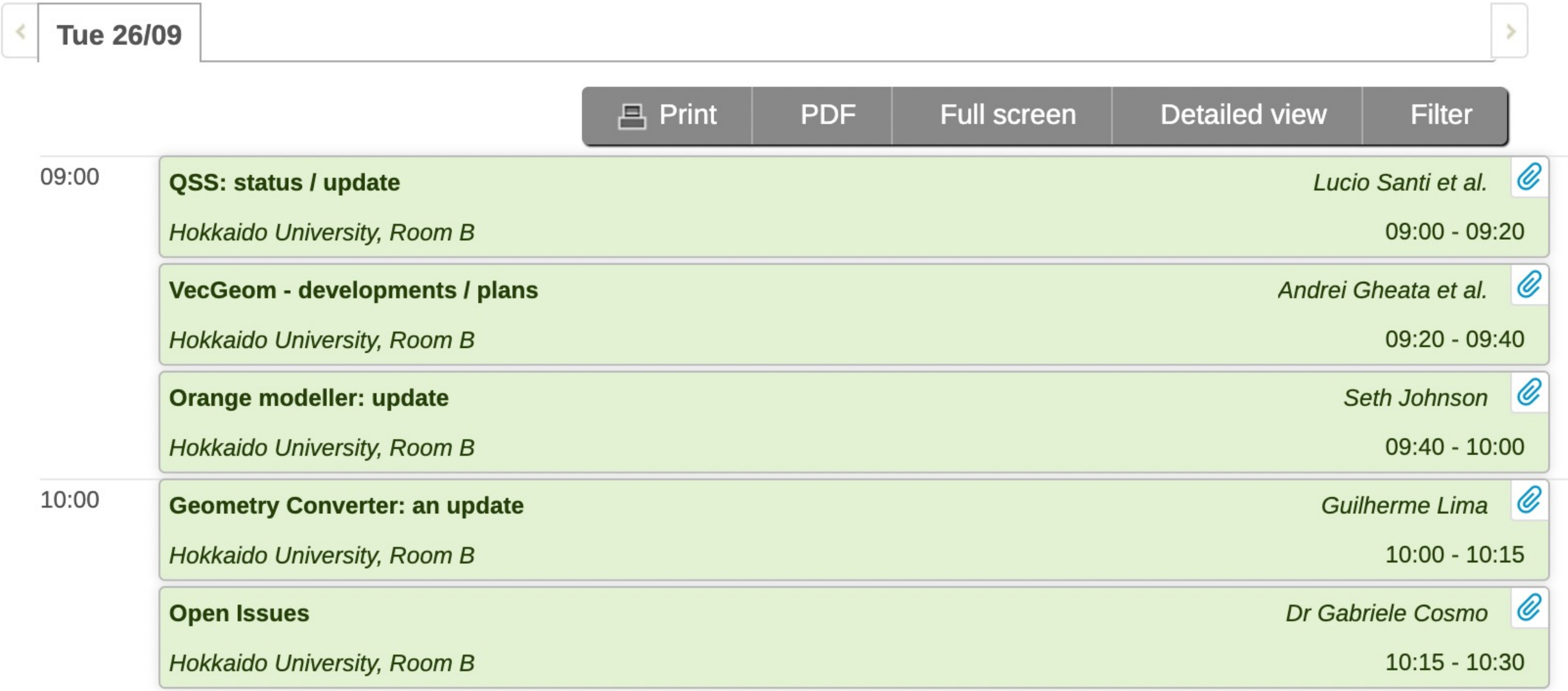
Summary of
Parallel 2B: Geometry

J. Apostolakis



Tue 26/09

09:00

QSS: status / update — *Lucio Santi et al.*
Hokkaido University, Room B — 09:00 - 09:20

VecGeom - developments / plans — *Andrei Gheata et al.*
Hokkaido University, Room B — 09:20 - 09:40

Orange modeller: update — *Seth Johnson*
Hokkaido University, Room B — 09:40 - 10:00

10:00

Geometry Converter: an update — *Guilherme Lima*
Hokkaido University, Room B — 10:00 - 10:15

