R&D Overview

Witek Pokorski

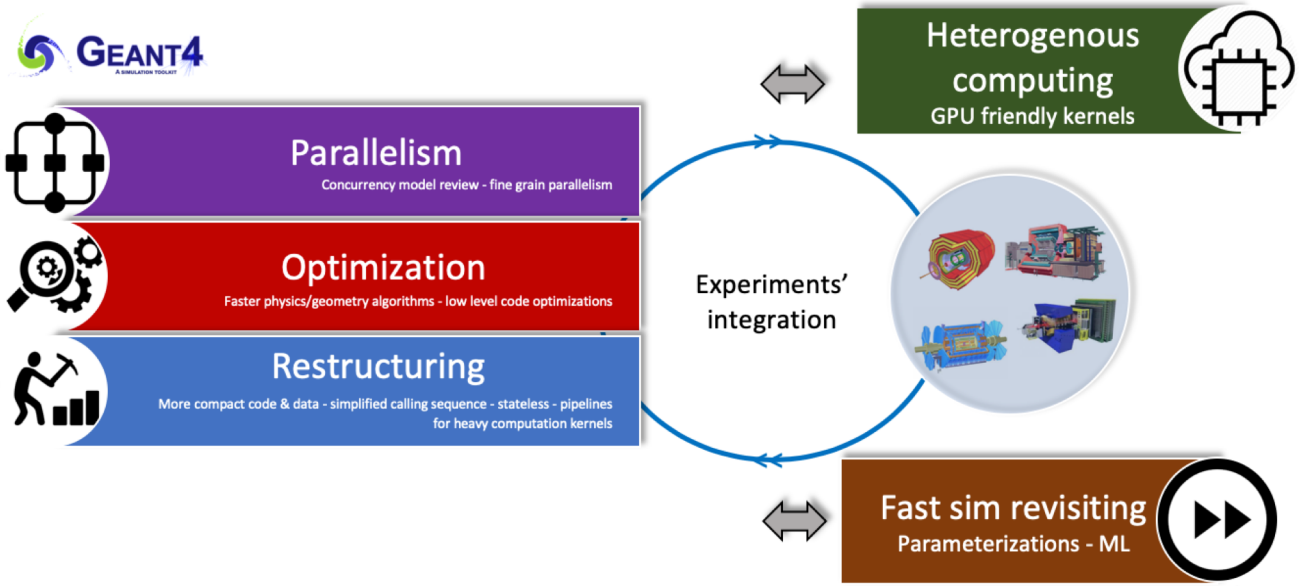
for the R&D activities

20/09/2021

26th Geant4 Collaboration Meeting

Three Main Axes of Development

- **Improve, optimise and modernise** the existing Geant4 code to gain in performance for the detailed simulation
  - Re-structure the code to make possible major changes (task-oriented concurrency, specialisation of the physics, better data formats, etc.)



- Trade precision for performance using **fast simulation techniques** both with parameterisations and with ML methods, and integrate them seamlessly in Geant4
  - Use detailed simulation to 'train networks' or to 'fit parameters' that later can deliver approximative detector responses well integrated within Geant4
- Investigate the **use of accelerators** such as GPUs
  - With novel approaches for organising the computational work

G4HepEm

- G4HepEm and Specialized Stepping/Tracking in Geant4 presented last Tuesday

- dedicated G4HepEm Project talk this Wednesday

G4HepEm and Specialized Stepping/Tracking in Geant4

Jonas Hahnfeld, Benjamin Morgan, Mihály Novák

G4HepEm: motivations in a nutshell

- initiated by the **Geant4 EM physics working group** as part of looking for solutions **to reduce the computing performance bottleneck** experienced by the **HEP detector simulation** applications
- **targeting** the most performance critical part of the HEP detector simulation applications, i.e. the **EM shower generation** covering(initially) e^-/e^+ and γ particle transport
- the main goal is to investigate the **possible computing performance benefits of**
  - **providing alternative, highly specialised** (for particle types, e^-/e^+, γ and HEP applications) **optional stepping loops** beyond the current general one
    $\Longrightarrow$ **giving up the** *"unutilised"* **flexibility with the hope of some performance gain**
  - having a very **compact and efficient implementation of** all the related **run time functionalities** required for an EM shower simulation
    $\Longrightarrow$ **compact run time library** and **data layout with the hope of some performance gain**
- the main design principles
  - separation of initialisation- and run-time functionalities $\Longrightarrow$ in order to have a compact run-time library
  - separation of data and functionality $\Longleftarrow$ since data are filled at initialisation- while used at run-time
- resulted in a run-time **EM shower simulation library with** many attractive characteristics such as the device(**GPU**) side **support** of all related computations (utilised in **AdePT**) or its **stateless** property that, together with its simplicity, provides an excellent domain to check many further interesting ideas
- see the **initial presentation** or the one at the last **Geant4 technical forum** on G4HepEm for more details

G4HepEm

- prototype showing impressive CPU performance gains
- details of the integration and interfacing to the current Geant4 event loop require further discussion, but the general approach already giving strong hints of the direction to go (specialized, streamlined code)

