Geometry & Persistency
Recent & ongoing developments

Gabriele Cosmo, CERN EP/SFT

for the Geant4 Geometry & Persistency Working Groups

Contents

- Development and fixes in the last year
  - Introduced in release 10.3 and patches
- Features expected in Geant4 10.4
  - Currently under development and scheduled for inclusion in the next release

Geometry

Solids

10.3

- Added ability to scale shapes along Cartesian axes
  - New class *G4ScaledSolid* providing ability to scale a shape in dimensions along Cartesian axes X, Y or Z
- Reviewed algorithms for computation of extent of all shapes
  - Providing more precise and efficient voxelisation and memory reduction
  - Applied also to USolids/VecGeom wrappers for generation of same voxelisation structure
  - Allowing for exact comparison of tracking between VecGeom and Geant4 original solids
  - See detailed presentation in parallel session 6B
- Fix to *SurfaceNormal()* in G4SubtractionSolid to consider rare cases when a point is not located on the surface

10.3.p01

- Fix in *GetCubicVolume()* and *GetSurfaceArea()* for *G4CutTubs*, to make use of the cached values; also fix on Inside() for case where inside points were considered as points on surface

10.3.p02

- Boosted computation of *GetPointOnSurface()* for Boolean shapes
- Optimisation in *G4SubtractionSolid* to directly return previously computed distance in *DistanceToIn(p,v)* if no progress is made (zero step)
- Revised implementation of GetSurfaceArea() for G4EllipticalCone, to return exact value

10.3.p01

- Improved precision in computation of distances for *G4Torus*

Improved calculation of Extent

10.3

- Motivated to allow for exact comparison of intersections between Geant4 and VecGeom shapes
- Each Geant4 solid has a member function **`CalculateExtent()`** which calculates the minimum and maximum extent of the solid under the specified transform and within the specified limits. This function is used for building the voxel structure for optimisation of the geometry

```
G4bool CalculateExtent(const EAxis pAxis,
                       const G4VoxelLimits& pVoxelLimit,
                       const G4AffineTransform& pTransform,
                       G4double& pMin, G4double& pMax) const;
```

- To facilitate and factorise implementation of this function in the different solids a new class **`G4BoundingEnvelope`** has been introduced
  - "Bounding envelope" is a sequence of 3D convex polygons that bounds the solid. The polygons should have equal number of vertices except first and last polygons which may consist of a single vertex
  - **`G4BoundingEnvelope`** has a method for computing the extent, so there is no longer need to implement calculation of extent explicitly inside the solid
- Computation of extent more precise, leading to more compact memory for voxelisation and faster tracking and initialisation time

Improved Extent: impact to voxelisation

10.3

`CalculateExtent()` method has been re-implemented for all solids

- includes 10 CGS solids, 16 specific solids as well as reflected, scaled and Boolean constructs
- improved voxel structure with default voxels parameters: reduced memory footprint and faster optimization time
- improvement in the performance of tracking (~2% measured in pure ray-tracing “geantino” tests)

Geant4 10.2.p03
CMS setup:
Total memory consumed for geometry optimization: 22209 kByte
Total CPU time elapsed for geometry optimization: 4.6 seconds

ATLAS setup:
Total memory consumed for geometry optimization: 290665 kByte
Total CPU time elapsed for geometry optimization: 110 seconds

Geant4 10.3.p02
CMS setup:
Total memory consumed for geometry optimization: 16418 kByte (5.8 MByte - 26% less)
Total CPU time elapsed for geometry optimization: 3.5 seconds (1.1 s faster)

ATLAS setup:
Total memory consumed for geometry optimization: 272640 kByte (18 MByte - 6%less)
Total CPU time elapsed for geometry optimization: 89 seconds (21 s faster)

More on Solids

Expected in release 10.4

- Introduction of G4MultiUnion structure — 10.4-β
- Reviewed implementation of *G4Box, G4Trap, G4Trd, G4Para, G4EllipticalCone, G4Orb*; made more compact and performant — 10.4-β
  - See detailed presentation in parallel session 6B
- Improved interface to VecGeom shapes — 10.4-β
  - Getting rid of dependency on USolids implementation from release 10.4
  - See detailed presentation on VecGeom in parallel session 6B
- Added specialised implementation in *G4ExtrudedSolid* for constructs defining a convex right prism — 10.3.ref09
  - CMS requirement