Open Issues — *Dr Gabriele Cosmo*
Hokkaido University, Room B — 10:15 - 10:30

UBA
Universidad de Buenos Aires

DEPARTAMENTO
DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

SED Discrete
Event
Simulation Lab

ICC Instituto de Ciencias
de la Computación

CONICET

Progress in adapting the QSS Stepper to the current version of Geant4

Testing and benchmark results

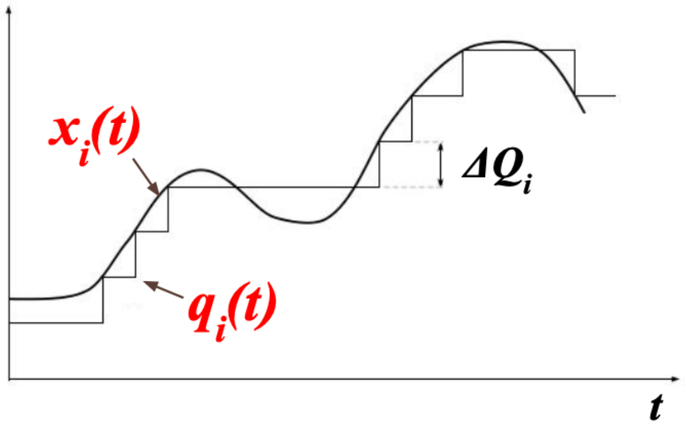
Rodrigo Castro, Lucio Santi, Alejandro Mignanelli

University of Buenos Aires and
ICC-CONICET, Argentina.
rcastro@dc.uba.ar

Quantized State System (QSS) numerical methods

- Based on **state variables quantization**
- QSS methods discretize the system **state variables** as opposed to classical solvers which discretize the **time** (e.g. family of Runge-Kutta methods)
- ***Continuous* state variables** approximated by ***Quantized* state variables**
  - A **quantization function** is in charge of controlling **error and accuracy** throughout the simulation

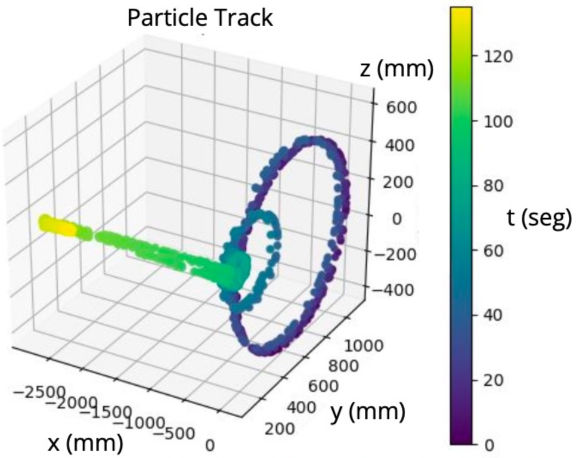
$$\underbrace{\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(\mathbf{t}))}_{\text{ODE system}} \Rightarrow \underbrace{\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{q}(\mathbf{t}))}_{\text{ODE quantized system}}$$



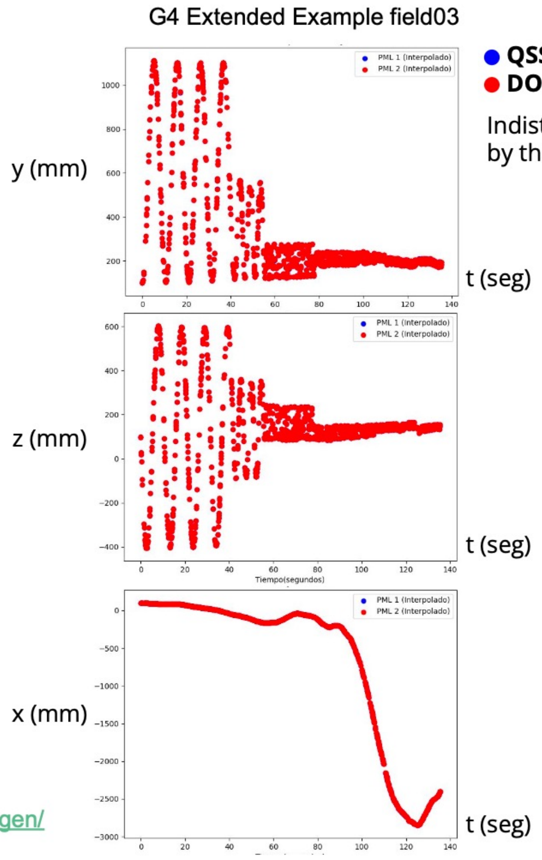
New: Logging of time series for error assessment