Setup: TestEm3, 100k e^-, 10 GeV, 24 threads on `AMD Ryzen 9 3900` (**default EM settings**)

	Physics List	Specialised Tracking	difference
G4NativeEm	500 s	426 s	-14.8 %
G4HepEm	459 s	373 s	-18.7 %
difference	-8.2 %	-12.4 %	-25.4 %

Setup: `cms2018`, 1000x the same `gg2ttbar` event, 24 threads on `AMD Ryzen 9 3900` (**optimised EM**)

	Physics List	Specialised Tracking	difference
G4NativeEm	2889 s	2747 s	-4.9 %
G4HepEm	2847 s	2660 s	-6.6 %
difference	-1.5 %	-3.2 %	-7.9 %

Note: significant performance gain due to the specialised tracking of e^-/e^+ and γ even already using GEANT4 native processes that is boosted further with `G4HepEm` (even in its current, preliminary phase)

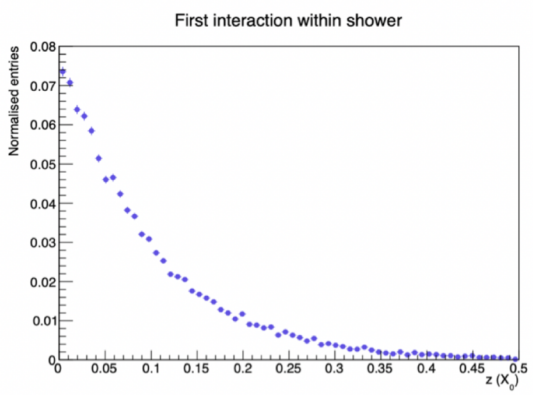
Fast simulation

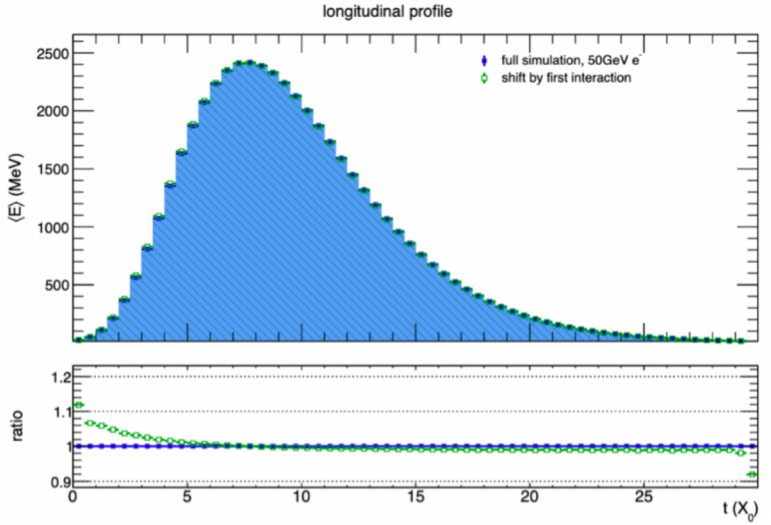
- see presentation by Dalila
- classical
  - activity resuming now
- ML-based

