the GPU→ more efficient than e. g. copying optical photons.
- Generation and tracing of optical photons is offloaded to Opticks (GPU/device) at stepping level whenever a certain number of photons is reached.

**Opticks will only run on:**
**NVIDIA® hardware and NVIDIA® software**
**Software: NVIDIA® CUDA, OptiX**
**OptiX 6: allows to select/deselect RTX**
**OptiX 7: RTX cores are used when present**
(RTX is not usually available on HPC systems)




| | Graphics card | Data center GPU |
|---|---|---|
| | GeForce RTX 3090 | A100 |
| architecture | NVIDIA Ampere | NVIDIA Ampere |
| Compute capability | 8.6 | 8.0 |
| CUDA cores | 10,496 | 8192 |
| Boost Clock | 1,7 GHz | 1.41 GHz |
| Memory | 24 GB | 40GB |
| Memory bandwidth | 936 GB/sec | 1555 GB/sec |
| RT cores | 82 (2nd-gen) | none |
| Tensor cores | 382 (3rd-gen) | 432 (3rd-gen) |
| Shared Memory size | 64kB | up to 164 kB |

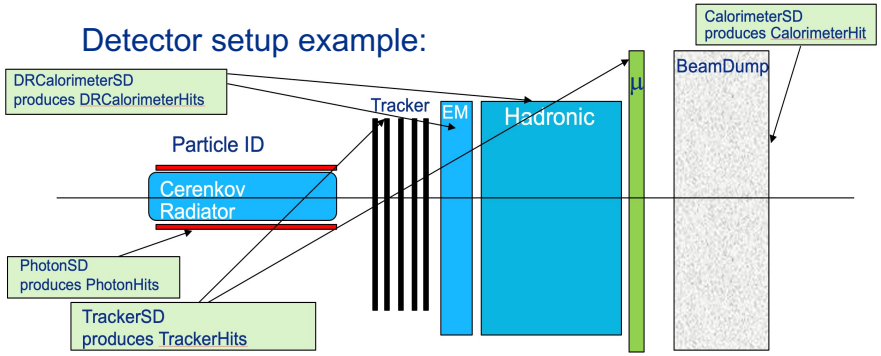
**RT core:** based on bounding volume hierarchy (BVH), a commonly used acceleration structure in ray tracing, ray-triangle intersection.

# CaTS: Calorimeter and Tracker Simulation

- Advanced Geant4 example application (introduced with Geant4 11.0).
- No changes to Geant4 required to integrate Opticks! Only make use of provided interfaces: UserActions, Sensitive Detectors...
- Modular and extendible, allows to build detector setups from predefined components.
- Use GDML with extensions for flexible Detector construction. GDML extensions are used to:
  - Assign sensitive detectors to logical Volumes. A library of various sensitive detectors is provided.
  - Assign step-limits and energy cuts to logical Volumes.
  - Assign visualization attributes.
- Creation of Hit collections and ROOT IO based IO thereof is automated.
- Currently supports legacy/new Opticks interface.
- Uses G4PhysListFactoryAlt to define and configure physics at runtime via command line option

  ../CaTS -g simpleLArTPC.gdml -pl 'FTFP_BERT+OPTICAL+STEPLIMIT' -m time.mac
- G4(CX)Opticks/Geant4 is a runtime/build time option.
- Collection of Scintillation and Cerenkov Gensteps by Geant4.

Detector setup example:



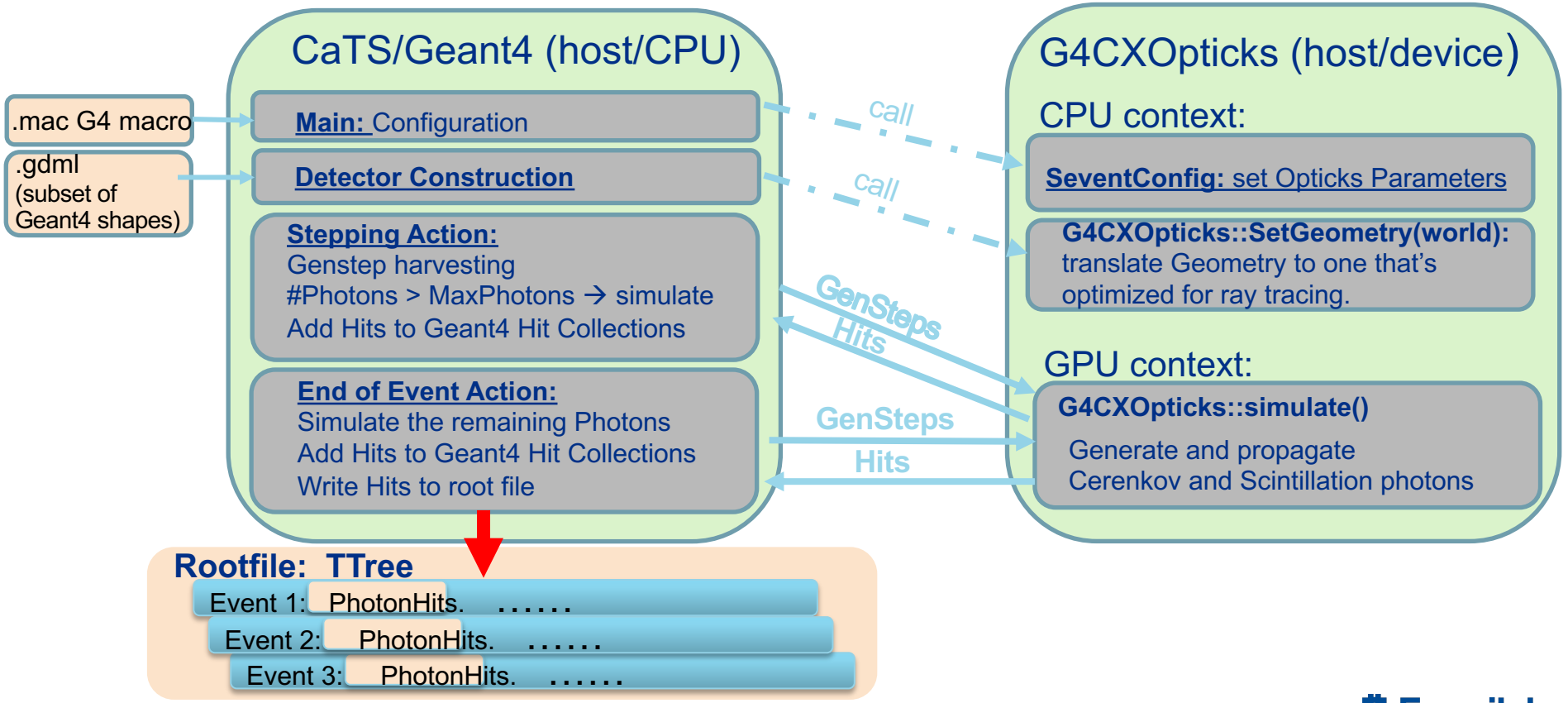

https://geant4.kek.jp/lxr/source/examples/advanced/CaTS/,
https://github.com/hanswenzel/CaTS (development)

# Recent developments

Re-implementing Opticks for OptiX ® 7 required huge changes due to the new and very different OptiX ® 7 API → This gave an opportunity to redesign the Opticks code. Goals of re-implementation: flexible, modular GPU simulation, easily testable, less code. For details see previous presentation.

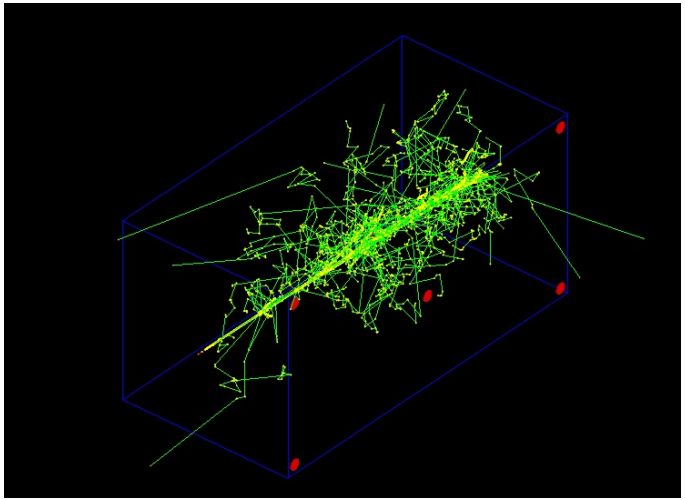
- CaTS has been modified to use the new Opticks API. The CaTS workflow has been adjusted accordingly. User Actions were utilized → no changes to Geant4 itself required.
- CaTS/Opticks were modified to work with Geant4 API changes introduced in 11.1.
- With legacy versions of Opticks (based on Optix 6) we observed speed ups in the order of $2x10^2$ . Evaluation and optimizing the performance with recent updates is in progress.