- Calculation of the Mean Square Error (MSE) for x(t),y(t),z(t) and the Track Length L(t)
- Thorough systematic comparison of deviation between methods for **different accuracies**
- Interpolation of asynchronous time series
- E.g.: QSS2 vs DOPRI

```
dQRel=1e-5, dQMin=1e-6
X_MSE = 1.64
Y_MSE = 0.00072
Z_MSE = 0.0014
L_MSE = 0.0
```



https://geant4-userdoc.web.cern.ch/Doxygen/examples_doc/html/Examplefield03.html


G4 Extended Example field03

- ● QSS2
- ● DOPRI

Indistinguishable by the naked eye

Results highlights

- 11 examples tested and verified successfully:
    - **Basic** (B2a, B2b, B4c, B4d, B5), **Extended** (with magnetic field: 01, 02, 03, 06), **Advanced** (ams_ECAL)
    - **FullSimLight**, a lightweight standalone Geant4 simulation tool that supports the full ATLAS geometry and the ATLAS magnetic field map
- Benchmarks made against G4 (ver. 11.0.0-ref-02) default stepper (DOPRI with Interpolation Driver)
- In 5 cases *there exist* ***QSS accuracy parameters*** that can outperform DOPRI
    - **However**, the ratio of geometry intersections per G4 step remains below 19% in all tested examples (typically around 5%) => these are **not** "QSS-friendly" scenarios (not too many intersections per step)
- Particle trajectories were compared visually using Paraview and VTK output files
- Benchmarking software: we continue developing a **toolset for repeatable benchmarking** that can be parameterized to produce systematic performance comparisons across G4 Steppers

Benchmark computing platform

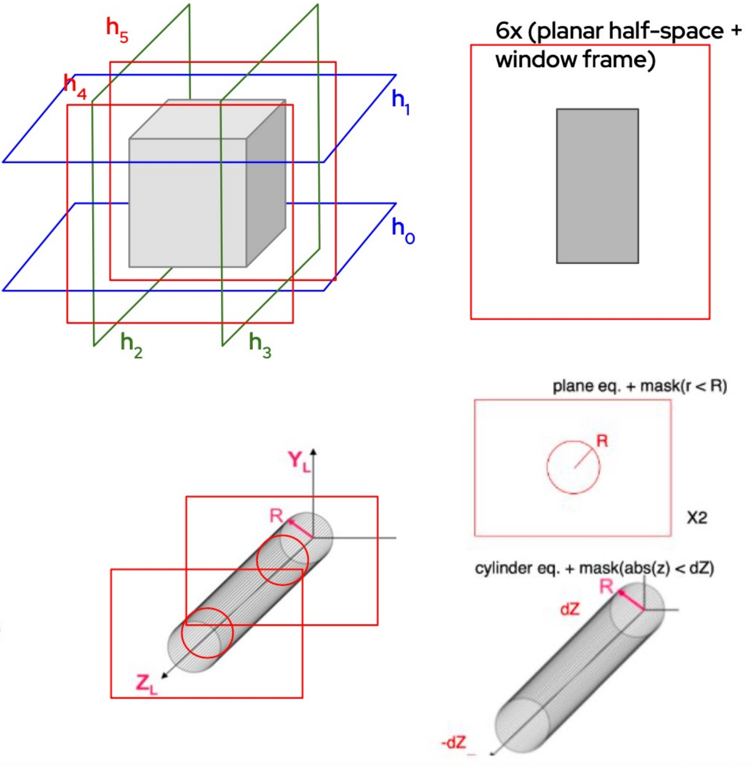
- All experimentations carried out in CERN's OpenLab (controlled environment)
- Hardware specs: Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz (64 CPUs) 64 GB RAM

Surface-based GPU model in VecGeom

Slides prepared by Andrei Gheata for the AdePT team

Bounded surface modeling

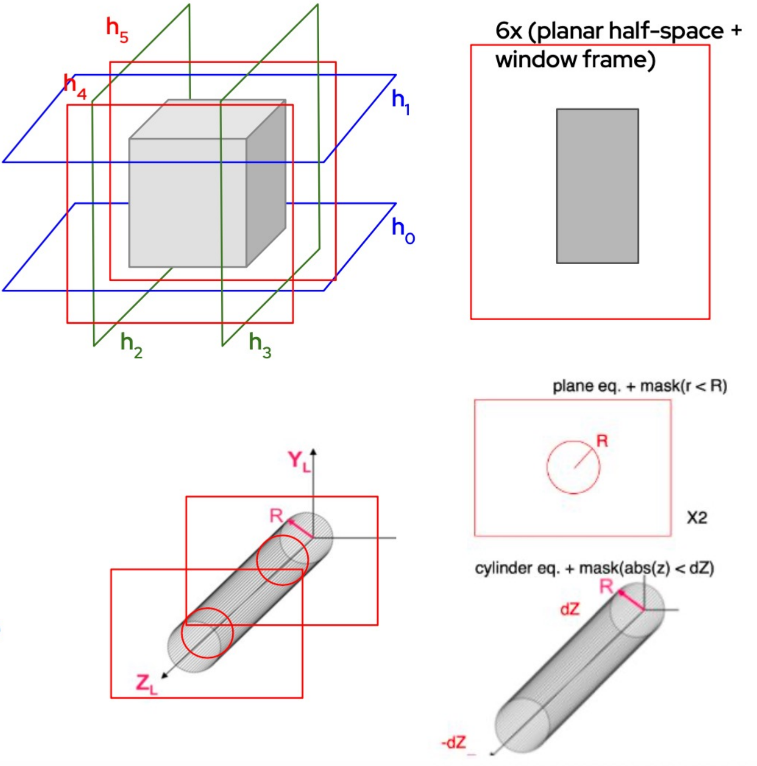
- 3D bodies represented as Boolean operation of half-spaces*
  - First and second order, infinite
  - Just intersections for convex primitives
    - e.g. box = h_0 & h_1 & h_2 & h_3 & h_4 & h_5
  - Similarities with the Orange model
    - Evaluated Orange to start with
- Storing in addition the solid imprint (frame) on each surface: FramedSurface
  - Similarities with detray (ACTS)
  - The frame information allows avoiding to evaluate the Boolean expression for distance calculations to primitive solids