Anna Zaborowska

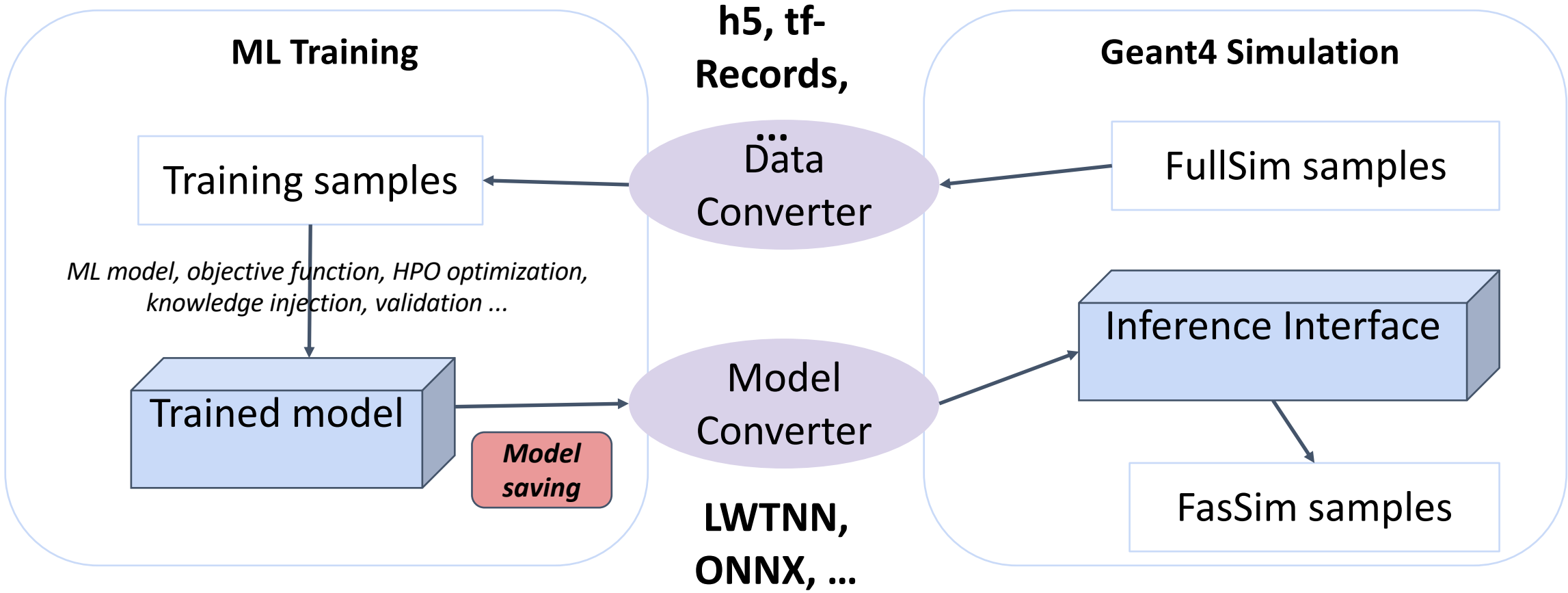
FastSim: Classical Parametrization

- Continuous integration of tuning tools and generalization procedures
- Tuning procedures of parameters of GFlash-based models
  - Start of shower tuning
  - Transverse shower profile tuning



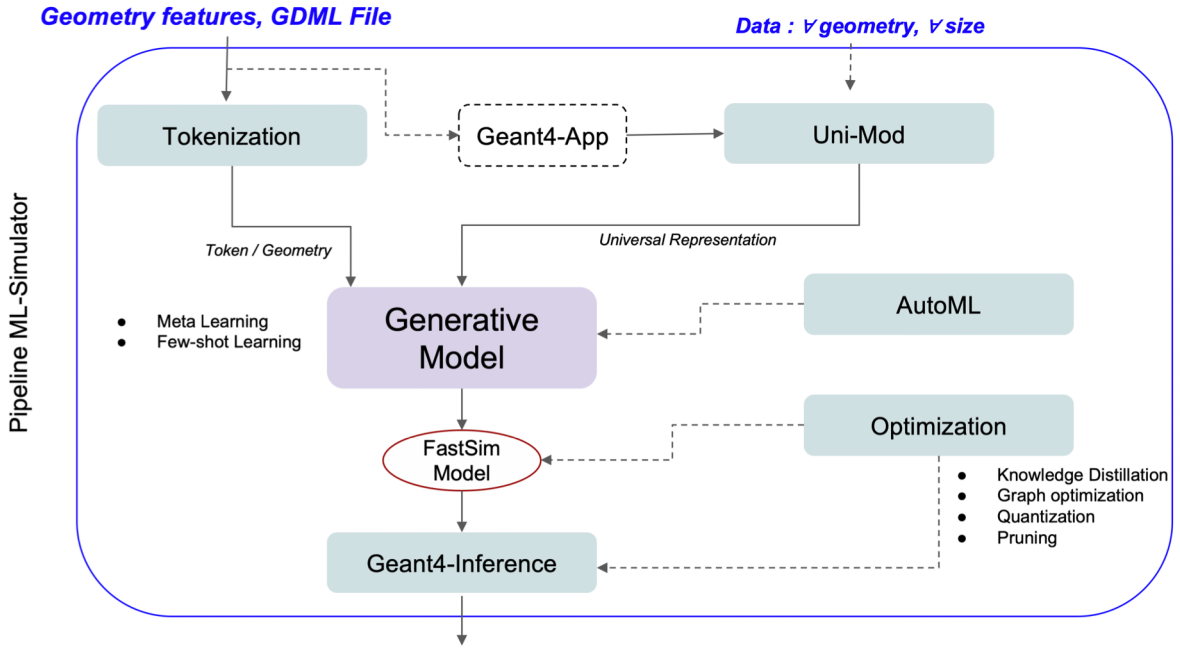


From ML training to Geant4 fast simulation



Dalila Salamani

Towards a generalizable generative modeling



- Work in progress
  - Adding the **CMS** (ECAL) to the set of geometries
  - Re-design of the key components for generalizability and adaptability
  - ML model **meta-learns** P(shower | **energy**, **angle**, **geometry**)

Meta Learning : learning-to-learn “Fast”