Revised implementation of shapes

10.4-β

- Most of Geant4 solids implemented long time ago
  - now, better compilers with more stable, performant and advanced optimizers
  - now, enhanced implementation of math functions
  - e.g. `std::min(a,b)`, `std::max(a,b)` no longer template functions but processor instructions
- Several places where code can be improved and simplified or simply reshuffled to benefit better caching of temporaries

```
EInside G4Box::Inside(const G4ThreeVector& p)
const {
   EInside in = kOutside ;
   G4ThreeVector q(std::abs(p.x()), std::abs(p.y()),
std::abs(p.z()));
   if ( q.x() <= (fDx - delta) ) {
      if (q.y() <= (fDy - delta) ) {
         if ( q.z() <= (fDz - delta) ) {
            in = kInside ;
         } else if ( q.z() <= (fDz + delta) ) {
            in = kSurface ;
         }
      } else if ( q.y() <= (fDy + delta) ) {
         if ( q.z() <= (fDz + delta) ) {
            in = kSurface ;
         }
      }
   } else if ( q.x() <= (fDx + delta) ) {
      if ( q.y() <= (fDy + delta) ) {
         if ( q.z() <= (fDz + delta) ) {
            in = kSurface ;
         }
      }
   }
   return in ;
}
```

```
EInside G4Box::Inside(const G4ThreeVector& p)
const {
   G4double dist = std::max(std::max(
                        std::abs(p.x())-fDx,
                        std::abs(p.y())-fDy),
                        std::abs(p.z())-fDz);
   if (dist > delta) return kOutside;
   return (dist > -delta) ? kSurface : kInside;
}
```

Speed-up up to 3 times!

Revised implementation of shapes - 2

10.4-β

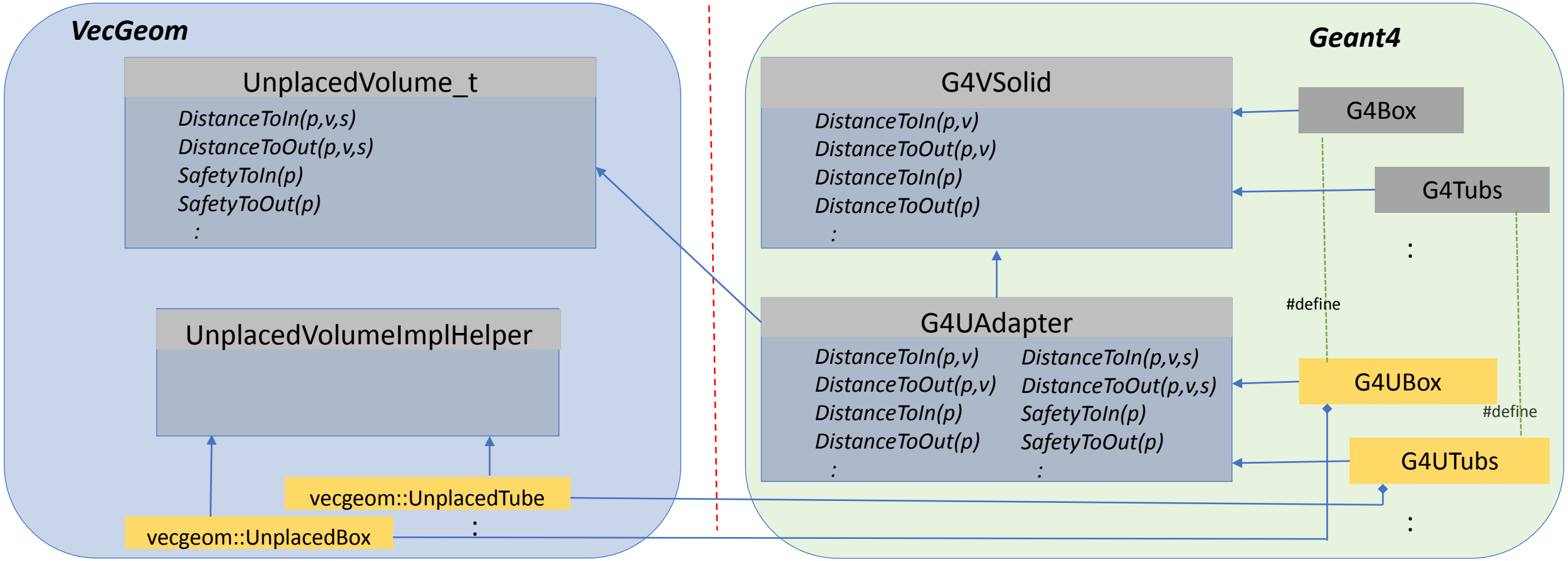
G4Box, G4Trd, G4Para, G4Trap have been revised
Benchmark: 100 million calls, MacBook Pro, 2.7 GHz Intel Core i5