Surface-based GPU model in VecGeom

Bounded surface modeling

- 3D bodies represented as Boolean operation of half-spaces*
  - First and second order, infinite
  - Just intersections for convex primitives
    - e.g. box = h_0 & h_1 & h_2 & h_3 & h_4 & h_5
  - Similarities with the Orange model
    - Evaluated Orange to start with
- Storing in addition the solid imprint (frame) on each surface: FramedSurface
  - Similarities with detray (ACTS)
  - The frame information allows avoiding to evaluate the Boolean expression for distance calculations to primitive solids



Outlook

- As GPU simulation gains in weight and geometry is on critical path, VecGeom develops dedicated surface-based GPU support
  - Surface model enriched with solid frame information
  - Collaboration with Celeritas/ORANGE team on commonalities and convergence paths
  - Transparent implementation, better work-balanced and friendlier to GPU
- Currently implemented all the features required by particle transport, for a subset of solids
  - Locate, relocate, distance and safety calculation
  - Promising preliminary numbers
- Working on extending the model, memory and performance optimization
  - Targeting cms_2018 setup working in AdePT/Celeritas by the end of the year

ORANGE surface geometry progress

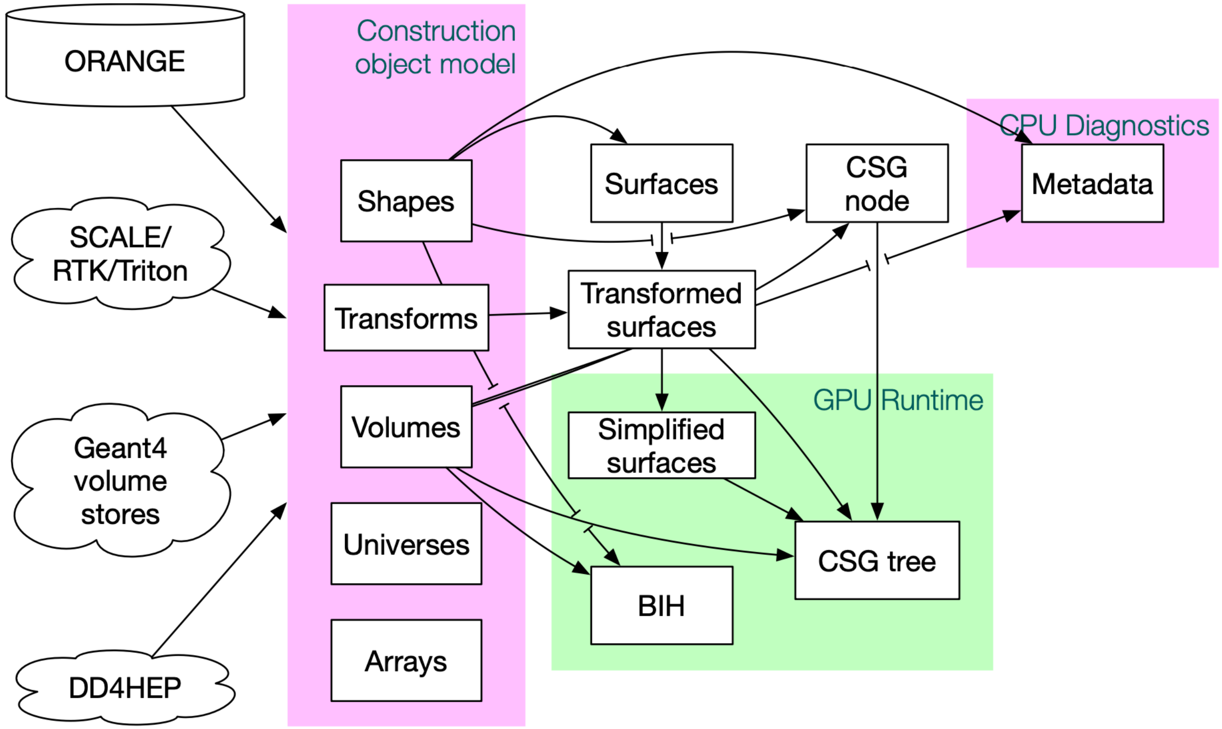
Seth R Johnson, Elliott Biondo, Tom Evans