Reptile

On First-Order Meta-Learning Algorithms

Alex Nichol and Joshua Achiam and John Schulman
OpenAI
{alex, jachiam, joschu}@openai.com

arXiv:1803.02999

Abstract

MAML

Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

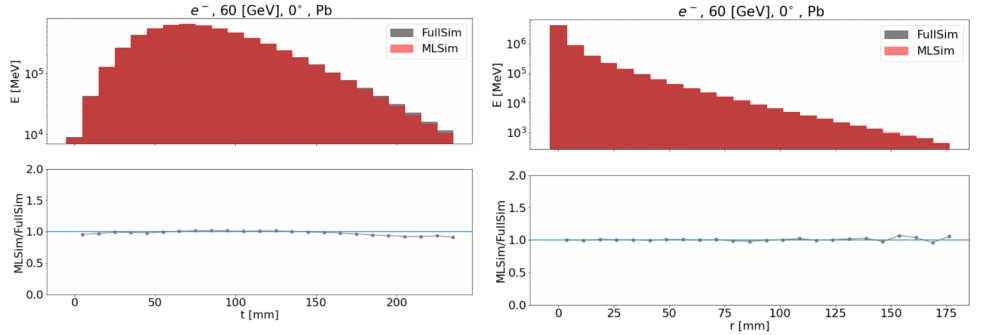
Chelsea Finn, Pieter Abbeel, Sergey Levine

arXiv:1703.03400

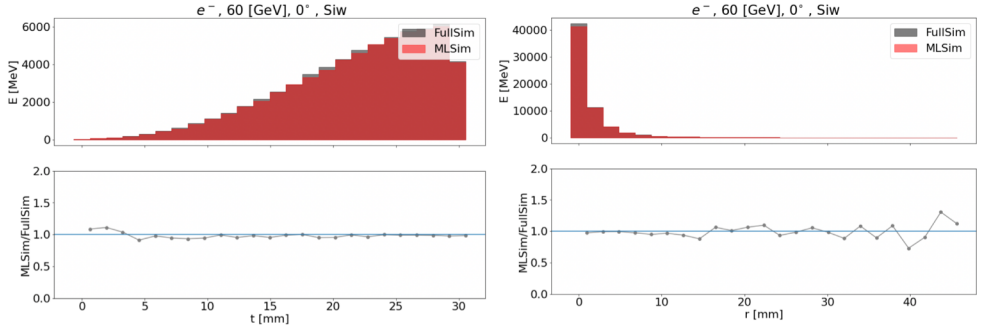
We propose an algorithm for meta-learning that is model-agnostic, in the sense that it is compatible with any model trained with gradient descent and applicable to a variety of different learning problems, including classification, regression, and reinforcement learning. The goal of meta-learning is to train a model on a variety of learning tasks, such that it can solve new learning tasks using only a small number of training samples. In our approach, the parameters of the model are explicitly trained such that a small number of gradient steps with a small amount of training data from a new task will produce good generalization performance on that task. In effect, our method trains the model to be easy to fine-tune. We demonstrate that this approach leads to state-of-the-art performance on two few-shot image classification benchmarks, produces good results on few-shot regression, and accelerates fine-tuning for policy gradient reinforcement learning with neural network policies.

Algorithm 1 Reptile (serial version)

PBWO$_4$

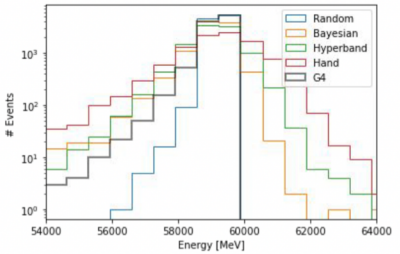


SiW



Automatic Machine Learning (AutoML)

- Summer project
  - Report by Poliana Nascimento Ferreira: https://indico.cern.ch/event/1060366/
- Automatically search the best hyperparameters according to a metric
  - Combined ML-Physics metric (image reconstruction, total energy, energy per layer)
- Compared to a grid search AutoML has the advantage of changing more than one hyperparameter at the same time
- AutoML approaches:



	Random Search	Bayesian Optimization	Hyperband
+	Simple	Tracking of the tuning process Approximate objective function for fast scoring	Fast Explore larger space Tracking of the tuning process
-	No control of the tuning process	Too long to approximate a good enough function	Discard some trials too fast

Opticks integration

- see report from Hans

Integration of Opticks[1] and Geant4 (an advanced example: CaTS)

Hans Wenzel
Krzysztof Genser
Soon Yung Jun
Alexei Strelchenko

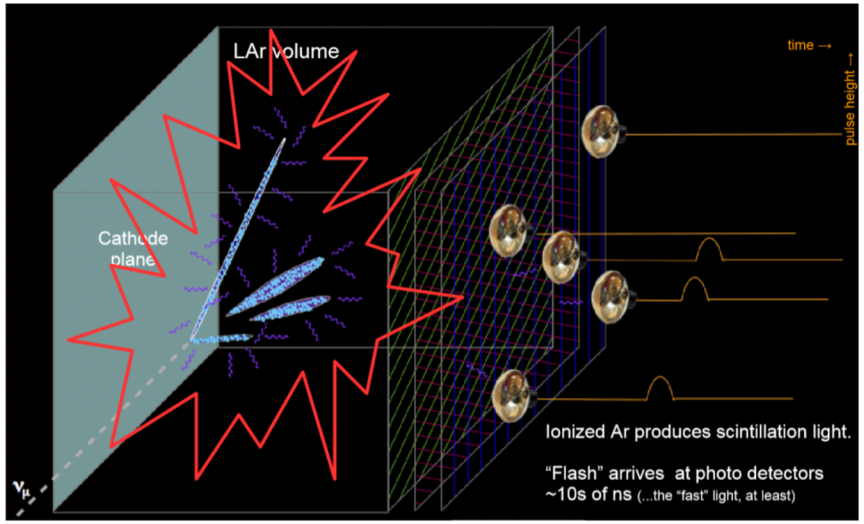
Fermilab-SLIDES-21-106-SCD

GEANT4 A SIMULATION TOOLKIT

[1]developed by Simon Blyth
(Institute of High Energy Physics, Chinese Academy of Sciences)