	Box		**Trd**		**Para**		**Trap**	
	new	old	new	old	new	old	new	old
Inside							**Type 1 / Type2 / General**	
kInside	**0.39 s**	0.39 s	**0.51 s**	0.58 s	**0.59 s**	0.48 s	**0.47 / 0.61 /0.71 s**	1.06 s
kSurface	**0.39 s**	1.09 s	**0.51 s**	1.51 s	**0.60 s**	1.27 s	**0.47 / 0.61 /0.70 s**	1.37 s
kOutside	**0.32 s**	0.77 s	**0.40 s**	0.89 s	**0.50 s**	0.78 s	**0.40 / 0.56 /0.60 s**	0.88 s
Safety to In	**0.31 s**	0.35 s	**0.42 s**	2.13 s	**0.50 s**	0.94 s	**0.39 / 0.58 / 0.60 s**	0.81 s
Safety to Out	**0.34 s**	0.52 s	**0.45 s**	1.16 s	**0.52 s**	0.91 s	**0.42 / 0.58 / 0.63 s**	0.84 s
Distance To In	**1.15 s**	2.96 s	**2.07 s**	3.21 s	**2.15 s**	2.26 s	**2.48 s**	4.03 s
Distance To Out	**1.79 s**	3.61 s	**2.59 s**	4.53 s	**2.75 s**	4.06 s	**3.06 s**	4.49 s
Surface Normal	**0.60 s**	1.62 s	**1.59 s**	3.22 s	**1.71 s**	2.27 s	**1.79 s**	1.98 s

Type 1 – rectangular YZ cross-section + symmetrical X-sides
Type 2 – just rectangular YZ cross-section

Improved interface to VecGeom solids

10.4-β



- Wrappers in Geant4 making VecGeom shapes look like native Geant4 solids types
- Adapter in Geant4 wrapping VecGeom implementations (improved design expected in release 10.4)
- Completely transparent to users – i.e. requiring no changes in users code

Volumes & Navigation

- New classes representing an extended logical volume for crystal description — 10.3
- Optimisation fix in G4NavigationHistory default constructor to favor reuse of already allocated space from the navigation-history pool — 10.3.p01
- Fix in *G4Navigator::GetGlobalExitNormal()* to synchronise caching of '*fExitNormalGlobalFrame*' before returning. Addressing problem report #1750 — 10.3
- Implemented proper destruction of geometry objects in MT-mode — 10.4-β
  - Extended splitter classes and corrected destructors in field & manager classes

New field steppers & integrator drivers

10.3

- New stepper classes *G4BogackiShampine23* (BS23), *G4BogackiShampine45* (BS45) and *DormandPrince745* (DP45), implementing third order (BS23) and fifth order (BS45, DP45) embedded RK
- New stepper classes embedded RK method: *DoLoMcPriRK34* (6-stage 3/4 RK, interpolation), *DormandPrinceRK56* (9-stage 5/6 RK, interpolation, FSAL-able), *DormandPrinceRK78* (13-stage 7/8 RK, interpolation) and *TsitourasRK45* stepper
- First version of FSAL classes: FSAL Integrator Driver (concrete, stand-alone driver); FSAL Integrator Stepper (base class); *FBogackiShampine45* (FSAL-version of *BogackiShampine45* stepper); *FDormandPrince745* (FSAL-version of *DormandPrince745* stepper)
- *G4MagIntegratorStepper*: added counter for calls to equation RHS, with *Get/Reset()* methods
- *G4HelixMixedStepper*: fixes and added new (5th order) stepper choices

➢ See detailed presentation on validation in parallel session 6B

More on field…

- Revised behavior of SetDetector() in G4FieldManager — 10.4-β
  - Now passing the Field to the Equation, as is the natural user expectation
- Fixed caching of momentum, field location & value in *G4NystromRK4* stepper — 10.3.ref08
  - Problem reported by ALICE

Persistency

GDML module

- Added ability to import and export scaled shapes (GDML 3.1.4) — 10.3
- Added ability to automatically export the names of sensitive detectors as auxiliary information — 10.3
- Fixed schema for replicas along angular axis — 10.4-β
- Added possibility to specify material properties tables for optical surfaces; added general optical surface properties block to GDML schema (GDML 3.1.5) — 10.4-β
- Added ability to export limited number of levels in the geometry hierarchy, by specifying this through the *G4GDMLParser::SetMaxExportLevel(G4int)* method — 10.4-β
- Added commands to *G4GDMLMessenger* for enable/disable stripping of names for reading and for appending or not pointers to names for writing — 10.3.ref07
- Handle all possible variants for *G4TwistedTubs*, included asymmetric ones. Fixed export of shape to correctly export end-inner/outer radii — 10.3.ref08
  - Extended schema for coping with all possible parameters for G4TwistedTubs (GDML 3.1.6)