Celeritas/ORANGE/Shift developers

Background

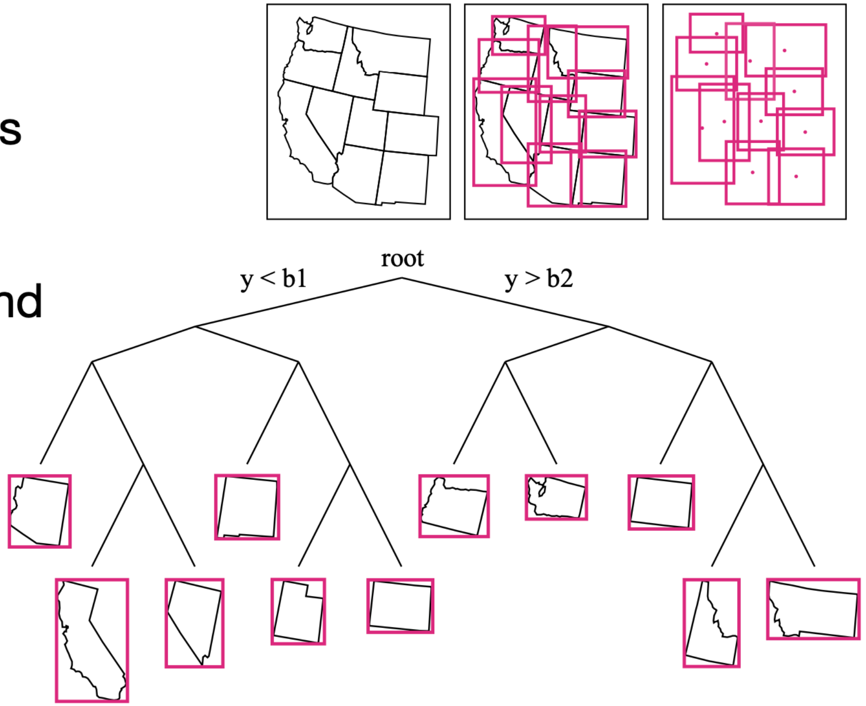
- Many nuclear engineering codes use “unbounded” surfaces and constructive solid geometry
  - MCNP, KENO, earlier codes: >40 years of history
  - Quadric surface cards, CSG cell cards
  - Neutral particles or no magnetic fields
- 2017: Shift GPU code (part of ECP) uses simple but efficient surface-based reactor model (nested cylinders and arrays)
- 2021: Initial GPU port for Celeritas
- 2023: GPU port integrated into Shift

User-to-runtime construction



Acceleration: bounding interval hierarchy

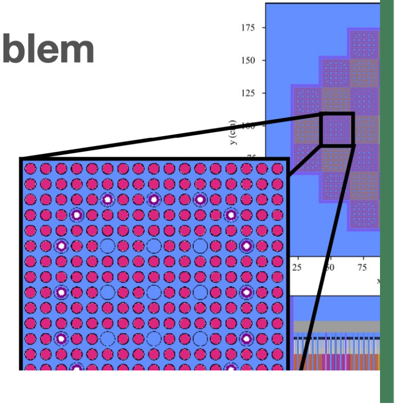
- Inputs: volume bounding boxes
- Recursive partitioning scheme
- Tree traversal at initialization and surface crossing
- Low memory requirements
  - Single-precision bounding boxes
  - Tree nodes are ~16B