Motivation

Liquid Argon TPC's are the technology of choice for many neutrino experiments: DUNE, protoDUNE, μBoone, LArIAT, ICARUS, SBND… as well as dark matter searches: DarkSide, ARGO, …



See e.g.,: Dorota Stefan:
https://indico.cern.ch/event/575069/contributions/2326563/attachments/1363382/2064171/LArPrinic

GEANT4 A SIMULATION TOOLKIT

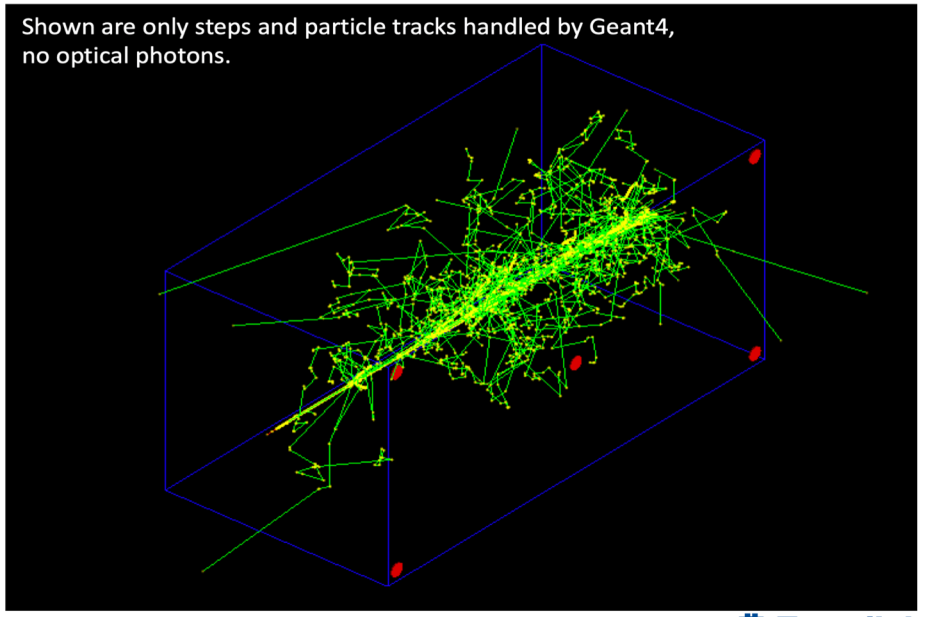
The computational challenge: Simulating a single 2GeV electron shower results in about 70 million VUV photons

Simple 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 2GeV electron (shower not fully contained)
(low Z=18, low $\rho = 1.78$ g/cm^3).

- 70 000 000 scintillation photons are produced.
- Using Geant4 to simulate photon generation and propagation on the CPU takes:

~41 minutes/event

(Compared to 0.056 sec/event when no optical photon simulation)



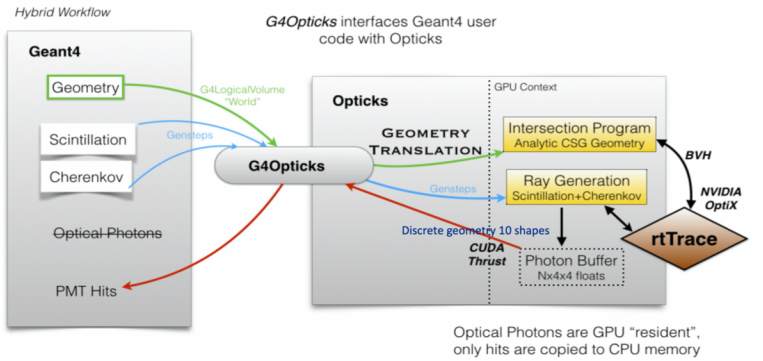
G4Opticks

G4Opticks (part of Opticks): interfaces Geant4 user code with Opticks. It defines a hybrid workflow where generation and tracing of optical photons is offloaded to Opticks (GPU/device) at stepping level when a certain amount photons is reached. Geant4 (CPU/host) handles all other particles.

The Geant4 Cerenkov and Scintillation (C/S) processes are only 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 is different for Cerenkov and Scintillation (C/S).

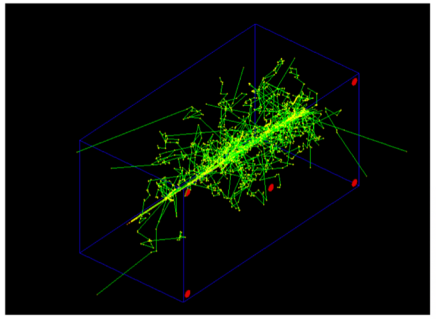
Photon Hits are collected at the end of the G4Opticks call and added to the event hits collection.