Expected geometry features in Geant4 10.4

Geometrical primitives

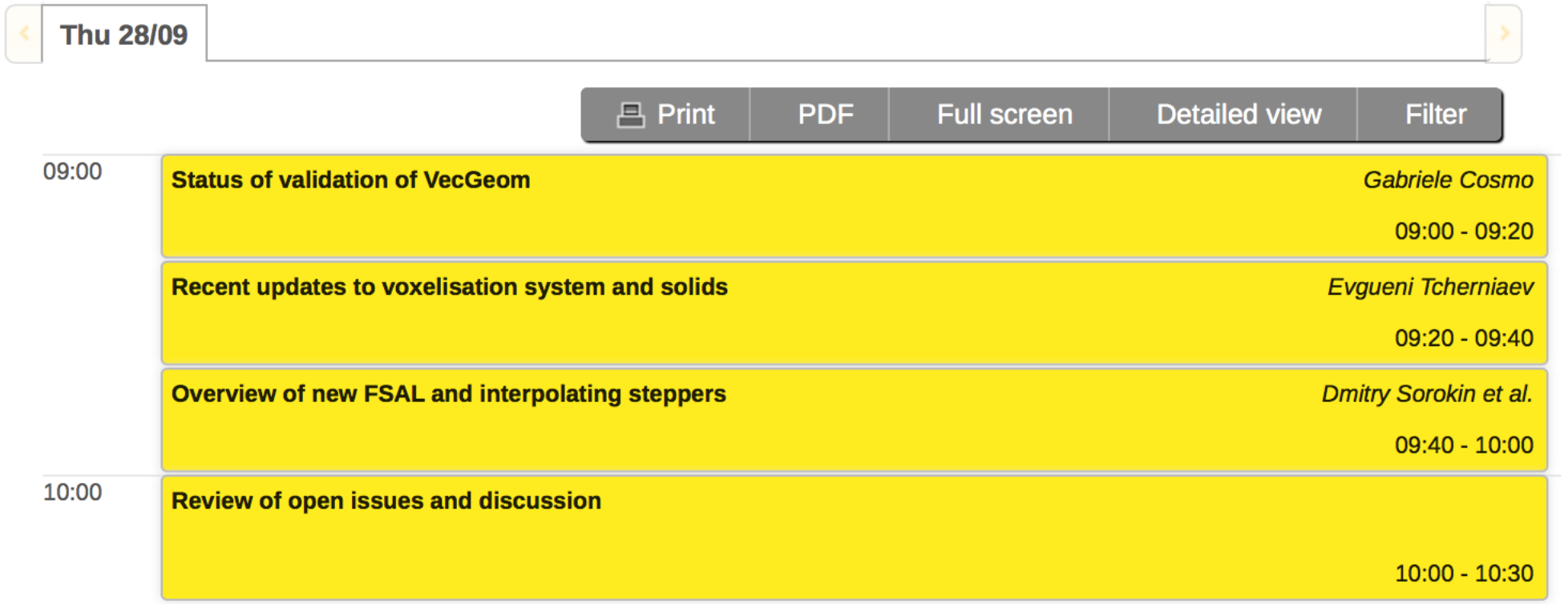
- Simplify adoption of VecGeom shapes with direct interface to VecGeom from wrappers
  - Phase-out of USolids dependency; deleted extra layer (USolids => VecGeom) when using VecGeom shapes
  - Progress in the implementation of the Unified Solids library with progressive adoption of shapes from VecGeom
- More robustness tests of VecGeom shapes on realistic geometry setups
  - Exact tracking comparison with original Geant4 shapes implementation
  - Including new snapshots from LHC detectors
- Multi-union structure as native type in Geant4

Navigation & field

- Separation of safety computation and state from navigator
- Profiling and optimisation of multiple navigation
- Prototype study of specialized navigator with VecGeom
  - Applicable to placed/static geometries only
- Validation of FSAL/interpolation in field propagation

Code improvements: use of C++11/14 constructs in key areas

Thanks !


Thu 28/09
Print
PDF
Full screen
Detailed view
Filter
09:00
Status of validation of VecGeom
Gabriele Cosmo
09:00 - 09:20
Recent updates to voxelisation system and solids
Evgueni Tcherniaev
09:20 - 09:40
Overview of new FSAL and interpolating steppers
Dmitry Sorokin et al.
09:40 - 10:00
10:00
Review of open issues and discussion
10:00 - 10:30