Results

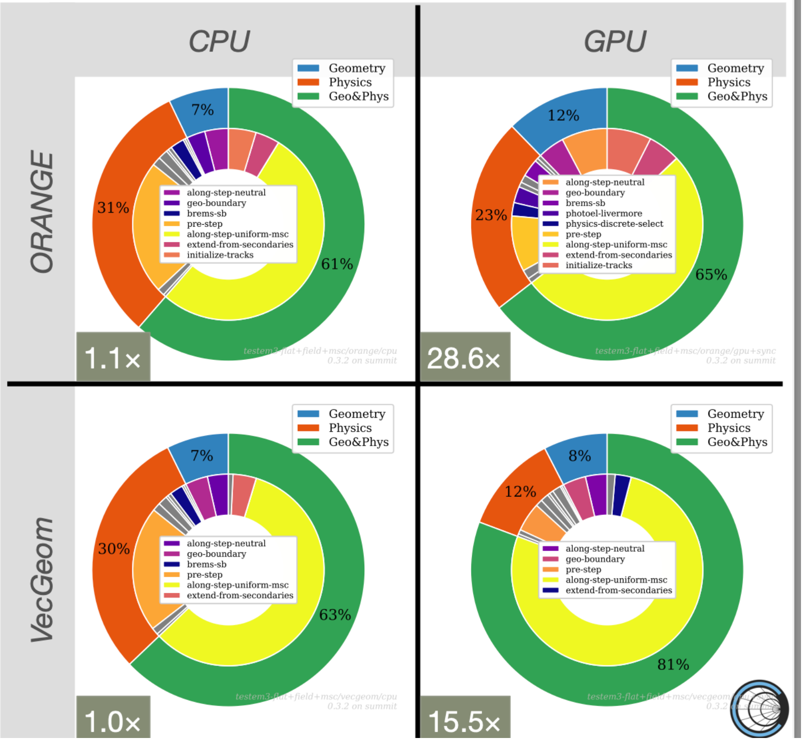
Small Modular Reactor problem

- Array of array of “pin cells”
- Water and uranium
- Neutron-only physics



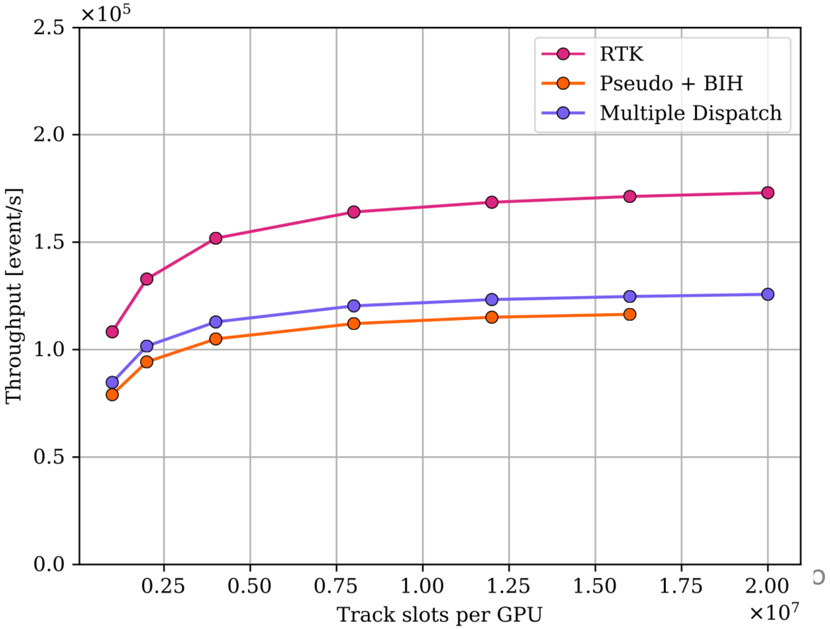
Current ORANGE/VecGeom performance for TestEM3

- VecGeom 1.x navigation on GPU known to be suboptimal
- Results from Summit
  - 7 CPU Power9 cores vs 1 V100 GPU
  - 1T uniform field
  - 1300 × 10 GeV e- per event × 7 events
  - Speedup relative to CPU VecGeom



SMR on Frontier (AMD MI250)

- Template metaprogramming “multiple dispatch” faster on AMD for this problem
- ORANGE is 30% slower than “reactor toolkit” geometry
  - RTK is extremely limited
  - Highlights performance/functionality tradeoff



Next Steps

- Optimisation
  - Reduce memory use by recognising “same” surface
  - Precalculating information for quadratic surfaces
- Extension of capabilities
  - Tracking simultaneously at multiple levels
  - Safety calculation (to determine if beneficial or needed)
  - Aggregate “poly” shapes (e.g. polygon)