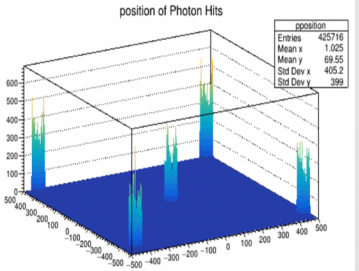
Figure from Simon's presentation



Performance:

Hardware:	
CPU	Intel(R) Core i7-9700K 3.6GHz 32 GB memory.
GPU	GeForce RTX 2070 CUDA Driver Version /11.3 CUDA Capability: 7.5 VRAM: 7981 Mbytes Cores: 2304





Timing results (Geant4 10.7.p01):	
Geant4 optical physics	2438 sec/event
G4Opticks, RNGmax[1] 10	6.45 sec/event
G4Opticks RTX enabled, RNGmax[1] 10	2.72 sec/event
G4Opticks, RNGmax[1] 100	6.86 sec/event
G4Opticks RTX enabled, RNGmax[1] 100	2.87 sec/event

1) Memory pre allocated for pre-initialized (at installation) curandState files to load.

Geant4/(Geant4 + Opticks) comparison:
2438/6.45 = 378 (x 2.4 ~ 900 with RTX) x speed up
RTX Ray tracing hardware acceleration is usually not available on HPC platforms

Summary and Outlook:

- The work was performed as part of the Geant4 R&D Activity.
- We integrated Opticks with Geant4 (>10.7.p01). Small changes to Geant4 interfaces were introduced when needed.
- Provided timing/memory usage results. For our test case we observe an overall speedup by a factor of 390/900 (without/with RTX) compared to running Geant4 optical simulation on the CPU.
- The example CaTS:
  - Provides ROOT persistency for Hit collections.
  - Provides Sensitive detector plugins for different detector types.
  - Provides various detector geometries and configurations as gdml.
  - Gensteps chunks are collected in-situ during the stepping loop, once a predetermined chunk size, that can be set via a Geant4 messenger class, is reached the optical photon propagation is offloaded to the GPU (device). The rest of simulation and Gensteps collection is done on the CPU (host). The remaining Photons are processed at the end of Event.
- GPU Hardware is developing fast (accelerating). All results here are based on GeForce RTX 20 Series Turing platforms with first generation RTX.
- Expect performance boost from moving to OptiX™ 7 (Simon Blyth is working on it).

AdePT status

Accelerated demonstrator of electromagnetic Particle Transport

Guilherme Amadio (CERN), John Apostolakis (CERN), Predrag Buncic (CERN), Gabriele Cosmo (CERN), Daniel Dosaru (EPFL), Andrei Gheata (CERN), Stephan Hageboeck (CERN), Jonas Hahnfeld (CERN), Mark Hodgkinson (Sheffield University), Benjamin Morgan (Warwick University), Mihaly Novak (CERN), Adrian Antonio Petre (ISS Bucharest), Witold Pokorski (CERN), Alberto Ribon (CERN), Graeme A Stewart (CERN), Pere Mato Vila (CERN)

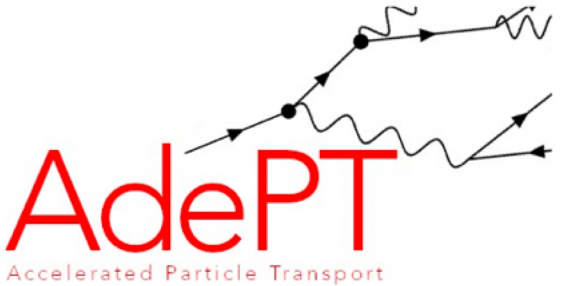
26th Geant4 Collaboration Meeting
Sep 16, 2021

see AdePT presentation

R&D for EM physics transport simulation on GPU

- Project started one year ago, trying to address inability to use GPU cards for detailed simulation of collider experiments
  - Transforming simulation code to be more GPU-friendly, see how far we can go…
  - Some initial ideas shown last year
- Now close to having a complete prototype for e^+, e^- and γ shower simulation
  - Completing the set of EM interactions with the last models supported by G4HepEm
  - Magnetic field propagation in detector geometry
  - Code producing user `hits` data transferred from the GPU back to the host
- Support for both GPU standalone and hybrid modes
  - Integrating with Geant4 CPU workflow, `stealing` particles and offloading to GPU
- Optimizing performance on realistic use-cases
  - To understand usability and realistic benefits for the current workflows