# CaTS workflow using the new version of Opticks based on OptiX®7:

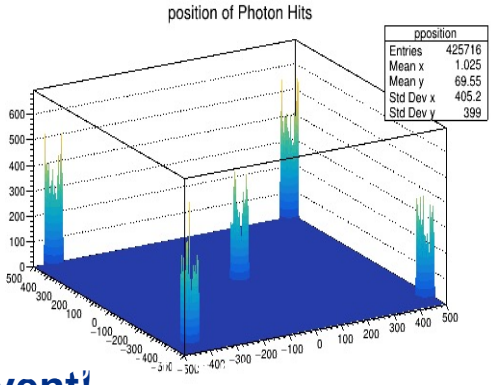


# Performance:

| Hardware: | |
|---|---|
| CPU | Intel® Core i9-10900k@ 3.7 GHz, 10 CPU cores |
| GPU | NVIDIA GeForce RTX 3090 @ 1.7 GHz, 10496 cores |
| **Software:** | |
| Geant4: 11.0, Opticks based on OptiX® 6 | |



| Number of CPU threads | Geant4 [sec/evt] | Opticks [sec/evt] | Gain/speed up |
|---|---|---|---|
| 1 | 330 | 1.8 | 189x |




**→It becomes feasible to run full optical simulation event by event!**

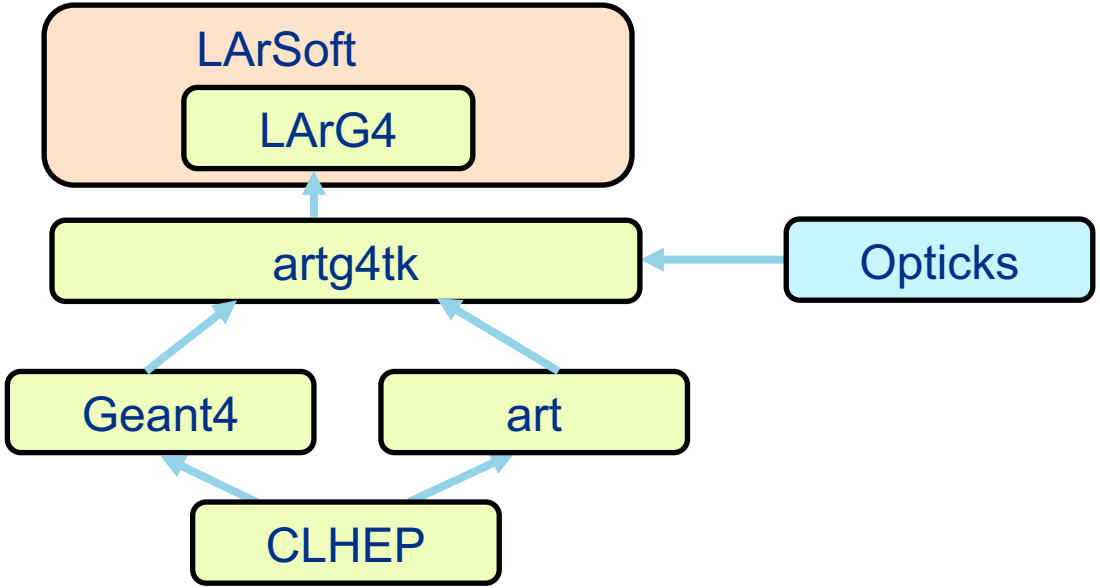


Fermilab

# artg4tk/larg4: making Opticks available to the LArTPC Experiments

**artg4tk:**

- General Geant4 module for the art Event Processing Framework used by various Fermilab experiments: https://github.com/art-framework-suite/art-g4tk
- Very flexible. Geant4 User Actions (Stacking, Stepping, Tracking, Event, Run…) are implemented as Art Framework Services which can be selected and configured at run-time.
- Physics and Detector configuration is identical to CaTS.
- Geant4 data objects (e.g. Hits) are added to the RootIO based art event record.




**larg4:** is based on artg4tk. It is a module for LArSoft: https://larsoft.org/about-larsoft/. All LArSoft dependencies (e.g. lardata objects) are encapsulated here: https://github.com/LArSoft/larg4

In the process of integrating Opticks with artg4tk/larg4,

# Summary

- Geant4 advanced example CaTS was made available.
- It demonstrates a Geant4 hybrid workflow where the generation and tracing of optical photons is offloaded to a GPU using Opticks, while the rest of the simulation is done on the CPU.
- Modified Opticks to work with the latest version of Geant4 (v11.1.p01).
- In the process of integrating Opticks with artg4tk/larg4 to make it available to the LArTPC based experiments.