In-memory Geometry Converter

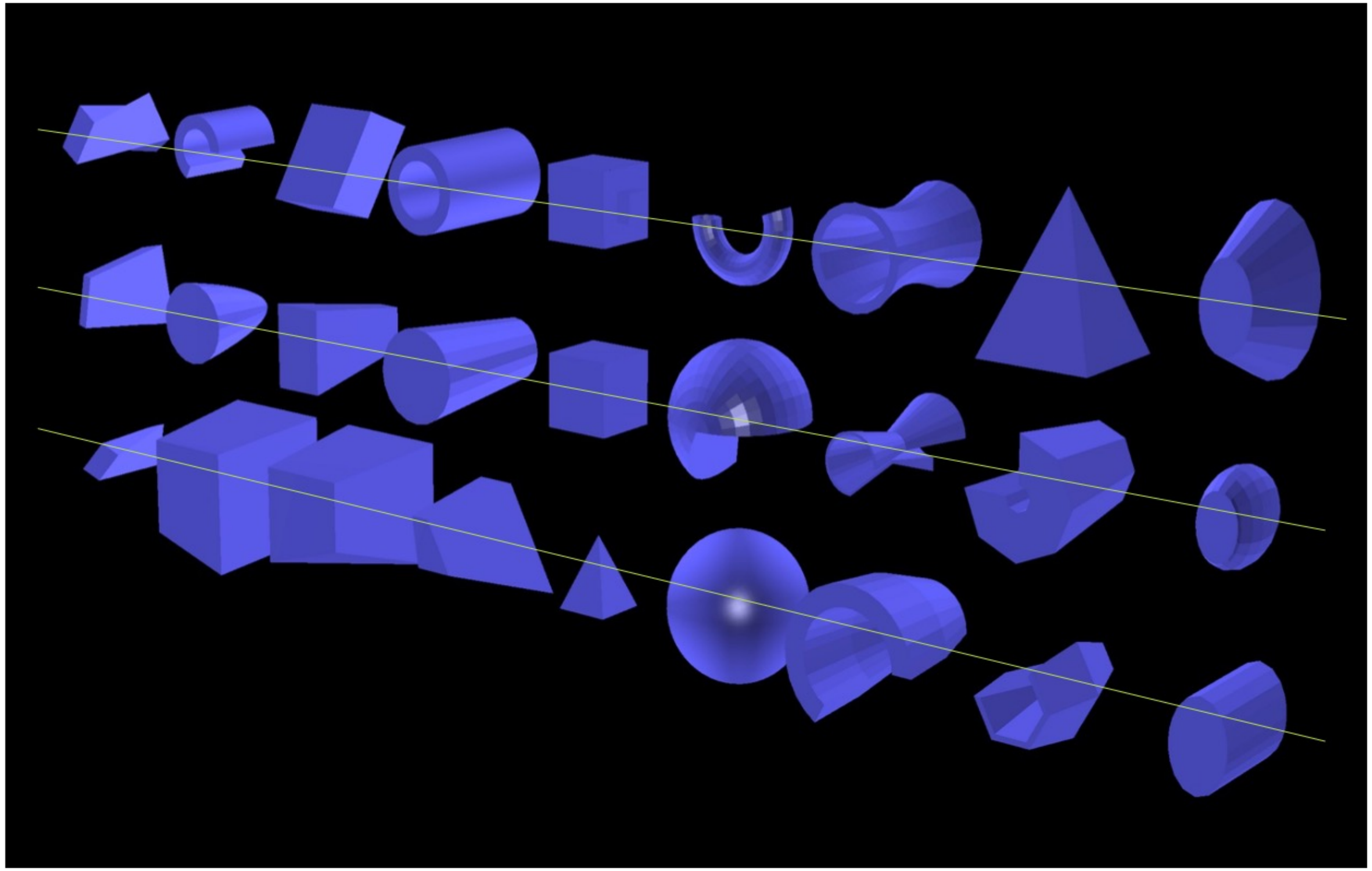
Guilherme Lima

Geometry converter: what is it?