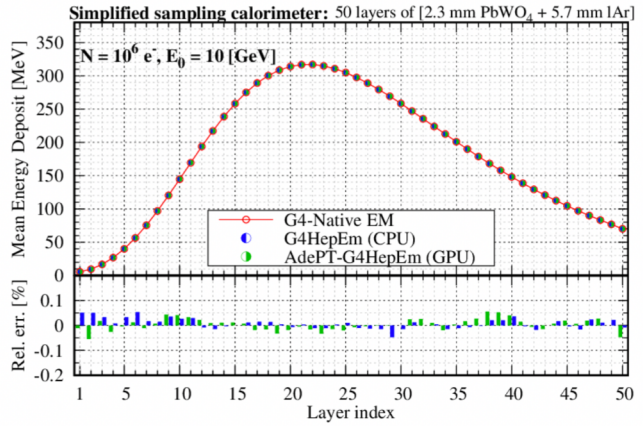
AdePT ingredients: G4HepEm physics



- G4HepEm: compact library of EM processes for HEP
  - Covers interactions of $e^-/e^+/\gamma$, including multiple scattering (not yet integrated into AdePT)
  - Separation between data initialization (relies on Geant4) and computation of cross-sections and final state sampling during run-time (standalone, independent of Geant4)
- Design of library very supportive for heterogeneous simulations
  - Implicit support in the interfaces: Standalone functions without global state
  - Explicit support for transfer of physics tables and other data structures to GPUs
  - Makes it possible to re-use > 95% of the code from G4HepEm for GPU shower simulation
- For details, see presentations by M. Novák in December, at Geant4 Technical Forum in March, and recent status update
  - Also talk in the Kernel parallel session and in Plenary 3 next Wednesday

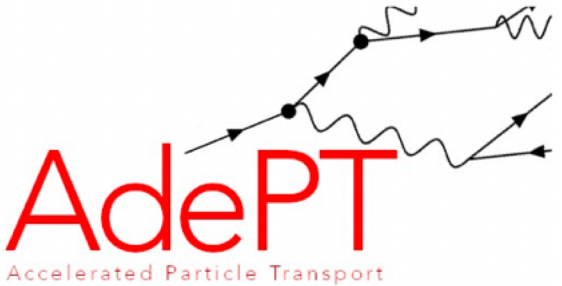
G4HepEm validation

- Previously validated simulation results on GPU against Geant4
  - Agreement at per-mill level in the mean energy deposit (shown below) and other quantities (number of secondaries, number of steps, charged track length)



AdePT ingredients: VecGeom geometry

- Using VecGeom as GPU-aware library describing the detector geometry
  - Same CPU and GPU algorithms compiled separately for host and device
  - Geometry data re-constructed on GPU based on the transient data on the host
  - Navigation layer customized for GPU use
- Improving gradually GPU support
  - Developed custom optimised navigation state, single-precision support
  - Moving to a simple looper to an optimized BVH navigator (see talk of G.Amadio)
  - Adopting modern CMake GPU support
  - See talks in the Geometry working session
- Moving forward: specializing the VecGeom GPU navigation support
  - Important step towards portability and performance

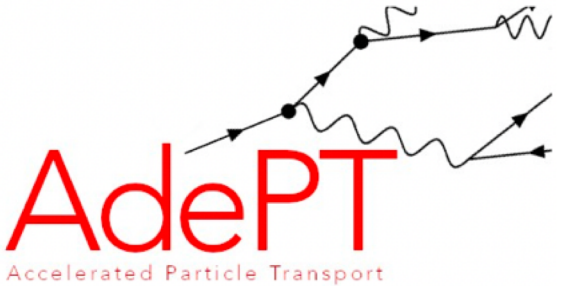


More recipes: AdePT Geant4 integration

- Integrate AdePT GPU workflow with standard Geant4 simulation on CPU
  - One of the main goals of the AdePT prototype
  - We target a Geant4 plug-in to offload work to GPU for increased throughput and improved energy/cost efficiency
    - If proven efficient, this can become a plug-in for offloading part of Geant4 simulation on device
- Inserted in the Geant4 stepping loop using the *FastSimulationModel* interface
  - Intercepts and removes from stack all EM particles entering the region(s) where the “model” triggers
  - Buffering and showering them in AdePT (now synchronous)
  - Return escaping particles and hits to CPU
  - Flushing the buffer until empty before ending the Geant4 event
- Detailed showcase of the recipe in Witek’s presentation

Advanced examples

- Sampling calorimeter example (TestEm3)
  - Constant field on/off, Physics = ALL except MSC + photoelectric, Geant4 and GPU versions
  - Quite similar throughput for Geant4 MT on AMD Ryzen 9 3900 (12C/24T), versus AdePT on GeForce RTX 2070 SUPER after some performance improvements (J. Hahnfeld talk)
  - MT example shows limited benefits in the current workflow due to bottleneck in feeding concurrently input particle buffers
- Generic geometry with generic scoring example (CMSapp, currently PR)
  - Field on/off, Physics = ALL except MSC + photoelectric, AdePT version only for now
  - Doing initialization of Geant4, G4HepEm and VecGeom data structures based on GDML info
  - Generic scoring in all volumes, in the future only for sensitive volumes
  - Example recipe will be adopted in the integration prototype



