Interface Geant4 navigation with VecGeom

- Task Overview:
  - Beyond the shape-primitives, make available other features developed in VecGeom to G4 navigation
  - Ultimately target higher CPU performance in a Geant4 simulation
  - See slides from last year
- Several motivations:
  - Use SIMD accelerated data structures for efficient ray – shape queries in Geant4
  - Profit from VecGeom shape factory mechanism and dispatch to best possible runtime instantiation of a shape
  - Possibility to dispatch to per volume-specialized algorithms (code generated plugins, lookup-tables) instead of generic G4Voxel treatment
  - Access industry-standard algorithms such as Intel Embree via VecGeom
  - For ALICE: Use VecGeom navigation instead of TGeo navigation in their VMC simulation
- Approach taken:
  - Subclass G4Navigator and override navigation functions to dispatch to VecGeom API

Overview/status of main developments

Current development variants	Description	Status	Comments
Override **all virtual G4Navigator functions** in new navigator class	**Complete replacement** of geometry calls to VecGeom functionality. • Similar to existing T4RootNavigator dispatching to Root/TGeo.	**Basic version available** • Works well for test particles (geantinos, charged geantinos) • Some problems in relocation across boundaries when field enabled. • Needs improved "state" transfer and use across various APIs.	**Original goal (especially for ALICE)** From experience … a hard development task (Look at how intricate the state and case handling in G4Navigator is)
Override **some virtual G4Navigator functions** in new navigator class	**Partial replacement** of geometry calls to VecGeom functionality. • May mix best of 2 worlds.	**Demonstrator version available dispatching ComputeSafety to VecGeom**	Probably only works for safety because much of the important state information in G4Navigator (is currently) is private and not accessible to child classes.
Override **some or all virtual functions of** helper classes such as **"G4VoxelNavigation"**	**Partial replacement** of geometry calls to VecGeom functionality. • May mix best of 2 worlds • Can reuse existing case/state handling in G4Navigator because state transfer between G4Navigator and helpers is nicely encapsulated.	**Version available overriding the ComputeStep function of G4VoxelNavigation** • Seems to work nicely with test cases • Needs least synchronization between G4 and VecGeom objects	**Maybe the "sweet spot".**

Testing framework developments, plans

In order to test and validate the initial navigation developments:

- **A test geometry** was created which is automatically scalable in complexity (number of daughters) as well as in content (solids, materials)
  - Geometry is simple enough to spot problems and make comparisons across navigators (to not get lost in boundary cases)

- **An analysis and instrumentation workflow** was created to:
  - Automatically perform the same simulation:
    - using various existing navigators (G4 native, TGeo, new VecGeom developments)
    - For various configurations of (geometry setup x PDG x field)
  - Automatically extract high-level plots (comparisons of step numbers, step values, cycle count)

Development focus is to have a coherent picture for CHEP19 conference