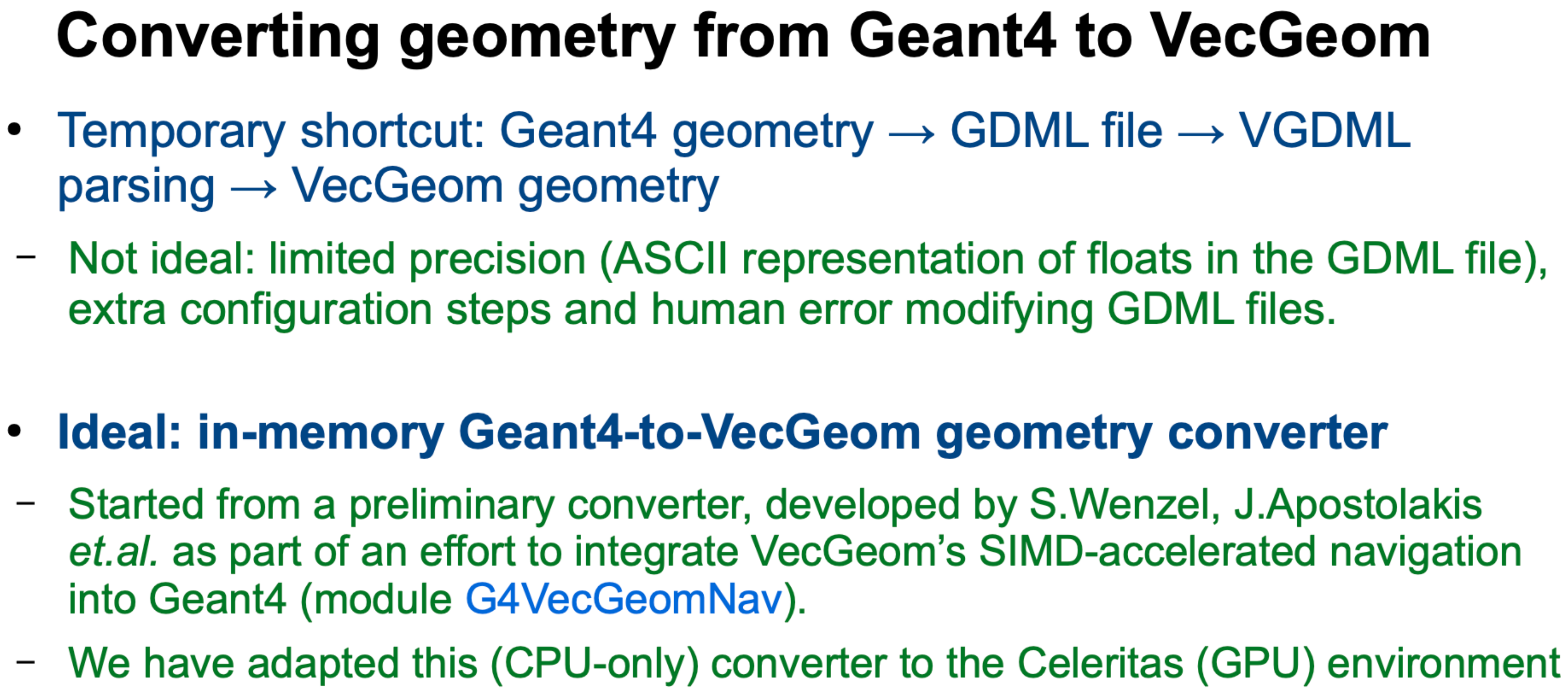
- Converter developed within the context of Celeritas, which goal is to allow a Geant4 job to offload some of the tracking to a GPU device
  - see Seth Johnson's talk for more details on the Celeritas project
- VecGeom (VECtor GEOMetry) was developed to promote SIMD-vectorized algorithms. Since its algorithms could also run well on GPGPUs, it became a natural choice for HEPsim-on-GPU prototypes like Celeritas and AdePT.
  - Celeritas uses VecGeom v1.x for now, at least while surface-based systems (VecGeom 2.x and ORANGE) are being developed
- In order to offload some tracking to the GPU, the Geant4 geometry needs to be made available in VecGeom.

Validation: current status of solids.gdml

- Added support to several shapes
- Similar shape dimensions and adequate positioning and spacing
- Detailed tracking, comparing coordinates of each boundaries crossed



What?



Converting geometry from Geant4 to VecGeom

- Temporary shortcut: Geant4 geometry → GDML file → VGDML parsing → VecGeom geometry
  - Not ideal: limited precision (ASCII representation of floats in the GDML file), extra configuration steps and human error modifying GDML files.

- **Ideal: in-memory Geant4-to-VecGeom geometry converter**
  - Started from a preliminary converter, developed by S.Wenzel, J.Apostolakis *et.al.* as part of an effort to integrate VecGeom's SIMD-accelerated navigation into Geant4 (module G4VecGeomNav).
  - We have adapted this (CPU-only) converter to the Celeritas (GPU) environment

Geometry converter: status and plans

- In-memory Geant4-to-VecGeom geometry converter is now available
  - From a preliminary prototype in G4VecGeomNav, further developed under the Celeritas environment
  - Debugged, fixed, validated and released: produces in-memory VecGeom model
  - The VecGeom model is readily available for tracking in the GPU
  - Has been (partly) ported back into the G4VecGeomNav module
    - Still to be ported: reflected shapes, multi-union, simplifying refactoring

- Prospects:
  - New: surface-based geometry approach, still under development – see previous talks

    → expected to be supported by this converter, no roadblocks anticipated
  - More shapes to be added as needed (e.g. triggered by other complex GDML files used)
  - TBD: long-term repository (requirements, dependencies, constraints)