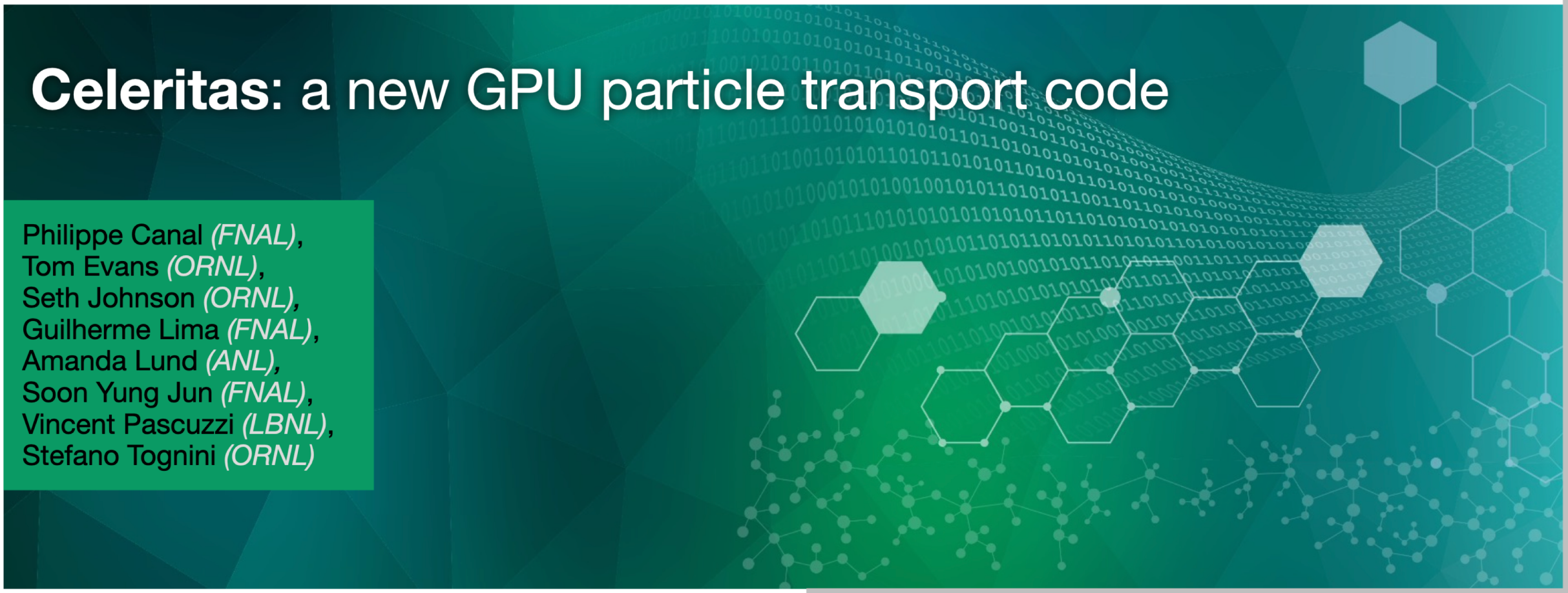
Overlook

- Converging to a prototype combining complete particle transport features
  - Most functionality demonstrated in examples, combined CPU-GPU workflow for a realistic LHC geometry being developed
- R&D work and improvement of components
  - Several improvements and adaptations in VecGeom: navigation and state, optimizers, single-precision support
  - Support for EM physics in a GPU-friendly way
  - RNG optimisation, magnetic field support, workflow optimization
- We can answer the question “can it work?”, now assessing the “how efficient?” part
  - Only a realistic full prototype can give us an unbiased measure of performance

OAK RIDGE National Laboratory

Celeritas: a new GPU particle transport code

Philippe Canal *(FNAL)*,
Tom Evans *(ORNL)*,
Seth Johnson *(ORNL)*,
Guilherme Lima *(FNAL)*,
Amanda Lund *(ANL)*,
Soon Yung Jun *(FNAL)*,
Vincent Pascuzzi *(LBNL)*,
Stefano Tognini *(ORNL)*



ORNL is managed by UT–Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Celeritas overview

- **GPU-targeted** re-implementation of a subset of Geant4 physics leveraging both HEP physics community and HPC/GPU particle transport domain knowledge
  - Data layout and memory access patterns are **critical** to GPU performance
  - Program flow requires structural reorganization
- Short-term application: offloading EM tracks from Geant4 to GPU (*Acceleritas* bridge library)
- Long-term application: direct access library for higher performance integration into LHC analysis workflows

Celeritas status

- Fully implemented on device and host:
  - Standard EM physics models for e, γ except MSC
    (bremsstrahlung, ionization, photoelectric with atomic relaxation, …)
  - Transport loop components
    (process/model selection, secondary production, energy loss fluctuations, …)
  - Geometry navigation using VecGeom
  - Uniform and nonuniform magnetic field
- Current efforts (through end of calendar year 2022):
  - Testing and benchmarking integrated transport loop
  - Physics validation

Conclusion

- R&D continuing along the 3 axes
  - improvements to the CPU implementation
  - fast simulation
  - GPU-based prototypes

- all projects bringing very interesting results at different levels of advancement (some production ready, some early prototypes)

- first GPU-based EM calorimeter simulation prototype expected before end of the year to allow an assessment of the performance and to decide on the future direction